University of Nebraska – Kearney
CSIT 150: Object Oriented Programming
HW4 – GUI, Connect-4 Game Display Program Challenge

Connect-4 Game Display Program Challenge. See the HW4 program for the multiple choice questions and the other details of this assignment.

This assignment is part 1 of 2 parts of the implementation of the game Connect-4. Part one will consist of supplying the GUI display of the game. This will consist of:

1. Drawing the game board
2. Providing the event handlers that will react to mouse clicks by placing the alternating colored balls in the columns clicked.
3. Creating a window to hold the game board

The full game will be implemented with at least three classes, you may use more if you wish, but these three must be present:

1. **Con4Game**: This class supplies all logic for the game. A barebones game file of this file has been provided for you to start this assignment. *You are not to change any of the code found in this file for this assignment.* This has been supplied to aid your understanding of the object oriented paradigm. In HW 5, you will be drastically altering this file. At this point, there should be no logic in the game except to track the positions of the game tokens added to the board by the players.

2. **Connect4Pane:** This class creates the graphic display by extending a Pane. It is responsible for the event handlers listening for mouse clicks and for telling the game which column was clicked as related to the game (not coordinates). When the mouse is clicked, should store a token in the appropriate spot in the Con4Game, by calling its makeMove() method. (You will have to figure out which spot based on the x coordinate of the mouse click.) No other data structure is need in this class, except if you would like an array of colors.

   You will need to call a **`paint()`** method to draw your game. You'll need to draw a circle (with your choice of color) when a player wants to position its checker piece in a column. If there is already a checker in that column, the new checker is placed on top of the existing checker.

*I'm leaving the **`paint()`** method's details a little vague … some of your grade will be determined from the clarity and creativity you use in your drawing. See chapter 14 and the animation examples on Canvas for more information on drawing graphical components. Include items such as:*

```
Label lab = new Label("Players turn");
getChildren().clear(); //clear the last time the pane was painted
getChildren().add(lab); //Add the label to the top of the pane
For each column:
        Draw the background rectangle add it to the pane
        Using a loop draw circles that correspond with the array of tokens stored in  that
        column of the Con4Game's array.
```

3. **Connect4Window**: Is a container for the game board by extending **Application**. This is the executable class and as such needs a main method. The basic logic required of this class for

this program is to set up the stage, scene, menus and panes. (One of those panes will be the Connect4Pane.). Menus will be required in HW 5 and are bonus in this assignment.

For this assignment, the focus is on the Connect4Pane and the Connect4Window classes. The main task is to install the mouse listeners in the pane and to capture the coordinates of the mouse click, so you can calculate which column was clicked.

The diagram below is an example of a possible display. It is merely a suggestion. You may make your display look however you would like as long as the following criteria are met:

- Clearly indicated columns
- Player tokens should be in two colors, one per player.
- Text area that indicates which player has the current turn.

As a heads up, for HW5 assignment, you can choose to complete the Connect-4 game, continue with the robot maze or write your own game. If you choose to complete theConnect-4 game, you will need to fill out the class Con4Game which will include such features as validating move (detecting a full column), detecting winning configuration, detecting ties, clearing a game, saving a game, tracking best scores per player name, … etc. As you work on this assignment, begin to consider how you would work these more advance features into your implementation.