String I/O

string is an object of **std::string** class that represents sequence of characters. We can perform many operations on strings such as concatenation, comparison, conversion etc.

This string is actually a one-dimensional array of characters which is terminated by a **null** character '\0'. Thus a null-terminated string contains the characters that comprise the string followed by a **null**.

The following declaration and initialization create a string consisting of the word "Hello". To hold the null character at the end of the array, the size of the character array containing the string is one more than the number of characters in the word "Hello."

```
char greeting[6] = {'H', 'e', 'l', 'l', 'o', '\0'};
```

If you follow the rule of array initialization, then you can write the above statement as follows −

```
char greeting[] = "Hello";
```

```cpp
#include <iostream>
#include <cstring>

using namespace std;

int main () {

   char str1[10] = "Hello";
   char str2[10] = "World";
   char str3[10];
   int  len ;

   // copy str1 into str3
   strcpy( str3, str1);
   cout << "strcpy( str3, str1) : " << str3 << endl;

   // concatenates str1 and str2
   strcat( str1, str2);
   cout << "strcat( str1, str2): " << str1 << endl;

   // total lenghth of str1 after concatenation
   len = strlen(str1);
   cout << "strlen(str1) : " << len << endl;

   return 0;
}
```

When the above code is compiled and executed, it produces result something as follows −

strcpy(str3, str1) : Hello
strcat(str1, str2): HelloWorld
strlen(str1) : 10

Function	Description
int compare(const string& str)	It is used to compare two string objects.
int length()	It is used to find the length of the string.
void swap(string& str)	It is used to swap the values of two string objects.
string substr(int pos,int n)	It creates a new string object of n characters.
int size()	It returns the length of the string in terms of bytes.