# Lab 4 - 2 Player Tic Tac Toe

### Purpose of this Lab:

The purpose of this lab is to get you used to networking and using sockets. (In our case there will be a client and server on our local machine). The main libraries to be used is the sockets library (`socket`), tkinter or pygame.

As for all labs this quarter you will need to ensure your code works using IDLE with the version being used this quarter and refer to my lecture on sockets for a refresher on how to use sockets.

**Introduction to the Lab :**

For this lab, I would like you to use sockets to pass the player moves between the client and server. Assuming our tic tac toe game uses a 3 x 3 board, we will use x/X and o/O for the values that we will be passing back and forth. For those not familiar with Tic Tac Toe, it is a 2 player game where one player's game piece is X and the second player's game piece is O (letter). Each player takes a turn putting their game piece in one of the empty slots in the 3 by 3 board. First place to get 3 of their game pieces that align wins. If the board has no empty slots and nobody wins then the game ends in a tie.

**Program Details:**

You will create 3 new modules with the following details:

- BoardClass (gameboard.py)
  - The board game will consist of the following public interface. (Note this module can be imported and used by both player1 and player2 modules
  - The class should have a minimum of the following (you can add more if you need to):
    - Class Variables:
      - Players user name
      - User name of the last player to have a turn
      - Number of games played
      - Number of wins
      - Number of ties
      - Number of losses
    - Class Methods:
      - updateGamesPlayed()
        - Keeps track how many games have started
      - resetGameBoard()

        - Clear all the moves from game board
    - updateGameBoard()
        - Updates the game board with the player's move
    - isWinner()
        - Checks if the latest move resulted in a win
        - Updates the wins and losses count
    - boardIsFull()
        - Checks if the board is full (I.e. no more moves to make - tie)
        - Updates the ties count
    - printStats()
        - Prints the following each on a new line:
            - Prints the players user name
            - Prints the user name of the last person to make a move
            - prints the number of games
            - Prints the number of wins
            - Prints the number of losses
            - Prints the number of ties
- Player 1 Module(Module Name: player1.py)
    1. Player 1 will ask the user for the host information of player 2:
        1. Prompt the user for the host name/IP address of player 2 they want to play with
        2. Prompt the user for the port to use in order to play with player 2
    2. Using that information they will attempt to connect to player 2
        1. Upon successful connection they will send player 2 the their user name (just alphanumeric user name with no special characters)
        2. If the connection cannot be made then the user will be asked if they want to try again:
            1. If the user enters 'y' then you will request the host information from the user again
            2. If the user enters 'n' then you will end the program
    3. Once player 1 receives player 2's username or if the users decides to play again (see step 4.2 below)
        1. Player 1 will ask the user for their move using the current player display area.
        2. and send it to player 2.
            1. Player 1 will always be x/X
            2. Player 1 will always send the first move to player 2
                1. Each move will correspond to the area on the board they user clicks on.
            3. Once player 1 sends their move they will wait for player 2's move.
            4. Repeat steps 3.1.2 - 3.1.4 until the game is over (A game is over when a winner is found or the board is full)
    4. Once a game as finished (win or tie) the user will indicate if they want to play again using the user interface.
        1. If the user enters 'y' or 'Y' then player 1 will send "Play Again" to player 2

2. If the user enters 'n' or 'N' then player 1 will send "Fun Times" to player 2 and end the program
    1. Once the user is done player they will print all the statistics.

- Player 2 Module(Module Name: player2.py)
    1. Player 2 will accept incoming requests to start  a new game
    2. When a connection request is received and accepted, player 2 will wait for player 1 to send their user name
    3. Once player 2 receives player 1's user name, then player 2 will send "player2" as their user name to player 1 and wait for player 1 to send their move.
        1. Once player 2 receives player 1's move they will ask the user for their move and send it to player 1 using the current player display area.
            1. Each move will correspond to the area on the board they user clicks on.
        2. Once player 2 sends their move they will wait for player 1's move.
        3. Repeat steps 3.1 - 3.2 until the game is over (A game is over when a winner is found or the board is full)
    4. Once a game as finished (win or tie) player 2 will wait for player 1 to indicate if they want to play again using the user interface.
        1. If player 1 wants to play again then player 2 will wait for player 1's first move.
        2. If player 1  does not wants to play again then player 2 will print the statistics
- User Interface:
    - You will create a user interface using either the PyGames library or the Tkinter library.  The user interface should include the following components:
        - A Tic Tac Toe board that allows the user to select their move by clicking on that
        - A dialog that asks the user if they want to play again
        - A display area where you display the reporting statistics when they are done and any other printed information specified in the lab.
        - A text input area for the players name
        - An area where the user name of the current players turn will be displayed.
        - Quit button that allows the user to quit prior to finishing a game
- Other Considerations:
    - All user inputs are case **insensitive**.
    - Hint: Maintain the current state of the board on both client and server side to determine if someone won, lost or the board is full.
- Useful Links:
    - **https://en.wikipedia.org/wiki/Tic-tac-toe (https://en.wikipedia.org/wiki/Tic-tac-toe)** (This is for those that don't know the game Tic Tac Toe)
    - **https://www.python-course.eu/tkinter_events_binds.php (https://www.python-course.eu/tkinter_events_binds.php)**
    - **https://steelkiwi.com/blog/working-tcp-sockets/ (https://steelkiwi.com/blog/working-tcp-sockets/)**

- https://www.geeksforgeeks.org/python-after-method-in-tkinter/ (https://www.geeksforgeeks.org/python-after-method-in-tkinter/)