Yijun Liu

CS 170Q

Dr. Li

2 Dec 2020

Tic Tac Toe Report

Design Basics

This project is aimed to develop a Tic Tac Toe board game and present it with GUI. After consulting with Dr. Li, I designed a user against computer Tic Tac Toe game. Users can click the space on the board to place a chess piece to start, and computer will then play. Therefore, each step will be shown on the board. Also, I included multiple sizes of the game boards, ranging from 3 x 3 to 16 x 16. The game ends when there is a winner or a tie.

Data Structures

To fulfill the design goals in a clear manner, I used three classes:

1. ***Chess*** class to record each chess piece.

   In the Chess class, I used x and y to represent the coordinates of the chess; I also used a boolean variable to determine who the player is.

2. ***Gameboard*** class included methods to paint the canvas, check results (winner/tie), and to handle the mouse events.

   In the Gameboard class, I overrode the paintComponent(Graphics g) method to graph the board. After each click, the method will be called and repaint the board. I wrote a computer() to randomly place a chess piece on the board by the program. I did not use any algorithm to perfect the difficulty. For the mouse events, I overrode the

mousePressed(MouseEvent arg0) method to place the chess piece, check each step, and decide whether there is a winner or a tie or not. I also overrode the mouseMoved(MouseEvent e) method so that the cursor will change to a hand shape when it moves to places that can place chess pieces. To check whether there is a winner or a tie, I wrote the following methods: checkResult(), checkFull(), checkRow(char inputCha), checkCol(char inputCha), checkDia(char inputCha), and checkAntidia(char inputCha). checkFull() is to check whether the chess board is full of chess pieces or not; checkRow(char inputCha), checkCol(char inputCha), checkDia(char inputCha), and checkAntidia(char inputCha) are used to check whether there is a winner winning from fulfilling a row, a colu

mn, the diagonal line, or the antidiagonal line respectively. checkResult() is the overall check function that uses the methods mentioned above.

3. ***Start*** class used to set the game frame, place the restart and exit buttons, and allow users to choose the board size.

   In the start class's constructor, I added each component in the user interface using swing and awt. I included a ComboBox and two buttons. The ComboBox is used to select the board size, with an itemlistener bonded to it; one of the buttons is restart, and the other button is the exit. For the two buttons, because there are actions need to be performed, I created an object to link the events to the buttons. There is also the main function that used to start the game.

Critical Points

- The game is a user vs. program game.

- User needs to start the game first.
- After choosing a board size, users need to click restart to refresh the game board.
- Each step is shown on the board.
- The game will end when there is a winner or a tie. The program will send a message on the result (you won/you failed/tie).

Explanations

I did a computer vs. user because I found it more interesting to play with the program. This program is evolved from a two-user game.

I limited the board size to 16 x 16 because I found it very difficult to display chess pieces for board larger than 16 x 16.