CS 342 **Project#4** **Fall 2020**

Threaded AI Tic Tac Toe

Due Date: Sunday, November 22nd @11:59pm

You may submit up to 24 hours late for a 10% penalty

Description:

In this project, you will create an application with a graphical user interface that has two computer opponents playing games of tic tac toe against each other. All of the moves, for each computer opponent, will be chosen by using the AI Min-Max algorithm included. Game play must be on a separate thread (other than the application thread) and each computer opponent will utilize the Min-Max algorithm on a separate thread that will return the result of the algorithm. All threading must be done using the Executors class in Java. This project will be developed as a single Maven project. You can use the same Maven template that you downloaded for project #2 from BB. You should change the artifact ID in the POM file to "AITicTacToe.

This is an individual project. You are NOT allowed to work together on this.

How the program works:

The game will start with a welcome screen that is displayed for a few seconds and then goes to the game play screen. The user can select the game play level for each computer opponent (novice, advanced, expert). The user will also select how many games the computer opponents will play at the current level settings(1-10 games). Once those selections are made, the user can start and the program will begin playing the first game. All moves will be displayed on the tic tac toe board for the user to see and there should be a proper amount of time between moves to simulate actual game play. When one of the computer opponents wins or there is a tie, a third screen will show the results of that game. It will remain visible for a few seconds to allow the user to read it and then return to the game play screen to either, play the next game or allow the user to again set the play level for each opponent and number of games.

Implementation Details:

Your user interface will be created using JavaFX. It should include the following:

- An initial scene welcoming people to the game
- A second scene for game play including
  - instructions on how to play
  - the tic tac toe board
  - a way for the user to select how many games will be played in a row
  - a way to set the level of play for each computer opponent
  - a way to keep score (games won) for each computer opponent
  - a way to start play
  - a way to exit the program
  - a way to communicate to the user what they need to do
- A third scene that will display the result of each individual game played
  - this will be shown after each game and then automatically return to the second scene after a few seconds time.

The two computer opponents are playing against each other. The user of your program is essentially "watching" them play games of tic tac toe. **Each move for both opponents must be the result of running the Min-Max algorithm on a separate thread and returning the result of the algorithm in a Future object.**
Remember that the get() method for a Future object is a blocking call and should not block the application thread.

The Min-Max Algorithm:

In the included code, the Min-Max algorithm takes in the current state of the Tic Tac Toe game. It assumes that the next move is always **X.** This algorithm goes back and forth between two recursive methods, exploring every possible state of the game for every possible next move by **X** and response by **O**. When it reaches an end state, it will percolate that result up as the recursion unfolds and give **X** a score for each possible current move.

You should run the code with several different states to understand what it is doing!

You will need to include the .java files for the Min-Max algorithm in your Maven project. You may only use the Min-Max code provided for deciding each move of the computer opponents.
The algorithm will always return the best move. You can consider this “expert” mode. You are free to determine the strategies for “novice” an “advanced”.

Testing Code:

You are required to include JUnit 5 test cases for your program. Add these to the src/test/java directory of your Maven Project.

UI and UX design:

You are required to use the best practices we discussed in lecture for designing your programs user interface. The user should know what is happening and what to do at all times. Your interface should be intuitive and not cluttered.

Electronic Submission:

Zip your Maven project and name it with your netid + Project4: for example, I would have a submission called mhalle5Project4.zip, and submit it to the link on Blackboard course website.

Assignment Details:

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.

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.