Shawn S Hillyer

CS 352

10/30/2015

Lab 6 – Reflections - LinkedList implementations

I decided to follow the book's example of making both of my linked list ADT's encapsulated in a class.

I used the same framework I've been using all class to set up a simple set of functions that I could use to create and test a couple of lists.

I also decided to just code my classes using the template format because I'm familiar with it. I did this because I wanted to make sure it would work with other types without much/any work. I suspect that the assignment4 coming up will have us use the implementations we came up with to make a program based on the modules.

The tests I came up with were simple. For each class, I set up an automated test where I instantiated the class then simply added 10 values to the stack, then removed them.

I knew from my previous studies how to implement these on a general level, so after mapping it out on paper, I coded my stack list first. Adding and removing values worked fine in my loop, so I moved on to the queue. Once that was done, I decided to get more rigorous with my testing.

Specifically, I set up a simple input loop that would take user input and allow the user to type in values until they enter -1 and it would add it to the stack or queue as appropriate. Once done, they could press any key to remove values.

This worked very well, but I realized I could add exception handling to my class to make sure I didn't allow the removal of a node that didn't exist to both classes. So I threw in a try catch and put a throw statement inside my node for the case of removing a node.

A stack basically takes an input items and returns them in reverse order. So you should be able to add 1, 2, 3, 4, 5 and pop off 5, 4, 3, 2, 1 in that order.

Likewise, with a queue, it should come off in the order they were added. So 1, 2, 3, 4, 5, would remove and pop off 1, 2, 3, 4, 5 from the front of the list.

The only thing I forgot to test at first was an add, pop, add, pop type of sequence where some values were added, then remove some, then add some more, and remove some more. So I added that to the end of the existing test methods real quick and everything is working as expected.