Koby Miller

Project 1

Determinant of a Matrix using a Recursive algorithm.

I coded this project using a linked list. I created a Node class that was used in each link of the linked list. Each Node had an int for its data and 2 other ints to keep track of that nodes row and column in the matrix. Lastly, each node would also have a next and prev nodes. When a piece of data is added a new node is made and put in the list. If the data is 0, there is no node (so it won’t take up any memory).

Furthermore, I didn’t know how we were receiving the matrix exactly, so I first made my code as if I already had a matrix (it is weird to me not to be writing the main class). But what most of my methods do is traverse the list and look for a specific piece of information in each node. Once it finds this information, it completes that method (whether it be adding or removing or getting. Each method is kind of self-explainable). If the method needs to run through the entire matrix (0’s and all) it doesn’t just traverse through the list. It tries to find a node with each specific coordinate and if it cannot find that coordinate than the data must be 0.

My determinant method is very similar to the equation we were given. The only difference is that instead of iterating the rows, I iterated the columns. I did this because it visually helps me and because that is how I know how to do it on paper so if I had any problems I know I could physically work out the determinant.

I chose this method because honestly it was the first thing that came to mind. I knew I needed to find a way to not allocate memory for 0’s. So, I thought of how I could create a coordinate system and if the specific coordinate was empty than it would really be 0. With one list I thought I would be able to reach any information needed easily.

Computational complexity is in the comments on the code. It will be commented on the same line as the name of the method.