C++ String

Topics : CPP
Written on April 15, 2023

In C++, a string is a sequence of characters stored in memory as an array of characters. The string class in C++ provides a convenient way to work with strings, and is part of the standard library.

To use the string class, you first need to include the <string> header file:

```
#include <string>
```

You can then declare a string variable like this:

```
 std::string myString;
```

To initialize the string with a value, you can use a string literal:

```
std::string myString = "Hello, world!";
```

You can also read a string from input using the std::getline() function:

```
std::string myString; std::getline(std::cin, myString);
```

Once you have a string, you can perform various operations on it, such as finding its length:

```
std::string myString = "Hello, world!";
std::cout << "Length of myString: " << myString.length() << std::endl;
```

You can also concatenate strings using the + operator:

```
std::string str1 = "Hello, ";
std::string str2 = "world!";
std::string str3 = str1 + str2;
std::cout << str3 << std::endl; // Output: Hello, world!
```

You can access individual characters in a string using the [] operator:

```
std::string myString = "Hello, world!";
```

```
std::cout << myString[0] << std::endl; // Output: H
```

You can also use the string::find() function to search for a substring within a string:

```
std::string myString = "Hello, world!";
std::string subString = "world";
if (myString.find(subString) != std::string::npos) {
    std::cout << "Found substring!" << std::endl;
} else {
    std::cout << "Substring not found." << std::endl;
}
```

Finally, when you are done with a string, you don't need to manually deallocate its memory; the string class will automatically handle memory management for you.