Jason Goldfine-Middleton
932675265

CS 162 Lab 6 Reflection

Testing this code was relatively simple since the nodes for both the Stack and Queue contained int items. I used print statements in my add() and remove() functions for the two classes that displayed the values of the int being added or removed and also what the current top, back, or front items were. I also had to deal with the case of node pointers being null, so I used if/else conditions to print information in these cases. While I didn't run into any issues with the Stack class, I did have issues with the Queue class. It turned out that I was receiving segmentation faults as a result of trying to access the "next" pointer of a front node pointer that was set to NULL in certain circumstances. I resolved this by using an if statement to check whether the pointer was null before trying to access the "next" pointer. Another issue that I resolved via these print statements was that I had forgotten to make the old back node point to the new back node in the Queue::add() function. This made removing any node added after the first possible.

To wrap up my testing, I created a driver main() function with an array of ints and added and removed them to Stack and Queue objects. I also made sure to verify that, when trying to remove an item from an empty Stack or Queue, I was returned a nonsense value. Everything ended up fine thanks to careful testing and planning. I look forward to seeing how this will be used in Assignment 4.