

## The `string` Class

A string, to put it simply, is text. For example, “Welcome to my program” is a string. Naturally, it is useful to be able to manipulate strings as variables for storing names and the like. Text can be considered as an array of characters, but is difficult to manipulate as such. The standard library for C++ includes a string class, which permits manipulation of strings similar to other variables. A `string` can be declared just like any other variable. Take a look at this example:

```
#include <iostream>

using namespace std;

int main()
{
    string message = "Your name is "; // Set on initialization
    string name;
    cout << "Enter your name: ";
    cin >> name;                      // Read in via cin
    cout << message << name;          // Displayed with cout
    return 0;
}
```

Notice that quotation marks are used to surround a string when it appears in code. There are a number of operators available for manipulating strings, and they are summarized in the following table.

### Select `string` Class Members

Member	Use
=	Assigns one string to another string: <code>S = "Hello"; t = s;</code>
== > >= != < <=	Compare two strings alphabetically: <code>if (t &gt;= s) cout &lt;&lt; "s comes first!";</code> <code>if (s == t) cout &lt;&lt; "Same string!";</code>
[n ]	Like vector notations; Returns the (n+1)th character of the string. <code>cout &lt;&lt; s[0] &lt;&lt; s[1];</code>
+= (str) += (char)	Appends string <code>str</code> Appends character <code>char</code>
+	Concatenates (puts together) two strings (or a char and an string): <code>t = "Hello " + name; s = 'a' + name; w = name + 's';</code>

<code>size()</code>	Returns the size (number of characters) of the string: <code>int len = s.size();</code>
<code>resize(num)</code>	Changes the size(number of characters) of the string: <code>s.resize(s.size()+5);</code>
<code>substr(pos, len)</code>	Returns len characters of the string beginning at index pos. <code>string s = t.substr(2, 3);</code>
<code>find(str)</code> <code>find(ch)</code>	Returns the index of the first occurrence of string <code>str</code> in the string, -1 if nothing is found. Can also be used to find first occurrence of a character <code>ch</code> in the string.
<code>find_last_of(str)</code> <code>find_last_of(ch)</code>	Returns the index of the first character within the current string that matches any character in <code>str</code> , doing a reverse search, -1 if nothing is found. Can also be used to find first occurrence of a character <code>ch</code> in the string.
<code>find_first_of(str)</code> <code>find_first_of(ch)</code>	Returns the index of the first character within the current string that matches any character in <code>str</code> , doing a forward search, -1 if nothing is found. Can also be used to find first occurrence of a character <code>ch</code> in the string.
<code>getline(infile, s)</code>	Reads one line from <code>infile</code> into string <code>s</code>
<code>atoi(s)</code>	stdlib function. Returns <code>char * string s</code> converted to an integer. To use with string, use <code>c_str()</code> function. For example: <code>string nstr = "12345";</code> <code>int num = atoi(nstr.c_str());</code>
<code>atof(s)</code>	stdlib function. Returns <code>char * string s</code> converted to a float. To use with string, use <code>c_str()</code> function. For example: <code>string nstr = "12345.234";</code> <code>float num = atof(nstr.c_str());</code>

For additional member functions of the `string` class, please consult additional on-line sources, or posted C++ textbooks.