COM S 213 – Spring 2003

ASSIGNMENT #4: **Dynamic Arrays in the Menu Class**

DATE GIVEN: **2/13/03**

DATE DUE: **2/20/03**

PURPOSE:
To apply what we've learned about dynamic arrays and strings to our Menu class.

ASSIGNMENT:
In assignment #3 we hard coded the number of menu items in our solution by deciding on a number of items in the menu file AND by hard coding one member variable for each choice. Obviously, this is not very scalable!

We can improve on our menu class by using a dynamic array of strings to store the menu choices instead of relying on knowing the number of items in the menu at compile time!

Part One
This part of your assignment is to remove the member variables you used to store the text of the menu choices in assignment #3 and replace them with a dynamic array of strings. So, you'll need a member variable to store the dynamic array, and you should keep a member variable around that stores the size of the dynamic array. You'll need to adjust all code that previously relied on the "hard coded" member variables to rely on the dynamic array.

Part Two
You will need to adjust the code that reads the menu data from a file to something more generic. Since our data type now can deal with (in theory) any size menu data file, you should provide a separate member function to read in the data. The new format of the menu data file is as follows:

```
4
New File…
Open File…
Close Window
Print
```

What we see here is that the file begins with an integer that details how many entries are stored in the file. There could be any positive integer in this first position in the file. You can be assured that whatever the integer is, it will be followed by that number of strings which represent menu choices.

The member function you create for reading in the menu choices will be used in two places. First, any Menu object can have its menu choices changed at any time during its lifetime by having this public member function called. As such, it will be a public function. You should then have two constructors: a simple constructor which just creates a Menu with no entries, and a constructor which takes a filename as an argument and calls the public function which reads in the menu data from the specified file.

Part Three

You'll notice that, in my example file above, the menu choices may be more than one word (hence, they may have whitespace). From what we learned in lecture you will need to use the ifstream::getline() function to make sure things happen properly. Consult the lecture notes from Lecture #8 if you need to review.

I will provide two menu data files to test with. Your program should load the first menu data file when your menu object is declared in main(), and you should also demonstrate how the public member function can be called directly to change the menu choices after the object has been created. This should be doable with the following code:

```
// Code snippet
void main()
{
     Menu myMenu("menu1.dat");

     cout << "About to test first menu…" << endl;
     myMenu.Run();

     cout << "Testing second menu… " << endl;
     myMenu.readMenuData("menu2.dat");

     myMenu.Run();
     cout << "Good bye!" << endl;
}
```

OK, now the assignments are beginning to get harder. Please start this one as soon as you can so you have adequate time to ask questions. I will post the two data files on the web site in the assignments section.