Author: Marcus Brooks

Assignment 3 Lessons Learned

During this assignment I learned how to implement an iterator. I hadn’t used iterators before this assignment and I’ve learned how they work and how to build one. I also hadn’t worked with any circular data structure before this assignment. So implementing all of the methods of these linked lists and the iterator was an interesting challenge with the linked list being circular. It was a little difficult at first to come up with ways to tell loops when to stop, since there would be no node who’s next or previous field would equal null in a circular linked list (this could be remedied perhaps by having a dummy node and instead checking each node’s data value for null). However, in my remove() method for example, I just opted to have one variable reference the current node, and one variable reference the nextNode, and I would just stop the loop once the nextNode reference was equal to the head of the linked list or the data field of the next node was equal to the data to be removed.

This assignment wasn’t terribly difficult, but it did require a lot of careful thinking. Working with and manipulating nodes in linked lists so much can be kind of taxing to think about to make sure you get everything right. I did eventually arrive at a working solution and I’m happy with my implementation. This would have been harder to build if it was a singly-linked list, like the document says. But after checking with my professor, we’re actually meant to build a doubly-linked list which is much easier. I’m not sure how you would implement the previous() method of the iterator in a singly-linked list.

I’m not sure if this assignment will change any of my future approaches to projects. Perhaps I’ll utilize an iterator if I think it would be easier than implementing my own iteration solutions. I can’t immediately think of anywhere where I’d want to use a circular data structure over a non-circular one. However, if I think of a usage in the future, I’ll understand how to create and use one!