C++ Strings

Strings are used for storing text.

A string variable contains a collection of characters surrounded by double quotes:

```
#include <iostream>
#include <string>
using namespace std;
int main()
 {
  string greeting = "Hello";
  cout << greeting;
  return 0;
}
```

String Concatenation

The + operator can be used between strings to add them together to make a new string. This is called concatenation:

Example

```
#include <iostream>
#include <string>
using namespace std;
int main ()
 {
  string firstName = "Disha  ";
  string lastName = "Computer";
  string fullName = firstName + lastName;
  cout << fullName;
}
```

Example

```
#include <iostream>
#include <string>
using namespace std;
int main ()
{
  string firstName = "Computer";
  string lastName = "Disha";
  string fullName = firstName + " " + lastName;
  cout << fullName;
  return 0;
}
```

Append

A string in C++ is actually an object, which contain functions that can perform certain operations on strings. For example, you can also concatenate strings with the append() function:

Example

```
#include <iostream>
#include <string>
using namespace std;
int main ()
{
  string firstName = "Disha ";
  string lastName = "Computer";
  string fullName = firstName.append(lastName);
  cout << fullName;
```

```
}
```

Adding Numbers and Strings

- C++ uses the + operator for both addition and concatenation.
- Numbers are added. Strings are concatenated.
- If you add two numbers, the result will be a number:

Example

```
#include <iostream>

using namespace std;

int main ()

{

  int x = 10;

  int y = 20;

  int z = x + y;

  cout << z;

}
```

If you add two strings, the result will be a string concatenation:

Example

```
#include <iostream>

#include <string>

using namespace std;

int main ()

{

  string x = "10";

  string y = "20";
```

```
  string z = x + y;
  cout << z;
}
```

String Length

To get the length of a string, use the **length()** function:

Example

```
#include <iostream>
#include <string>
using namespace std;
int main()
 {
  string txt = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
  cout << "The length of the txt string is: " << txt.length();
}
```

Tip: You might see some C++ programs that use the **size()** function to get the length of a string. This is just an alias of **length().** It is completely up to you if you want to use **length() or size():**

```
#include <iostream>
#include <string>
using namespace std;


int main() {
  string txt = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
  cout << "The length of the txt string is: " << txt.size();
```

```
}
```

Access Strings

You can access the characters in a string by referring to its index number inside square brackets [].

This example prints the first character in myString:

Example

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
  string myString = "Hello";
  cout << myString[0];
}
```

Change String Characters

To change the value of a specific character in a string, refer to the index number, and use single quotes:

Example

```
#include <iostream>
#include <string>
using namespace std;
int main()
{
  string myString = "Hello";
  myString[0] = 'J';
```

```
  cout << myString;
}
```

Strings - Special Characters

Because strings must be written within quotes, C++ will misunderstand this string, and generate an error:

The solution to avoid this problem, is to use the **backslash escape character**.

The backslash (\) escape character turns special characters into string characters:

Escape character	Result	Description
\'	'	Single quote
\"	"	Double quote
\\	\	Backslash

The sequence \" inserts a double quote in a string:

Example

```
#include <iostream>
using namespace std;
int main()
{
  string txt = "We are the so-called \"Vikings\" from the north.";
```

```
  cout << txt;
}
```

The sequence \' inserts a single quote in a string:

Example

```
#include <iostream>
using namespace std;
int main()
{
  string txt = "It\'s alright.";
  cout << txt;
}
```

The sequence \\ inserts a double backslash in a string:

Example

```
#include <iostream>
using namespace std;
int main()
 {
  string txt = "The character \\ is called backslash.";
  cout << txt;
}
```

The sequence \n a single backslash in a string:

```
#include <iostream>
using namespace std;
```

```
int main()
 {
  string txt = "Hello\nWorld!";
  cout << txt;
}
```

The sequence \t a single backslash in a string:

```
#include <iostream>
using namespace std;
int main()
 {
  string txt = "Hello\tWorld!";
  cout << txt;
}
```

C++ String compare()

```
#include<iostream>
using namespace std;
int main()
{
string str1="Hello";
string str2="Hi";
int k= str1.compare(str2);
if(k==0)
{
```

```
cout<<"Both the strings are equal";
}
else
{
cout<<"Both the strings are unequal";
}
}
```

C++ String replace()

This function replaces the portion of string that begins at character position pos and

spans len characters.

Syntax

Consider two strings str1 and str2. Syntax would be:

str1.replace(pos,len,str2);

```
#include<iostream>
using namespace std;
int main()
{
string str1 = "This is C language";
string str2 = "C++";
cout << "Before replacement, string is :"<<str1<<endl;
str1.replace(8,1,str2);
cout << "After replacement, string is :"<<str1<<endl;
```

```
}
```

C++ String resize()

This function is used to resize the string to the length of k characters.

Syntax

Consider a string object str. To resize the string object, syntax would be :

```
str.resize(k,c);
```

Example

```
#include<iostream>
using namespace std;
int main()
{
string str= "C++ programming";
cout<<"String is :"<<str<<endl;
str.resize(4);
cout<<"After resizing, string is "<<str;
return 0;
}
```