Ryan Hoffman
CS 202 Project 7 Documentation
Lab Section: 1107
4/1/2019

Compile Command: ./proj7.cpp MyString.cpp -o proj7

The purpose of this program was to practice allocating and deallocating memory for an array of chars that was supposed to simulate what happens when we create a string object. We also used pointers and classes to create this project. The first part of the MyString class was the constructors that allocated memory for a string based on the C-String that was passed into it. The next part was the functions that were used to get the size, length, and C-String equivalent of the string stored in the object. The last part was the operator overload functions that performed operator actions on the strings like =, + (concatenate), etc.

When creating this program I tried to think what would need to happen to create a string and after finding out a string is pretty much just an array of chars it made everything a lot easier. To know how much memory I had to allocate, I used the strlen() function and then added 1 for the null terminator. The string.h functions were very helpful for performing actions like string comparing and string concatenation. For memory allocation I tried to use try, catch, and throw to catch any errors and revert the changes to the memory. I found out that the deconstructors are for deallocating memory at the end of the program. The whole point of destructors was confusing to me earlier because we didn't put anything inside them.

The only problem I ran into was that I was getting a memory error because I didn't initialize anything in my default constructor. I didn't know we had to initialize the char pointer to null before allocating any new memory into it. The last problem I had was that instead of just deleting m_buffer as a whole, I kept trying to deallocate the individual array elements of m_buffer which kept giving me an error. It was a lot easier than I thought to delete the memory and allocate new memory if need be. I was confused on how I would print to an output file if an ofstream object was passed into the insertion operator overload but I think I made it correctly.

If I had more time I would make the extra bit memory allocation for the null character get allocated inside the allocation function. Instead, I just incremented the length by 1 when specifying the size of the data I wanted to allocate in the other functions. It was a little unclear why there were 2 bracket operator overload functions where the only difference was that one of them was completely const qualified. I'm not sure if the non-const version of the function was needed because they both do the same thing. I had to add a couple things to the proj7.cpp file to print out the strings to show that the manipulation or function called actually worked. Before allocating new memory to m_buffer I would delete the current memory. If I had more time I would like to find out how to reallocate memory to expand or limit the amount of memory that was allocated for a string.