Assignment 3 *Individual*

Tic-Tac-Toe Client-Server Assignment Instructions

Objective

In this assignment, you'll complete and enhance a client-server implementation of the classic game Tic-Tac-Toe. This project will help you understand network programming concepts, socket communication, and basic game logic implementation.

Setup

1. Ensure you have Python 3.6 or higher installed on your machine. Check your Python version by running:

   ```
   python -V
   ```

2. You have two Python files to work with:
   - `student_tictactoe_server.py`: The server-side script
   - `student_tictactoe_client.py`: The client-side script
3. No additional libraries are required; we're using Python's built-in `socket` module.

Tasks

1. Complete the Server Script (student_tictactoe_server.py)

a. Read the `check_winner` method and make sure you understand:

-
- Win combinations (rows, columns, diagonals) defined
- To check if any win combination is filled with the same non-empty symbol
  The return values: 'X' if X wins, 'O' if O wins, 'Tie' if it's a tie, None if the game is still ongoing

b. Implement the `handle_client_move` method:

- Validate the client's move:
  - Check if the move is a digit
  - Check if the move is within range (1-9)

    - Check if the chosen spot is empty
- If the move is invalid, send "Invalid move" to the client
- If the move is valid, update the board and break the loop

c. Implement game-over logic in the `play_game` method:

- After the client's move:
    - Check if there's a winner or tie
    - Send appropriate message to the client
    - Set `game_over` to True if the game has ended
- After the server's move:
    - Check if there's a winner or tie
    - Send appropriate message to the client
    - Set `game_over` to True if the game has ended
    - If the game hasn't ended, send the server's move to the client

2. Enhance the Client Script (student_tictactoe_client.py)

a. Implement the `get_player_move` method:

- Use a while loop to keep asking for input until a valid move is entered
- A valid move is a number between 1-9 that corresponds to an empty space on the board

b. Implement the `handle_server_response` method:

- Check different possible responses from the server:
    - "You win", "Tie", "Server wins", "Invalid move"
    - If it's none of these, it's the server's move (a number)
- Update the board with the server's move
- Return True if the game is over, False otherwise

Running the Game

1. Open two terminal windows or command prompts.
2. In the first window, run the server script:

```
python student_tictactoe_server.py
```

3. In the second window, run the client script:

```
python student_tictactoe_client.py
```

4. The client will attempt to connect to the server running on your local machine (server_name = '127.0.0.1').

Testing and Debugging

1. Play multiple games to ensure all scenarios are handled correctly.
2. Test edge cases (e.g., invalid moves, disconnections).
3. Use print statements liberally for debugging.
4. Pay attention to the game state on both client and server sides.

Tips

- Handle potential network errors and timeouts.
- Use meaningful variable names and add comments to explain your logic.
- Remember to close socket connections properly when the game ends.

Deliverables

1. Submit your completed client and server python files
2. Record a video thoroughly explaining both python files, including the code that you added and the code that was already written in these files. Show how the client and server are communicating using instructions.
3. In the same video, record yourself playing fully the game 3 times, and let your starting move be different each time you play a new game.

Good luck, and have fun implementing your Tic-Tac-Toe game!