Lab 03 Double Linked List

Bradley Grose

# Problem

In this lab, we must make a double linked list, meaning that the list can go in both directions. It most use the driver given. The linked list must be able to go forwards, backwards, reset its position, set new positions, delete positions, and print out conditions, as well as some true false values for the driver to display.

# Solution

First, I created my list node class which has a next link and a previous link node. Then I created the 3 needed pointers of the listnode, the data, the next link, and the previous link. Then I defined my two spot holders of current and the head, or the start point. The constructor for the list created the head and the current at the same spot as null. The goToNext function first checks to make sure there is data in the spots and if so it moved to the next link in the list. The goToPrev does the same, just in reverse. Then the moveToIterate is a Boolean that says if the two spots have data. The setCurrent function takes in the given data and then saves that current node as that data. The getCurrent function returns the data in the current spot, or null if applicable. The Insert function takes in data, then checks to see if the head is null, if it is it makes that data the head. If not, it runs through the whole list until it finds a null pointer and adds the data there. The insertAfterCurrent function is similar, however instead of searching for the spot, it breaks the link with the next node, links the previous one to the new node and then repairs the link. The deleteCurrent does the reverse of that, breaking the links and then connecting the previous and the next link together, removing the reference of the current node. Then the print function runs a while loop that goes through the whole list until it is null. Finally, the contains function is a Boolean function that takes in data, and checks the link list to see if that data exists anywhere in it.

# Implementation Problems Encountered

No problems were encountered in this lab.

# Lab Report Questions

1. The JVM will delete the information when the reference is lost or unreachable using its memory clearance that removes useless data or information
2. An advantage is you are able to move in both directions compared to a single linked list. Another advantage is the delete function will still be able to let u move backwards if needed. It also is better as you can insert a node at the middle of the list. The only disadvantage is it is more coding as another pointer is needed and needs to be maintained.