Creating a Matrix using 2D vector in C++ – Vector of Vectors

Varun November 27, 2015 std::vector, STL No Comment

In this article will discuss how to create 2D Matrix using vector of vectors in c++.

Requirement

Represent a 2D Matrix using vector in C++ i.e.

1 , 1 , 1 , 1
1 , 1 , 1 , 1
1 , 1 , 1 , 1
1 , 1 , 1 , 1
1 , 1 , 1 , 1

Declaration of 2D vector or vector of vector in C++

```
1 std::vector <std::vector<int> > vec2D
```

Initializing Vector of Vector – 2D vector

A vector can be initialized using parametrized constructor i.e.

std::vector <NUMBER OF ELEMENTS, VALUE OF EACH ELEMENT>

So,

```
std::vector<int> (4, 1)
```

Will create a vector of 4 integers whose values will be 1.

Now to create a vector of 5 vectors in which each vector is initialized as above, we will use following syntax,

```
std::vector <std::vector > vec2D(5, std::vector(4, 1));
```

Let's see the code to initialize and print 2D vector as follows,

```
std::vector <std::vector<int> > vec2D(5, std::vector<int>(4, 1))

for(auto vec : vec2D)
{
    for(auto x : vec)
        std::cout<<x<<" , ";
    std::cout << std::endl;
}
```

Output

1 , 1 , 1 , 1 ,
1 , 1 , 1 , 1 ,
1 , 1 , 1 , 1 ,
1 , 1 , 1 , 1 ,
1 , 1 , 1 , 1 ,

Iterator over 2D vector in C++

We can iterate over a vector of vector using [][] . Checkout the code below,

```
for(int i = 0; i < 5; i++)
    for(int j = 0; j < 5; j++)
        vec2D[i][j] = i*j;
```

Adding a new row in 2D vector

To add a new row, just push_back a new vector in the vector of vector i.e.

```
vec2D.push_back(std::vector<int>(4, 11));
```

Complete working Code is as follows,

```C++
#include <vector>

int main()
{
    std::vector <std::vector<int> > vec2D(5, std::vector<int>(4,

    for(auto vec : vec2D)
    {
        for(auto x : vec)
            std::cout<<x<<" , ";

        std::cout << std::endl;
    }

    std::cout << std::endl;

    for(int i = 0; i < 5; i++)
        for(int j = 0; j < 5; j++)
            vec2D[i][j] = i*j;

    for(auto vec : vec2D)
    {
        for(auto x : vec)
            std::cout<<x<<" , ";

        std::cout << std::endl;
    }


    vec2D.push_back(std::vector<int>(4, 11));

    std::cout << std::endl;

    for(auto vec : vec2D)
    {
        for(auto x : vec)
            std::cout<<x<<" , ";

        std::cout << std::endl;
    }
    return 0;
}
```

Compile the above code using following command,

g++ –std=c++11 2dvector.cpp

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

Related Posts:

- How to add an element in Vector using vector::push_back

- deque vs vector : What to choose ?
- Different Ways to Initialize a vector in C++
- C++ std::vector example and why should I use std::vector?
- 5 Different ways to Initialize a vector in c++
- c++ std::vector and Iterator Invalidation example
- Remove all occurences of an element from vector in...
- How to copy all Values from a Map to a Vector in C++
- How to search by value in a Map | C++
- How to read data from a csv file in C++ ?
- C++11: How to create Vector of Thread Objects ?
- std::initializer_list Tutorial & Examples | C++11
- How to use vector efficiently in C++?
- How to fill a vector with random numbers in C++
- How does std::vector works internally ?
- C++ : How to Initialize a map in one line using...
- Importance of Constructors while using User Defined...
- Iterating over a range of User Defined objects and...
- C++ : How to read a file line by line into a vector ?
- How to initialize an unordered_set in C++11
- std::for_each Tutorial : Usage Details with Examples
- Difference between Vector and List in C++
- std::generate Tutorial and example
- Different ways to Initialize a list in C++
- C++11 : std::any_of() Examples and Tutorial

2D vector, C++, c++11, std::vector, vector of vectors

No Comments Yet

Leave a Reply

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

Name *

Email *

Website

Post Comment

« How to trim strings in C++ using Boost String Algorithm Library

Different Ways to Initialize a vector in C++ »

Like us on Facebook

Like Share 449 peo

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

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 & Comparator

6.) Unordered_set & User defined classes

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

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

10.) Adding elements in Vector

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

Subscribe

Subscribe For latest Tutorials

* indicates required

Email Address *

Subscribe

Design Patterns

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

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

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

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

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

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

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()

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

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