Assignment #5

ECE 422C (Prof. Edison Thomaz) - The University of Texas at Austin – Fall 2023

Due Date: November 15th 2023 (See instructions for getting credit to this assignment below)

General Assignment Requirements

The purpose of this assignment is to use Java Socket Programming to build on the Mastermind board game from Assignment 2 with networking capabilities. You are free to use whatever classes and methods from the Java 8 library you wish. You may not use non-standard library features. We are providing you some sample code to start with.

In this assignment, you are asked to implement a game room application with both Server and Client. You should use Java Socket for network communication. You should design your application using OOP principles and the game requirements from Assignment 2. There is only one game room.

Server: The server is responsible for receiving guesses from clients and dispatching responses to appropriate clients. The server is responsible for generating the secret code. The server should support multiple clients. Multiple clients will independently play a game to determine the server's single code for a given round. You just need one server; call the main class of the server ServerMain.java. Make sure that ServerMain.java has a main() method.

Client: Clients submit guesses to the server and receive a message in return. Each client has its own number of guesses and can only receive feedback from their own guesses. The client that is the first to guess correctly is declared the winner of the game. If a client runs out of its own guesses, it should hang and wait until the round is over. If one client wins (guesses the code), the round should be over for all clients and the new game message should be prompted. Make sure that you have a ClientMain.java file with a main() method in it.

Some additional details: Clients are allowed to immediately play a new game once the server broadcasts a message (or acknowledgement) that a new game has started. This means that the server does *not* need to wait on acknowledgements from every client before starting a new game. You can assume that the client will send a response in a timely manner to the server (e.g., you will not be tested on an edge case where a client does not respond for an entire game).

Submission and grading

Name of zip file: Assignment_5_UTEID.zip

Put your code in a package named assignment_5, zip all your files and name the zip file as `Assignment_5_UTEID.zip`. Do not turn in your test code. Please make sure that the structure of the final zip is as follows, when unzipped:

Assignment_5_UTEID/
- README.pdf
- <other non-code files>
- <executable jar files for server and client>
- src/
  - ServerMain.java
  - ClientMain.java
  - <other code files>

To get credit for this assignment, you will need to meet with the TAs on one of the following days: November 10 (early checkout), 16, 17, or 18. Checkout locations outside of recitation will be announced on Canvas. Please sign up for a spot on this Google Sheets link. Please remember to submit to Canvas on, or before, November 15th 11:59 PM.

If you cannot meet with one of the TAs during recitation, schedule a time with them individually to show your working application. We will ask you to demo your implementation with at least 2 clients. When meeting with the TA, be prepared to show the source code of your application and answer questions about your design and implementation choices.