CS 342 **Project#3** **Fall 2023**

Multi-Threaded Server/Client Game

Due Date: Part 1: Sunday November 12th @11:59pm
Part 2: Tuesday November 21st @11:59pm
You may submit up to 24 hours late with a 10% penalty.

Description:

You will create a server that plays a word guessing game with each client that connects to that server (similar to Wheel of Fortune or Hangman). Each client that logs into the server must guess 3 different words in 3 different categories to win. First, the client picks a category. The server will send the client the number of letters in a word from that category to guess. The client gets to guess a total of six letters, one at a time. ***Correct guesses do not count towards the six guess total.***
The client will guess a letter and send it to the server. The server will respond with either: the letter is in the word and where that letter is located or the letter is not in the word and how many guesses are left.
If the client guesses the word within 6 letter guesses, they can not guess at another word in the same category but must chose from the two remaining. If they do not guess the word correctly, they are free to choose from any of the three categories for another word. Clients may guess at a maximum of three words per category. If they do not make a correct guess within three attempts, the game is over. The game is won when the client successfully guesses one word in each category. When the game is over, the client can either play again or quit.

Part #1, the planning:

You will create a class diagram for both your server program and for you client program. You will also create a wireframe for your client graphical interface. Submit these as PDFs, as many pages as needed, to the link on BB for Part #1. If you worked in a group, include a page with both of your names, netIDs and who worked on what.

CS 342 Project#3 Fall 2023

Implementation Details:

You will create two separate programs, each with a GUI created in JavaFX, one for the server and one for the client. Use the Maven template provided for each program.

You will need to change the <artifactId> in the pom file for each program: for instance, the server program would have:
<artifactId>serverProgramProjectThree</artifactId>

And the client program would have:
<artifactId>clientProgramProjectThree</artifactId>

All networking must be done utilizing Java Sockets (as shown in class). **The server must run on its own thread and handle each client on a separate thread. Each client must connect and communicate with the server on a separate thread.** You may not use libraries or classes not included in the standard Java 8 or 11 release.

The server:

When the server program starts, you must be able to enter in the port number to listen to and start the server. The server will NOT limit the number of clients connected at one time. For instance, if there are 8 clients connected to the server, then there will be eight individual games being played on eight separate threads.

The server will maintain a list of words in 3 different categories(for example: Animals, U.S. States, Superheroes, Foods …..etc).

The server will NEVER send the word to the client, only the number of letters, respond to guesses and keep track of remaining guesses, keep track of categories and words guessed as well as winning and losing of the game.

The server will not send the client the same word twice in any one game.

You will utilize a serializable class as the only means of passing information between the server and each client.

The serverGUI:

You must provide a way to enter the port to listen to and start the server.
You must provide a server log that reports on what the server is doing (similar to what was demonstrated during our BB Collab session)
You may add anything else you feel is appropriate or needed.

CS 342 Project#3 Fall 2023

The Client GUI:

You must provide a way for the client to log into the server
You must create an interface, utilizing the design principles discussed in class that is intuitive and allows the client to play and understand the progress of the game.

IT IS YOUR RESPONSIBILITY TO MAKE SURE THE USER KNOWS HOW TO PLAY YOUR GAME. IF THE GRADER OF YOUR PROJECT CAN NOT FIGURE OUT HOW TO PLAY IT, SIGNIFICANT POINTS WILL BE DEDUCTED.

There will be **5 points extra credit** for exceptional and creative UI and UX design. This is completely up to the judgement of who is grading your project (yes, it is subjective….just like users of your applications in real life).

Use of templates found on the web:

You may use ideas from GUI templates(fxml, css, images or other) for styling your programs found on the web. You may not import those templates and pass them off as your own work. If you use an idea or part of such template, include a reference to that work in the header of the file where used. Failure to do this will result in academic misconduct charges.
All other logic must be your original work.

Testing Code:

A minimum of 10 unit tests are required. They can be in either the server or client program or in both.

Electronic Submission:

If you worked in a group:

- only one of you needs to submit a project: for both part #1 and #2.
- You **must** include a PDF file called Collaboration.pdf. In that document, put both of your names and netIds as well as a description of who worked on what in the project for both part #1 and part #2.
- If you worked alone, no need for the Collaboration.pdf.

CS 342 Project#3 Fall 2023

For part one, submit your pdf to the link for part #1 on BB. For part two, zip both projects, server and client, together in a single archive and submit to the link for part #2 on BB. If you worked with a partner, submit the collaboration PDF to the collaboration doc link when you submit part #2. Name the pdf for part #1 and the zip for part #2 with your netid + Project3: for example, I would have a submission called mhalle5Project3.zip.

Please only submit one project per team.

Assignment Details:

There will be a 10% penalty for any project submitted up to 24 hours late. Anything after that will not be accepted and result in a zero.

We will test all projects on the command line using Maven 3.6.3. You may develop in any IDE you chose but make sure your project can be run on the command line using Maven commands. Any project that does not run will result in a zero. If you are unsure about using Maven, come see your TA or Professor.

This is a team project. You may share and communicate at will with your teammates but are forbidden from collaboration with other teams.

Unless stated otherwise, all work submitted for grading *must* be done individually. While we encourage you to talk to your peers and learn from them, this interaction must be superficial with regards to all work submitted for grading. This means you *cannot* work in teams, you cannot work side-by-side, you cannot submit someone else's work (partial or complete) as your own. The University's policy is available here:

https://dos.uic.edu/conductforstudents.shtml.

In particular, note that you are guilty of academic dishonesty if you extend or receive any kind of unauthorized assistance. Absolutely no transfer of program code between students is permitted (paper or electronic), and you may not solicit code from family, friends, or online forums. Other examples of academic dishonesty include emailing your program to another student, copying-pasting code from the internet, working in a group on a homework assignment, and allowing a tutor, TA, or another individual to write an answer for you. It is also considered academic dishonesty if you click someone

else's iClicker with the intent of answering for that student, whether for a quiz, exam, or class participation. Academic dishonesty is unacceptable, and penalties range from a

letter grade drop to expulsion from the university; cases are handled via the official student conduct process described at https://dos.uic.edu/conductforstudents.shtml.