std::string class in C++

Difficulty Level : Easy   •   Last Updated : 07 Jan, 2022

C++ has in its definition a way to represent a **sequence of characters as an object of the class**. This class is called std:: string. String class stores the characters as a sequence of bytes with the functionality of allowing **access to the single-byte character**.

std:: string vs Character Array:

- A character array is simply an **array of characters** that can be terminated by a null character. A string is a **class that defines objects** that be represented as a stream of characters.

- The size of the character array has to be **allocated statically**, more memory cannot be allocated at run time if required. Unused allocated **memory is wasted** in the case of the character array. In the case of strings, memory is **allocated dynamically**. More memory can be allocated at run time on demand. As no memory is preallocated, **no memory is wasted**.

- There is a **threat of array decay** in the case of the character array. As strings are represented as objects, **no array decay** occurs.

- Implementation of **character array is faster** than std:: string. **Strings are slower** when compared to implementation than character array.

- Character arrays **do not offer** many **inbuilt functions** to manipulate strings. String

Start Your Coding Journey Now!

Function	Definition
getline()	This function is used to store a stream of characters as entered by the user in the object memory.
push_back()	This function is used to input a character at the end of the string.
pop_back()	Introduced from C++11(for strings), this function is used to delete the last character from the string.

Let's look at the following example,

CPP

```cpp
// C++ Program to demonstrate the working of
// getline(), push_back() and pop_back()
#include <iostream>
#include <string> // for string class
using namespace std;

// Driver Code
int main()
{
    // Declaring string
    string str;

    // Taking string input using getline()
    getline(cin, str);

    // Displaying string
    cout << "The initial string is : ";
    cout << str << endl;

    // Inserting a character
    str.push_back('s');
```

Start Your Coding Journey Now!

```
    return 0;
}
```

Input

```
geeksforgeek
```

Output

```
The initial string is : geeksforgeek
The string after push_back operation is : geeksforgeeks
The string after pop_back operation is : geeksforgeek
```

2) Capacity Functions:

Function	Definition
capacity()	This function returns the capacity allocated to the string, which can be equal to or more than the size of the string. Additional space is allocated so that when the new characters are added to the string, the operations can be done efficiently.
resize()	This function changes the size of the string, the size can be increased or decreased.
length()	This function finds the length of the string.
shrink_to_fit()	This function decreases the capacity of the string and makes it equal to the minimum capacity of the string. This operation is useful to save additional memory if we are sure that no further addition of

```cpp
#include <iostream>
#include <string> // for string class
using namespace std;

// Driver Code
int main()
{
    // Initializing string
    string str = "geeksforgeeks is for geeks";

    // Displaying string
    cout << "The initial string is : ";
    cout << str << endl;

    // Resizing string using resize()
    str.resize(13);

    // Displaying string
    cout << "The string after resize operation is : ";
    cout << str << endl;

    // Displaying capacity of string
    cout << "The capacity of string is : ";
    cout << str.capacity() << endl;

    // Displaying length of the string
    cout << "The length of the string is :" << str.length()
        << endl;

    // Decreasing the capacity of string
    // using shrink_to_fit()
    str.shrink_to_fit();

    // Displaying string
    cout << "The new capacity after shrinking is : ";
    cout << str.capacity() << endl;

    return 0;
}
```

Output

Start Your Coding Journey Now!

Function	Definition
begin()	This function returns an iterator to the beginning of the string.
end()	This function returns an iterator to the end of the string.
rbegin()	This function returns a reverse iterator pointing at the end of the string.
rend()	This function returns a reverse iterator pointing at beginning of the string.

Let's look at the following example:

CPP

```cpp
// C++ Program to demonstrate the working of
// begin(), end(), rbegin(), rend()
#include <iostream>
#include <string> // for string class
using namespace std;

// Driver Code
int main()
{
    // Initializing string`
    string str = "geeksforgeeks";

    // Declaring iterator
    std::string::iterator it;

    // Declaring reverse iterator
    std::string::reverse_iterator it1;

    // Displaying string
    cout << "The string using forward iterators is : ";
```

Start Your Coding Journey Now!

```cpp
    return 0;
}
```

Output

```
The string using forward iterators is : geeksforgeeks
The reverse string using reverse iterators is : skeegrofskeeg
```

4) Manipulating Functions:

Function	Definition
copy("char array", len, pos)	This function copies the substring in the target character array mentioned in its arguments. It takes 3 arguments, target char array, length to be copied, and starting position in the string to start copying.
swap()	This function swaps one string with other.

Let's look at the following example:

CPP

```cpp
// C++ Program to demonstrate the working of
// copy() and swap()
#include <iostream>
#include <string> // for string class
using namespace std;

// Driver Code
int main()
{
    // Initializing 1st string
```

Start Your Coding Journey Now!

```cpp
    // Displaying char array
    cout << "The new copied character array is : ";
    cout << ch << endl;

    // Displaying strings before swapping
    cout << "The 1st string before swapping is : ";
    cout << str1 << endl;
    cout << "The 2nd string before swapping is : ";
    cout << str2 << endl;

    // using swap() to swap string content
    str1.swap(str2);

    // Displaying strings after swapping
    cout << "The 1st string after swapping is : ";
    cout << str1 << endl;
    cout << "The 2nd string after swapping is : ";
    cout << str2 << endl;

    return 0;
```

Data Structures   Algorithms   Interview Preparation   Topic-wise Practice   C++   Java   Pytho

```
The new copied character array is : geeksforgeeks
The 1st string before swapping is : geeksforgeeks is for geeks
The 2nd string before swapping is : geeksforgeeks rocks
The 1st string after swapping is : geeksforgeeks rocks
The 2nd string after swapping is : geeksforgeeks is for geeks
```

Must Read:

- C++ String Class and its Applications
- C++ String Class and its Applications | Set 2

Start Your Coding Journey Now!

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

Like   253

Previous   Next

Start Your Coding Journey Now!

02 How to convert a class to another class type in C++?
05, Mar 19

13, Nov 10

03 Difference between Base class and Derived class in C++
07, May 20

07 What happens when more restrictive access is given to a derived class method in C++?
20, Sep 12

04 Base class pointer pointing to derived class object
18, Feb 21

08 How to make a C++ class whose objects can only be dynamically allocated?
23, Sep 13

Article Contributed By :

GeeksforGeeks

Vote for difficulty

Current difficulty : Easy

Easy	Normal	Medium	Hard	Expert

Improved By :   Fenrear,   keshariashwani678,   shettykaran21,   thepranaygupta,   as5853535,   johnakup02,   anshikajain26

Article Tags :   cpp-string,   cpp-strings-library,   STL,   C Language,   C++

Start Your Coding Journey Now!

Writing code in comment? Please use **ide.geeksforgeeks.org**, generate link and share the link here.

Load Comments

5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org

Company

About Us

Careers

Privacy Policy

Contact Us

Copyright Policy

Learn

Algorithms

Data Structures

Languages

CS Subjects

Video Tutorials

Web Development

Web Tutorials

HTML

CSS

JavaScript

Bootstrap

Contribute

Write an Article

Write Interview Experience

Internships

Videos