Cosc 5/4735 Program 3
Due: March 25, 2020 50 points

1. Write the following program phone application. Make sure to comment your code and that your name is in a comment line at the beginning of code.

You can use NearBy or Bluetooth to make the connection between the two phones. Note for the rest of the assignment it will say NearBy, but you may be using Bluetooth. I will suggest that NearBy maybe easier, but the choice is yours

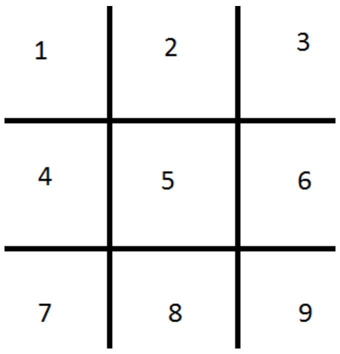
For Nearby the ServiceID will be "edu.cs4730.nearbyconnectiondemo" and use the Strategy P2P_STAR. As found in the nearbyConnectionDemo. This way everyone using nearby can play against each other's code.

For Bluetooth the UUID will be fa87c0d0-afac-11de-8a39-0800200c9a66 as found in the Bluetooth demo code. Again, so any Bluetooth players can play against each other code.

Tic-tac-toe program revisited again:
Using the Tic-Tac-Toe code from previous class (with any corrections needed), add in NearBy code to it. If you don't have the code, then write a new version. You should be able to play again any other student's and instructor's version. For grading purposes, your code will be played against the instructor/grader's code.

Protocol:
You program will need to connect up to another phone using NearBy. Your game will need to handle both the client and server side of the game. As reference for the rest of this doc, Server is the phone that accepts the connection, and client is other phone. The user only has to enter the number (touch, or click, or whatever) for the position to place an X or O. The application will then determine winner/tie conditions.

The board will look like this and the numbers refer to the position. So position 5 is the center square:



After the server and client connect, the first thing they need to do is agree on who is the X player (ie who goes first). The User on the Server always gets to choose.

The server app will then send one of two messages

player X	Meaning the server is playing X (going first)
player O	Meaning the server is playing O (going second)

The client application then sends back either (not user input)

agree	Meaning the client agrees
disagree	Meaning the client does not agree (a server error in the message!).

Example sequence to determine X and O player

Protocol messages (send/received)	Description
player X	Server has decided to be X
agree	Client agrees
	If disagree, then there is a program error. Game ends.
player O	Server has decided to be player O
agree	client is player X

Playing the game.   Messages are passed back and forth between the two.
Player X would send the position that X is going to.
Example player X picks 5. The message 5 is sent. The other side will now agree or disagree. Then a message is sent either “nowinner”, “winner”, “tie”, the otherwise will now agree or disagree. If either disagrees, there is likely a coding problem or one player is cheating and the game ends. Next Player O picks the position on the board, and then the application(s) do the rest. This sequence continues until winner or tie, and then the game is over.
Example protocol sequence of the game

Protocol messages (send/received)	Description
5	X chooses position 5. User specifies there want to place an x at 5.
agree	O agrees application agrees (no user input)
nowinner	X says no winner app sends nowinner (no user input)
agree	O agrees (same application agrees, no user input)
3	O chooses position 3
agree	X agrees that it is valid.
nowinner	O says no winner (no user input)
agree	X agrees (again no user input needed)
…	7 of moves later
tie	X player claims tie
agree	O player agrees there is a tie, the game is over

Once a game is over, the players need to decide if they want to play again. If either player say no, then the application is done. If both agree to play again, then the Server chooses to be X or O and the game starts again.   The server starts the message with either

playagain	//yes play again
exit	//no we are done

The client can then agree or disagree to playagain. If exit, the client must agree, since the server is closing the connection anyway and if the client disagrees to playagain, the game is over.

Example sequences for Player again:

Both agree to play again	**Server plays, but not client**	**Server is done.**
Sever: playagain	Server: playagain	Server: exit
Client: agree	Client: disagree	Client: agree
Clear board and Choose X player	Program ends	Program ends

TURN IN and GRADING:

Hard copy:

1. A copy page with Name, program #3, cosc 5735 or 4735

Soft copy:

1. Use this link to create your repo https://classroom.github.com/a/y5w4wJnM
2. Upload the project to your repo
3. Create/Edit the readme.md file, add the following:
    - Course number 5735 or 4735
    - Name
    - how to run the program,
    - Bluetooth or NearBy
    - Which phone/emulator to run on including special information like android version (ie v4.4) and screen size.
        - Or if you are using the borrowed, phone, Nexus 5X or Pixel
4. Lastly ensure everything has uploaded to the github website and not just the local repo.

Code will be graded on correctness, comments, and coding style.