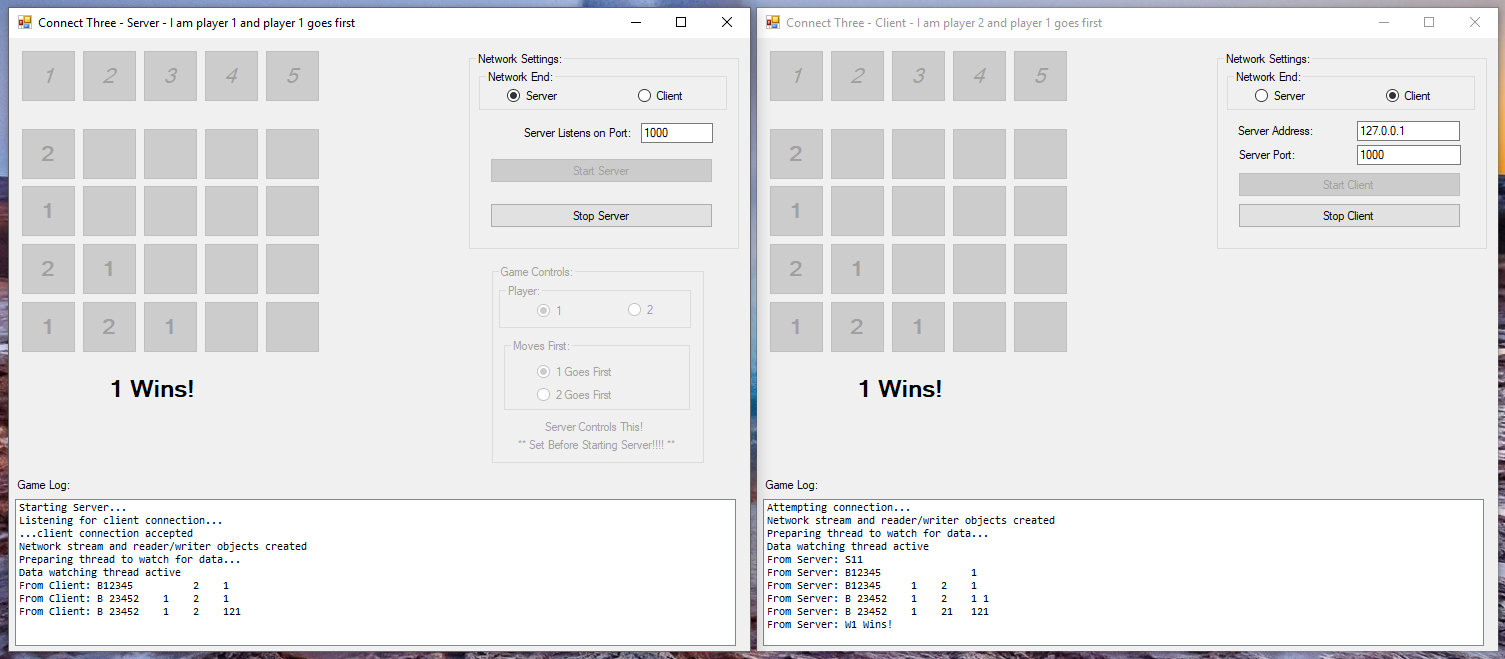
***CIS 311 – Assignment 11***

Network “Connect Three!” It’s just like Connect Four but played on a 4 x 5 grid and you only have to get three of your markers in a row, column or left or right slanting diagonal. The application will allow the server to set which side it plays as (1 or 2) as well as who gets to go first (1 or 2 again) before the server starts up. Once the server starts and the client connects to it, it is up to your software to make sure it fulfills the following requirements:

1. The software should control who gets to make a move in a turn-based manner between the two end points.
2. The software should not allow any illegal moves; e.g. it should not allow you to select a column to drop a marker where there are no more spaces left. Additionally, the game should not allow you to play when it is not your turn.
3. The game also needs to stop game play once a player wins or a tie has been reached and report that status to the other end point.

Here’s a screenshot of the server and client after a game has been played and won:



You really need to develop a simple protocol for the communication of the game information that gets exchanged back and forth between the two end points. I created a game log that shows the status of the client/server set up along with the messages sent back and forth.

My protocol was very simple with three message types in it:

S – the initial set up from the server to the client showing which player is which and who will be going first.

B – lists the columns that are still valid to allow drops on followed by the whole board setup mapped from 2-D space into a 1-D string.

W- sent whenever one player has either won or there is a tie.

Strategies to complete this:

1. Get the network piece functioning first. Until you can spin up a server with a listening thread, a client with a listening thread and are able to send a simple string between the two bidirectionally, you are dead in the water. This is a NETWORKED version after all.
2. Put some thought into how you are going to manage the board (it is the in-game representation of where the 1s and 2s live). You also need to think about how you will translate the board into the 2-D matrix that makes up the markers that have been placed. You also must track which columns still have open spots that drops can be made on. Write the code to allow a marker to be played by clicking on an available column button, the board gets updated, send the board across the network and then update the display on the other end point.
3. Consider the protocol that you are going to develop to send the various types of messages between the two end points.
4. Once you can send and receive the board, now you can worry about things like the initial set up, turn based play and “win” status between the two end points.

Obviously there are a lot of ways to attack this problem and as such I will be of limited help to you with regard to debugging and troubleshooting since there are so many ways that you can be trying to do things. You are going to need to become reliant on yourself for most of the problem solving when you run into issues on this assignment. This is partially to get you ready for next year because when I give you work in the capstone classes, you are going to have to be accountable for finding and fixing problems. As seniors about ready to enter industry, should you be at all surprised by this? I will be more than happy to talk strategy about this game program, but don’t expect me to look at program code in any amount of detail. You know everything there is to know about Connect Four (in our case Three) and I’ve supplied to you the necessary background on creating a client and server – you just need to think, write a little code and test that code. Repeat over and over until you have built this program. It is not at all an impossible assignment!

So, I don’t care if you write this as a single application (because so much of the logic is identical between the client and server) or if you want to write it as two separate applications (one for the client and one for the server).

Upload a copy of your program(s) to Canvas, print out your cover sheet/screenshots and turn those in during class on the due date.