Project 02: Connect 4

Release date: Friday September 30th, 2016
Due date: Friday October 14th, 2016 11:59 pm

Connect Four is a two-player connection game in which the players first choose a color and then take turns dropping colored discs from the top into a seven-column, six-row vertically suspended grid. The pieces fall straight down, occupying the next available space within the column. The objective of the game is to connect four of one's own discs of the same color next to each other vertically, horizontally, or diagonally before your opponent.

You can play the game here: Connect 4

In this project, you are going to write your own version of Connect Four. The two players of the game will be both human, that is, the move to be applied will be always chosen by the user(s). You will be creating a simple (non graphical) interface that will facilitate the game for the user.

Connect4 class

This is the class that will represent the game. You will need a way to represent a board and a way to represent the disks. Taking advantage of the object oriented nature of java, you will use instance variables to do this:

- A 6x7 two-dimensional character array to represent the board
- Constant characters to represent the colored discs: 'O' for red, 'X' for yellow and ' ' for an empty space.

You are required to provide an implementation for the following methods:

- `public Connect4()`

The constructor that initializes the instance variables.

- `public char[][] getBoard()`

This method should return a copy of the current board, that is the 2 dimensional array holding the game board.

- `public int putPiece(int column, char color)`

This method inserts a piece of a particular color in the particular column. The color and column are passed as arguments to the method. Remember that we don't need the row as an argument because the piece always goes to the lowermost row as it would happen in the real Connect4. game. The method returns the row where the piece have been put.

NOTE: Since we are representing this in a 2-d array, the highest row is 0 and it goes increasing when going down.

- `public char checkAlignment(int row, int column)`

This method checks the board to see if any of the two players has got 4 pieces in a row. Make sure you check vertically, horizontally and diagonal. You need to call this method every time a piece has been put into the board.

Hint: You don't have to check the whole board every time. The method returns the color if an alignment exists, or the char representing an empty space otherwise.

- `public void printScreen()`

Print the board of the game to the console to show the users the status of the game. There are some screenshots to give you an example of how this board might look, however, you are free to implement this in any way.

- `public void play()`

This method simulates a round of the game. The red player will always start the game. You should prompt the user for a column to play and call the `putPiece` method. Remember to swap the player after each round. The game should continue until any of the two players makes 4 pieces in a row, use the `checkAlignment` method for this.

Additionally, you need to make sure that the input column is valid. If it is not, you need to keep asking the user until you get a valid position. Make sure to report the winner when the game is done.

Game class

Create a class call `Game` that only contains a main method. The main method creates a Connect4 object and launches the game.

Grading Rubric

- **5% - Constructor**
- **40% - checkAlignment**
    - 8% - vertical
    - 8% - horizontal
    - 8% - diagonal /
    - 8% - diagonal \
    - 8% - no alignment
- **10% - putPiece**
- **5% - getBoard**
- **20% - printScreen**
- **20% - play**

Sample Output

Here are some screenshots of how the game might run

1. Player 1 chooses column 3.

```
    0   1   2   3   4   5   6
  -----------------------------
A |   |   |   |   |   |   |   |
  -----------------------------
B |   |   |   |   |   |   |   |
  -----------------------------
C |   |   |   |   |   |   |   |
  -----------------------------
D |   |   |   |   |   |   |   |
  -----------------------------
E |   |   |   |   |   |   |   |
  -----------------------------
F |   |   |   |   |   |   |   |
  -----------------------------
Current player: 'O'
Choose a column: 3
```

2. Player 2 chooses column 2.

```
    0   1   2   3   4   5   6
  -----------------------------
A |   |   |   |   |   |   |   |
  -----------------------------
B |   |   |   |   |   |   |   |
  -----------------------------
C |   |   |   |   |   |   |   |
  -----------------------------
D |   |   |   |   |   |   |   |
  -----------------------------
E |   |   |   |   |   |   |   |
  -----------------------------
F |   |   |   | O |   |   |   |
  -----------------------------
Current player: 'X'
Choose a column: 2
```

3. Player 1 chooses column 3.

```
      0   1   2   3   4   5   6
  -----------------------------
A |   |   |   |   |   |   |   |
  -----------------------------
B |   |   |   |   |   |   |   |
  -----------------------------
C |   |   |   |   |   |   |   |
  -----------------------------
D |   |   |   |   |   |   |   |
  -----------------------------
E |   |   |   |   |   |   |   |
  -----------------------------
F |   |   | X | O |   |   |   |
  -----------------------------
Current player: 'O'
Choose a column: 3
```

4. Player 2 chooses column 2.

```
    0   1   2   3   4   5   6
  -----------------------------
A |   |   |   |   |   |   |   |
  -----------------------------
B |   |   |   |   |   |   |   |
  -----------------------------
C |   |   |   |   |   |   |   |
  -----------------------------
D |   |   |   |   |   |   |   |
  -----------------------------
E |   |   |   | O |   |   |   |
  -----------------------------
F |   |   | X | O |   |   |   |
  -----------------------------
Current player: 'X'
Choose a column: 2
```

5. Player 1 chooses column 3.

```
 0   1   2   3   4   5   6
  -----------------------------
A |   |   |   |   |   |   |   |
  -----------------------------
B |   |   |   |   |   |   |   |
  -----------------------------
C |   |   |   |   |   |   |   |
  -----------------------------
D |   |   |   |   |   |   |   |
  -----------------------------
E |   |   | X | O |   |   |   |
  -----------------------------
F |   |   | X | O |   |   |   |
  -----------------------------
Current player: 'O'
Choose a column: 3
```

6. Player 2 chooses column 2.

```
    0   1   2   3   4   5   6
  -----------------------------
A |   |   |   |   |   |   |   |
  -----------------------------
B |   |   |   |   |   |   |   |
  -----------------------------
C |   |   |   |   |   |   |   |
  -----------------------------
D |   |   |   | O |   |   |   |
  -----------------------------
E |   |   | X | O |   |   |   |
  -----------------------------
F |   |   | X | O |   |   |   |
  -----------------------------
Current player: 'X'
Choose a column: 2
```

7. Player 1 chooses column 3.

```
  0   1   2   3   4   5   6
  -----------------------------
A |   |   |   |   |   |   |   |
  -----------------------------
B |   |   |   |   |   |   |   |
  -----------------------------
C |   |   |   |   |   |   |   |
  -----------------------------
D |   |   | X | O |   |   |   |
  -----------------------------
E |   |   | X | O |   |   |   |
  -----------------------------
F |   |   | X | O |   |   |   |
  -----------------------------
Current player: 'O'
Choose a column: 3
```

8. Player 2 chooses column 2.

```
 0   1   2   3   4   5   6
  -----------------------------
A |   |   |   |   |   |   |   |
  -----------------------------
B |   |   |   |   |   |   |   |
  -----------------------------
C |   |   |   | O |   |   |   |
  -----------------------------
D |   |   | X | O |   |   |   |
  -----------------------------
E |   |   | X | O |   |   |   |
  -----------------------------
F |   |   | X | O |   |   |   |
  -----------------------------

!!! Winner is Player 'O' !!!
```

9. Player 1 Wins.

Submission

Submit your Team.java file to Vocareum by following these instructions. Keep in mind that only your last submission will be considered.