Github Repo:

https://github.com/LuckyOwl69/Tic-Tac-Toe

# Overview

I found this exercise to be fairly difficult in the beginning. I attempted to use pointers to structure my assignment, similar to the previous linked list exercise. The most difficulty I had was in the logic behind my CPU function. The rest of the exercise was fairly straightforward and in the end I’m mostly happy with how it turned out.

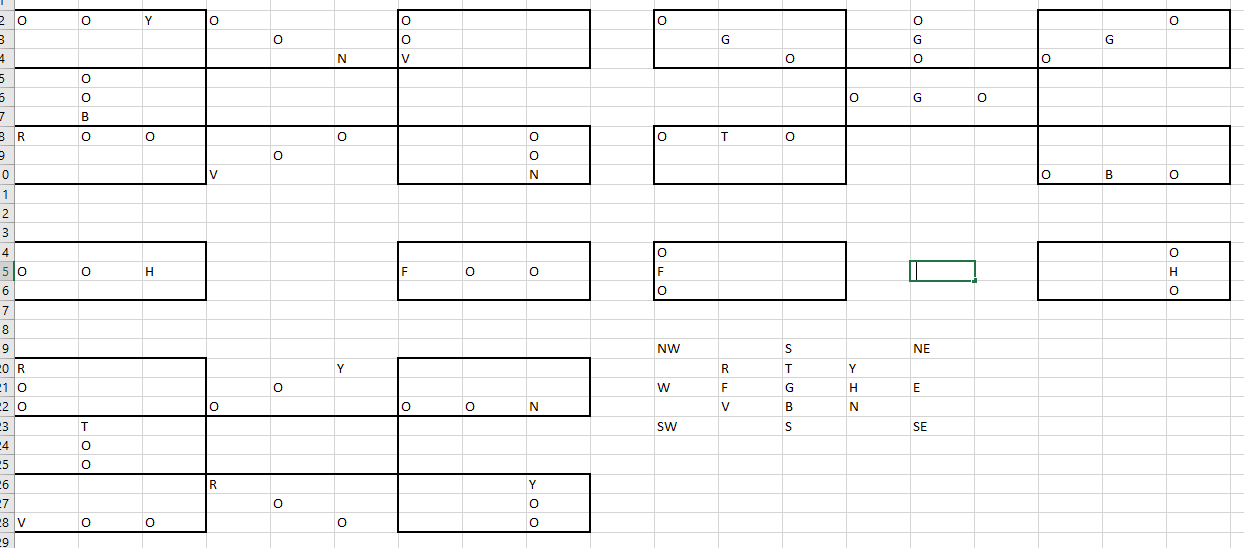
# What went right

Structuring the program went fairly smoothly, this is because it was based heavily on the previous linked list assignment. In fact most of the assignment was fairly simple after I had the structure implemented, but what was frustrating was how long it took to write everything up. Overall the entire exercise progressed mostly smoothly.

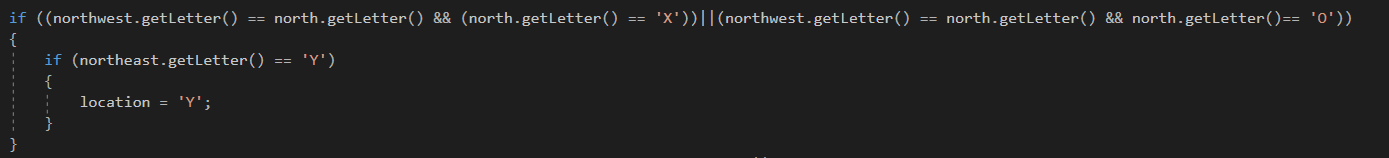
# What went wrong

The way I implemented pointers in this exercise could have done with a bit of a rework. They weren’t implemented with as much usefulness as I think I could have since each position in the map doesn’t point to each other position. If they did then certain options could have been implemented better, such as functions for checking for wins, losses, draws, etc.

I found implementing the CPU difficult because I couldn’t come up with a solution for how to refer to one player (or both) being CPU. Eventually I make it so that if the user types in “CPU” as their name then it will implement the CPU function. In order to write up the logic I drew all the options in a spreadsheet:



I am not sure if this is all the possible combinations however this is what I came up with. I then wrote a simple if function for when either “O” or “X” is in two positions in a row the next “location” the CPU will choose is the third option in the row:



The CPU also checks if the space is occupied before it does a move. However the logic I used doesn’t work correctly and the CPU still choses spots that’s are occupied.

# What I’m unsure about

So the CPU function I wrote accounts for every possible combination I could come up with however what I’m unsure about is how to write a function that doesn’t need to implicitly state all the options. For instance, maybe the function has pointers to every location on the board, and pointers from each location to every other location. I’m unsure if I could create a function that compares the data that two pointers that are adjacent to each other is, and then using this react appropriately.

Furthermore the logic of the CPU function that I wrote does not function perfectly. The CPU still choses spaces that are taken up already. After trying to fix it for a few hours I have determined that the logic behind my entire CPU function if flawed and needs a complete rewrite.

# Conclusion

This was a good exercise to bring together a few different parts of this subject. I found it challenging and in the end didn’t complete every part of it, but what I did complete I’m actually pretty happy with. The one thing I didn’t prepare myself for was how long the entire process would take. Given another day of work I think I would have figured out how to fix my problems with the CPU.