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`: `S = "Hello";  t = s;`

== > >=
!= < <=
	Compare two `strings` alphabetically:
	`if (t > s)  cout << "s comes first!";`
	`if (s == t)  cout << "Same string!";`

[n]	Like `vector` notations; Returns the (n+1)th character of the `string`. cout << s[0] << s[1];
+= (str) += (char)	Appends `string str` Appends character `char`
+	Concatenates (puts together) two `strings` (or a `char` and an `string`): `t = "Hello " + name; s = 'a' + name; w = name + 's';`

`size()`	Returns the size (number of characters) of the `string`: `int len = s.size();`
`resize(num)`	Changes the size(number of characters) of the `string`: `s.resize(s.size()+5);`
`substr(pos, len)`	Returns `len` characters of the `string` beginning at index `pos`. `string s = t.substr(2, 3);`
`find(str)` `find(ch)`	Returns the index of the first occurrence of `string str` in the `string`, -1 if nothing is found. Can also be used to find first occurrence of a character `ch` in the `string`.
`find_last_of(str)` `find_last_of(ch)`	Returns the index of the last character within the current string that matches any character in *str*, doing a reverse search, -1 if nothing is found. Can also be used to find last occurrence of a character `ch` in the `string`.
`find_first_of(str)` `find_first_of(ch)`	Returns the index of the first character within the current string that matches any character in *str*, doing a forward search, -1 if nothing is found. Can also be used to find first occurrence of a character `ch` in the `string`.
`getline(infile, s)`	Reads one line from `infile` into `string s`

`atoi(s)`	`stdlib` function. Returns `char * string s` converted to an `integer`. To use with `string`, use `c_str()` function. For example:
	`string nstr = "12345";`
	`int num = atoi(nstr.c_str());`

`atof(s)`	`stdlib` function. Returns `char * string s` converted to an `float`. To use with `string`, use `c_str()` function. For example:
	`string nstr = "12345.234";`
	`float num = atoi(nstr.c_str());`

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

Examples:

To capture a string s, which includes spaces, you can use getline with cin: getline(cin,s)

To find the first # in a string s, you can use find: `int pos = s.find('#')`

To "grab" the first 5 characters in string s, you can use substr: `string new = s.substr(0, 5)`