Nickolas Komarnitsky

2/23/2017

Assignment06

**Analysis of Linked and Array lists**

**Graphs**

These graphs show that it is constant for adding to the beginning and the end of a linked and array list. There are a few outliers in the data, most likely because of something in the computer taking too long. The linked list took far less time to add an item then the array list though. The array list has to do many more operations than the linked list does to add anything to the beginning or middle. Adding to the last for each takes very little time however. Removing from the beginning of the list took very little time for the linked list. The array list took a little more time, as it has to shift the entire array over.

**Classes**

Linked\_List and Array\_List both implement the list interface. They don’t interact otherwise. The linked list class uses a private class Node to keep track of the data. Multiple Nodes back the list. The Node class is necessary because you need an object to keep track of the data and the following Node. If you make a new linked list every time it will have the methods in linked list each time. The linked list is a list of Nodes. The linked list class modifies the Nodes as needed. It is a static inner class because it is declared inside the Linked\_List\_2420.java file. It is static because it always stays the same.

**Recursion**

The recursion in this project is doing the same thing as the iterative, simply in a different way. The recursive is simply another way of writing a while loop. It checks for certain conditions and then calls itself on a different number, doing the same checks. A for loop does the same thing, but it will stop if the checks return false. With recursion, it is possible to get an infinite loop, while a for loop is a little harder to.

**Reflection**

The most time consuming part of this project was this, the Analysis. Running the timing program took a while. I think I managed my time well enough. Regression testing is making sure that new code will work with old code. This is useful for making sure that a patch will work properly. Writing the tests first helped me get an idea of what I need to do to make it work correctly and to compensate for edge cases. A queue is created by adding to the end of the list and removing from the front, sorting the items by priority when they are added. The linked list simply puts the item into the list. It could be modified to check the entire list when an item is added and put it where it needs to be.

**Conclusion**

This project was an easier one, the coding was fairly simple. The thought process of completing this project was easy. The hardest part was making sure that it followed the assignment guidelines. The graphs took a while, but they came out well.