Overview

Connect 4 is a two-player game. Players drop tokens into columns, trying to connect 4 of their tokens either diagonally, horizontally, or vertically. In this project, you'll write logic for a virtual Connect 4 and create an AI opponent using minimax.

The project is split into 4 files:

1. **Main.java** - Runnable class with JFrame stuff
2. **Connect4.java** - Displays the pieces and handles mouse interaction
3. **Connect4AI.java** - Performs minimax to determine the best move found for the AI
4. **CheckWin.java** - Helper class that takes a board state and sees if one side has won.

More about Connect 4: https://en.wikipedia.org/wiki/Connect_Four

Steps

Step 0

Download and look over the starter code.

Step 1

(Connect4.java)

There are 3 possible states for a square on the board: has the player's token, has the AI's token, or is empty.

The empty state is already handled by the char ' ' (blank). Choose the other 2 characters, and complete the declarations.

```
private final char PLAYER_CHAR = _____;
private final char AI_CHAR = _____;
```

Step 2

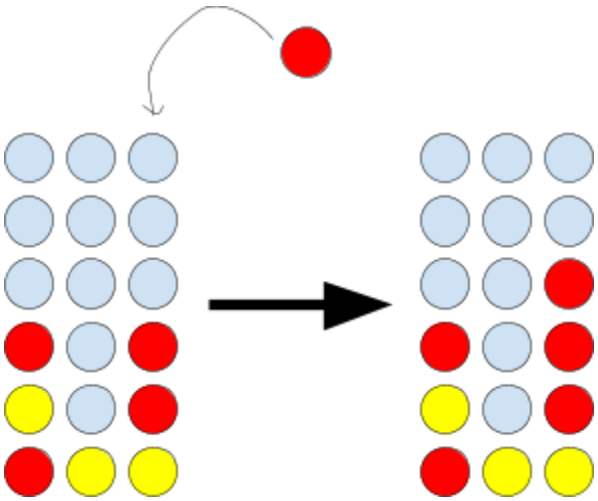
(Connect4.java)

Use a nested for loop to initialize the board in the constructor. At the start of the game, all squares should be empty.

Step 3
(Connect4.java)
Correctly display the tokens with the right colors (inside paintComponent).

Step 4
(MoveMaker.java)
Complete the MoveMaker class.

In the makeMove method: For the specified column, the program should place a token as far down as possible without overlapping with other tokens (to simulate dropping). If the column is full and we can't add another token, return false.



(A picasso-like rendition of this idea)

Likewise, for the undoMove method: the program should remove the last token placed (the token furthest up). If the entire column is empty and we can't remove anything, return false.

You might notice that the methods in MoveMaker.java are static. This is so that instead of writing

```
MoveMaker m1 = new MoveMaker();
m1.methodInsideClass();
```

You can instead write

```
MoveMaker.methodInsideClass();
```

Which is less of a hassle.

Step 5
(Connect4.java)
Using the playMove method, fill in the mouseClicked method. If the player clicks while the mouseX is within a certain column, the token will drop into that column. This can be achieved using if/else-if statements (or similar).

Your code should look something like

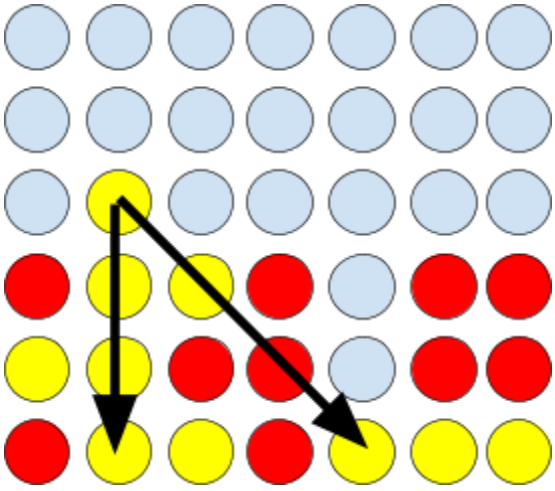
```
if( e.getX() > ___ && e.getX() < ___ && MoveMaker.canMakeMove(0) )
        MoveMaker.playMove(0);
else if( e.getX() > ___ && e.getX() < ___ && MoveMaker.canMakeMove(1) )
        MoveMaker.playMove(1);
…
```

(The size of the columns, SQUARE_SIZE, is given to you as a constant.)

Step 6
(CheckWin.java)
Write the CheckWin.java class.

checkDiagonal should see if a ***diagonal*** winning sequence starts at the specified position.

checkRow and checkCol should see if a ***horizontal*** or ***vertical*** winning sequence starts at the specified position.



Row 2, column 1
checkDiagonal(): nope, the yellow tokens don't form a sequence of 4.
checkCol(): yes, the yellow tokens form a sequence of 4.

Step 7
(Connect4.java)
Update the mouseClicked method. The structure should look something like

1. Make player move
2. Check if the player won with that move
3. repaint();

Step 8
(Connect4AI.java)
Implement minimax search in Connect4AI.java. This is the hardest part of the program. Feel free to either use the comments as a guide, or change the details depending on your preference.

Helpful resource: https://en.wikipedia.org/wiki/Minimax#Pseudocode

Step 9
(Connect4.java)
Update the mouseClicked method. It should look something like

1. Make player move
2. Check if the player won with that move
3. repaint();
4. Make AI move
5. Check if the AI won with that move
6. repaint();

Step 10
(Connect4.java)
Handle “game over” events. Nothing too fancy, a console message saying “You lost” or “You won” works.

The player should be prevented from making any more moves after a game is finished.

Step 11
Test everything to make sure things work as intended. Remember to fill out the “author” and “version” fields at the top of each class before submitting.

Extension Ideas

1. Add a menu screen with buttons (custom or JButton)
2. Record the moves and game result in a file.
3. Better error checking - for example, trying to place tokens in a full column