Array of Strings in C++ – 5 Different Ways to Create

Difficulty Level : Easy • Last Updated : 11 Mar, 2023

In C++, a string is usually just an array of (or a reference/points to) characters that ends with the NULL character '**\0**'. A string is a 1-dimensional array of characters and an array of strings is a 2-dimensional array of characters where each row contains some string.

Below are the 5 different ways to create an Array of Strings in C++:

1. Using **Pointers**
2. Using **2-D Array**
3. Using the **String Class**
4. Using the **Vector Class**
5. Using the **Array Class**

1. Using Pointers

Pointers are the symbolic representation of an address. In simple words, a pointer is something that stores the address of a variable in it. In this method, an array of string literals is created by an array of pointers in which each pointer points to a particular string.

Example:

C++

```
// C++ program to demonstrate
// array of strings using
// pointers character array
#include <iostream>

// Driver code
int main()
{
  // Initialize array of pointer
  const char* colour[4]
      = { "Blue", "Red", "Orange", "Yellow" };

  // Printing Strings stored in 2D array
  for (int i = 0; i < 4; i++)
      std::cout << colour[i] << "\n";

  return 0;
}
```

Output

```
Blue
Red
Orange
Yellow
```

Explanation:

- The number of strings is fixed, but needn't be. The 4 may be omitted, and the compiler will compute the correct size.
- These strings are constants and their contents cannot be changed. Because string literals (literally, the quoted strings) exist in a read-only area of memory, we must specify "const" here to prevent unwanted accesses that may crash the program.

2. Using a 2D array

A 2-D array is the simplest form of a multidimensional array in which it stores the data in a tabular form. This method is useful when the length of all strings is known and a particular memory footprint is desired. Space for strings will be allocated in a single block

Example:

C++

```
// C++ program to demonstrate
// array of strings using
// 2D character array
#include <iostream>

// Driver code
int main()
{
  // Initialize 2D array
  char colour[4][10]
      = { "Blue", "Red", "Orange", "Yellow" };

  // Printing Strings stored in 2D array
  for (int i = 0; i < 4; i++)
      std::cout << colour[i] << "\n";

  return 0;
}
```

Output

```
Blue
Red
Orange
Yellow
```

Explanation:

- Both the number of strings and the size of the strings are fixed. The 4, again, may be left out, and the appropriate size will be computed by the compiler. The second dimension, however, must be given (in this case, 10), so that the compiler can choose an appropriate memory layout.
- Each string can be modified but will take up the full space given by the second dimension. Each will be laid out next to each other in memory, and can't change size.
- Sometimes, control over the memory footprint is desirable, and this will allocate a region of memory with a fixed, regular layout.

3. Using the String class

The STL string or string class may be used to create an array of mutable strings. In this method, the size of the string is not fixed, and the strings can be changed which somehow makes it dynamic in nature nevertheless **std::string** can be used to create a string array using in-built functions.

Example:

C++

```
// C++ program to demonstrate
// array of strings using
// string class
#include <iostream>
#include <string>

// Driver code
int main()
{
  // Initialize String Array
  std::string colour[4]
      = { "Blue", "Red", "Orange", "Yellow" };

  // Print Strings
  for (int i = 0; i < 4; i++)
      std::cout << colour[i] << "\n";
}
```

Output

```
Blue
Red
Orange
Yellow
```

Explanation:

The array is of fixed size, but needn't be. Again, the 4 here may be omitted, and the compiler will determine the appropriate size of the array. The strings are also mutable, allowing them to be changed.

4. Using the vector class

A **vector** is a dynamic array that doubles its size whenever a new character is added that exceeds its limit. The STL container vector can be used to dynamically allocate an array that can vary in size.

This is only usable in C++, as C does not have classes. Note that the initializer-list syntax here requires a compiler that supports the 2011 C++ standard, and though it is quite likely your compiler does, it is something to be aware of.

Example:

C++

```
// C++ program to demonstrate
// array of strings using
// vector class
#include <iostream>
#include <string>
#include <vector>

// Driver code
int main()
{
  // Declaring Vector of String type
  // Values can be added here using
  // initializer-list
  // syntax
  std::vector<std::string> colour{"Blue", "Red",
                                  "Orange"};

  // Strings can be added at any time
  // with push_back
  colour.push_back("Yellow");

  // Print Strings stored in Vector
  for (int i = 0; i < colour.size(); i++)
      std::cout << colour[i] << "\n";
}
```

Output

```
Blue
Red
Orange
Yellow
```

Explanation:

- Vectors are dynamic arrays and allow you to add and remove items at any time.
- Any type or class may be used in vectors, but a given vector can only hold one type.

5. Using the Array Class

An array is a homogeneous mixture of data that is stored continuously in the memory space. The STL container array can be used to allocate a fixed-size array. It may be used very similarly to a vector, but the size is always fixed.

Example:

C++

```
// C++ program to demonstrate
// array of string using STL array
#include <array>
#include <iostream>
#include <string>

// Driver code
int main()
{
  // Initialize array
  std::array<std::string, 4> colour{"Blue", "Red",
                                    "Orange", "Yellow"};

  // Printing Strings stored in array
  for (int i = 0; i < 4; i++)
      std::cout << colour[i] << "\n";

  return 0;
}
```

Output

```
Blue
Red
Orange
Yellow
```

These are by no means the only ways to make a collection of strings. C++ offers several container classes, each of which has various tradeoffs and features, and all of them exist to fill requirements that you will have in your projects. Explore and have fun!

Conclusion: Out of all the methods, Vector seems to be the best way for creating an array of Strings in C++.

This article is contributed by **Kartik Ahuja**. If you like GeeksforGeeks and would like to contribute, you can also write an article using write.geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks. Please write comments if you find anything incorrect, or if you want to share more information about the topic discussed above.s.