UF UNIVERSITY of FLORIDA
Herbert Wertheim College of Engineering

COP3502

Lab 05: Connect-Four

Overview

This lab is designed to introduce students to 2-D arrays by recreating one of everyone's favorite childhood games: Connect-Four. You will loop through arrays, and manipulate them. Your end product should be robust enough to not have a single Out-of-bounds Exception!

Specification

You will first start by asking the user for what they wish the height and length of the board to be:

```
What would you like the height of the board to be? 4
What would you like the length of the board to be? 5
```

Then you will print the empty board:

```
- - - - -
- - - - -
- - - - -
- - - - -
```

And tell the players what their tokens are:

```
Player 1: x
Player 2: o
```

The players will take turns placing their tokens by choosing columns…

```
Player 1: Which column would you like to choose? 0
- - - - -
- - - - -
- - - - -
x - - - -

Player 2: Which column would you like to choose? 3
- - - - -
- - - - -
- - - - -
x - - o –
```

…until one of them wins!

```
Player 1: Which column would you like to choose? 0
x - - - -
x - - - -
x - o - -
```

```
x o x o o

Player 1 won the game!
```

Or until there is a tie!

```
Player 2: Which column would you like to choose? 2
o x o x o
x o o x x
x o o o x
x o x o x

Draw. Nobody wins.
```

Elements in the array should be accessible via row-major indexing (**board[row][column]**). In addition, you should follow the structure by storing the chips (inserting 'x' in column 0 and then inserting 'o' in column 2) in a 2D list array starting from row 0:

```
Row 0       x - o - -
Row 1       - - - - -
Row 2       - - - - -
Row 3       - - - - -
```

When printing the board, you should print the board upside down (by printing from the last row which is row 3 in our example). So that your output on the console after calling print_board(array) would be:

```
- - - - -
- - - - -
- - - - -
x - o - -
```

Assumptions

Students can assume that:

- the user will choose for the board dimensions to be 4x4 or greater.
- the user will input a valid column number (from 0 to length-1).
- the column that the user choses to place their token into has space (it is not filled already by other tokens).
- players can only win vertically or horizontally, but not diagonally.

Required Methods

`def` **`print_board`**`(array)`
This will take in the 2D character array for the board and print the board.

`def` **`initialize_board`**`(array)`
This will take in the 2D character array for the board, and this will set each spot in the array to "-".

`def` **`insert_chip`**`(array, col, chip_type)`
This will take in the 2D character array for the board. This function places the token ('x' or 'o' denoted as 'chipType') in the column that the user has chosen. Will find the next available spot in that column if there are already tokens there. The row that the token is placed in is returned.

`def check_if_winner(array, col, row, chip_type)`
This will take in the 2D character array for the board. After a token is added, checks whether the token in this location, of the specified chip type, creates four in a row. Will return True if someone won, and False otherwise.

Hint: Implement the methods in this order.

Submission

NOTE: Your output must match the example output *exactly*. If it does not, ***you will not receive full credit for your submission***!

Files: connect_four.py
Method: Submit on ZyLabs

Sample Output

```
What would you like the height of the board to be? 4
What would you like the length of the board to be? 5
- - - - -
- - - - -
- - - - -
- - - - -

Player 1: x
Player 2: o

Player 1: Which column would you like to choose? 0
- - - - -
- - - - -
- - - - -
x - - - -

Player 2: Which column would you like to choose? 3
- - - - -
- - - - -
- - - - -
x - - o -

Player 1: Which column would you like to choose? 0
- - - - -
- - - - -
x - - - -
x - - o -

Player 2: Which column would you like to choose? 1
- - - - -
- - - - -
x - - - -
x o - o -

Player 1: Which column would you like to choose? 0
- - - - -
x - - - -
```

```
x - - - -
x o - o -

Player 2: Which column would you like to choose? 4
- - - - -
x - - - -
x - - - -
x o - o o

Player 1: Which column would you like to choose? 2
- - - - -
x - - - -
x - - - -
x o x o o

Player 2: Which column would you like to choose? 2
- - - - -
x - - - -
x - o - -
x o x o o

Player 1: Which column would you like to choose? 0
x - - - -
x - - - -
x - o - -
x o x o o

Player 1 won the game!
```