COS 161 – Algorithms in Programming

Project 01 – Tic Tac Toe

Objectives

The objective of this assignment is to become familiar with creating multi-dimensional Arrays and utilizing them to store information. This project will also get you familiar with the DrawingPanel.java utility and give you a good handle on creating Java projects and files.

General Instructions for all Assignments

For each assignment you will write a Java program and test it. Start your programs with a comment block such as:

```
/*
   NAME: <your name>
   COS 161, Spring 2021 2020, Prof. <instructor name>
   Project ##
   File Name: CLASS_NAME.java
*/
```

Your programs should be neatly formatted, and follow the indenting, formatting, spacing, and commenting practices taught in class.

Read each assignment carefully to see what is required. If you do not understand something in the assignment, ask a tutor or the instructor. You will get partial credit if you finish only part of an assignment or it is not working correctly. Turn it in and explain what you completed and what issues it has. It is usually better to turn in an imperfect assignment on the due date, rather than falling behind in the course.

Part 1 (60 points) Regular Tic Tac Toe

For this part, we are going to attempt to recreate the age-old game of Tic Tac Toe. If you are unfamiliar with the game, check out the following Wikipedia article: https://en.wikipedia.org/wiki/Tic-tac-toe

Create a new Java project and add the DrawingPanel.java file. Also create class called TicTac.java. This will be the main class for Part 1. In this class you must create a game of Tic Tac Toe. Paste in the following code:

```
import java.awt.*;

public class TicTac {

     public static void main(String[] args) {
          // Creates drawing panel
          DrawingPanel panel = new DrawingPanel(250, 250);
          Graphics g = panel.getGraphics();

          g.drawLine(100, 50, 100, 200);
          g.drawLine(150, 50, 150, 200);
          g.drawLine(50, 100, 200, 100);
          g.drawLine(50, 150, 200, 150);

          /*
          Useful for clearing board
          g.setColor(Color.WHITE);
          g.fillRect(45, 45, 155, 155);
          */
     }
}
```

For a refresher on DrawingPanel.java and its uses, check Supplement 3G in the textbook.

Run the program and see what you get for output, should look like an empty Tic Tac Toe board. Now, you must do the following:

- Create a 2-dimensional array of type char called gameBoard to store the game information, this should be static and outside of the main method (so it can be accessed from other methods easily)
- Create a method called drawBoard that draws the board (reuse the supplied code or write your own) and places the X's and O's based on the values in gameBoard. Use the DrawingPanel for this.
- Create a method called takeTurn that prompts the user to enter input in the console for the coordinates of their move. It should take in a parameter of what player is taking their turn (char for X or O). This method should check to see if the move is valid, as well as handle any exceptions if the user enters something unexpected. If the move is valid, it should update the gameBoard and call the drawBoard method.
- Create a method called checkWin that checks if either player has won and returns the char corresponding to the winning player. It should

also return a specific char if it detects a draw has occurred and should return some other char if no player has won (meaning there are still empty spaces).

- Modify main to call takeTurn repeatedly and alternatingly and checkWin until one player has won or there is a draw. Have it print out who the winner was or that there was a draw.

Part 2 (20 points) Advanced Tic Tac Toe

Copy and paste your TicTac.java file into your project (rename it TicTacPlus.java). Make the following modifications:

- Change the dimensions of the gameBoard to be 4x4, you must also modify the DrawingPanel dimensions
- Modify drawBoard to draw the new 4x4 board.
- Modify checkWin to check to see if any of the following new conditions are met:
  - 4 down in a column
  - 4 across in a row
  - 4 along a diagonal
  - 4 in a 2x2 box, meaning anywhere a 2x2 box could fit, you must check to see they are all Xs or Os (think algorithmically, how would you describe the steps or go about doing this manually?). Lab 01 may help with this one.

Extra Credit (10 points) Artificial Intelligence

Create a new class called TicTacFoe.java and copy over the methods and main code from Part 1. Create a method that can be called instead of the second player that plays against them. This "Computer" player should always make the game end in a draw. I would recommend you alternate calling takeTurn and this method (which will use much of the code from takeTurn, with added logic to look at gameBoard and make strategic moves).

Note: WarGames (https://en.wikipedia.org/wiki/WarGames) references are appreciated but are not required for extra credit points. In my opinion, it is a great film though, check it out if you get the chance and have never seen it!

What to Turn In

Turn in a solution document as a single pdf file. This solution document should contain the following (make sure formatting is clear):

- The takeTurn() and checkWin() methods
- At least two screenshots of console input of game being played.
- At least three screenshots of the game board DrawingPanel
- If you completed the extra credit, include 3 additional screenshots, one from the console output and two from the DrawingPanel. Make sure these are clearly marked “Extra Credit”.

Once you have the solution document ready, submit it as an attachment on Brightspace. Be sure to also attach all the .java files you created or edited in this project. Do not zip or compress the files, submissions with archives of any type will be ignored.