Homework 2

Due Wednesday October 10th at 11:59 pm
100 Points

For this assignment we will continue to work on our ConnectX game, but we will be adding new requirements. The old requirements are still required, unless they are replaced by a requirement in this prompt.

New Requirements

- At the start of each new game the user will be able to provide the number of rows, columns and the number of markers in a row needed to win for the game. These parameters should be passed to the GameBoard class in the constructor. The maximum number of rows and columns for all GameBoards is 100, and the maximum number needed to win will be 25 for all GameBoards. GameBoard.java must still use a 2D array to represent it's game board, but it's ok if most of the array is ignored for smaller boards.
- Note that the formatting of the displayed board has changed to allow double digit numbers.
- If the users decide to play again, they will be able to change the size and number to win settings from the previous game.
- The program must validate the number of rows, columns, and needed to win that are provided by the user.
- You must provide an interface called IGameBoard that the GameBoard class will implement. The IGameBoard interface should contain all required methods of the GameBoard class (including new methods required in Homework 2.
- For each method, determine whether or not is a primary or a secondary method. Any Secondary methods in the IGameBoard interface should be provided as a default method in the interface itself.

New Methods to Add

<u>IGameBoard.java:</u>

Add the following methods to IGameBoard and GameBoard:

public int getNumRows() – returns the number of rows in the GameBoard
public int getNumColumns() – returns the number of columns in the GameBoard
public int getNumToWin() – returns the number of tokens in a row needed to win the game

The constructor should now take in the number of columns, the number of rows, and the number needed to win (in that order) as parameters.

No other methods (except those required from homework 1) should be public

Contracts and Comments

All methods (except for the main) must have preconditions and post-conditions specified in the Javadoc contracts. All methods (except for main) must also have the params and returns specified in the Javadoc comments as well. You must include a complete interface specification in IGameBoard, and

write the contracts not included in the interface specification in the GameBoard class. You must provide correspondences in your GameBoard class.

You code must be well commented. The Javadoc comments will only be enough for very simple methods. Remember, your comments will help the grader understand what you are trying to do in your code. If they don't understand what you are trying to do, you will lose points.

Sample input and output

```
How many rows should be on the board?
1
Must have at least 3 rows.
How many rows should be on the board?
200
Can have at most 100 rows
How many rows should be on the board?
8
How many columns should be on the board?
1
Must have at least 3 columns.
How many columns should be on the board?
200
Can have at most 100 columns
How many columns should be on the board?
12
How many in a row to win?
1
Must have at least 3 in a row to win.
How many in a row to win?
50
Can have at most 25 in a row to win
How many in a row to win?
5
| 0| 1| 2| 3| 4| 5| 6| 7| 8| 9|10|11|
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |

Player X, what column do you want to place your marker in?
15
Column cannot be greater than 11
Player X, what column do you want to place your marker in?
-1
Column cannot be less than 0
```

```
Player X, what column do you want to place your marker in?
4
| 0| 1| 2| 3| 4| 5| 6| 7| 8| 9|10|11|
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |X |  |  |  |  |  |  |  |

Player O, what column do you want to place your marker in?
7
| 0| 1| 2| 3| 4| 5| 6| 7| 8| 9|10|11|
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |X |  |  |O |  |  |  |  |

Player X, what column do you want to place your marker in?
5
| 0| 1| 2| 3| 4| 5| 6| 7| 8| 9|10|11|
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |X |X |  |O |  |  |  |  |

Player O, what column do you want to place your marker in?
8
| 0| 1| 2| 3| 4| 5| 6| 7| 8| 9|10|11|
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |X |X |  |O |O |  |  |  |
```

```
Player X, what column do you want to place your marker in?
9
| 0| 1| 2| 3| 4| 5| 6| 7| 8| 9|10|11|
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |X |X |  |O |O |X |  |  |

Player O, what column do you want to place your marker in?
7
| 0| 1| 2| 3| 4| 5| 6| 7| 8| 9|10|11|
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |O |  |  |  |  |
|  |  |  |  |X |X |  |O |O |X |  |  |

Player X, what column do you want to place your marker in?
2
| 0| 1| 2| 3| 4| 5| 6| 7| 8| 9|10|11|
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |O |  |  |  |  |
|  |  |X |  |X |X |  |O |O |X |  |  |

Player O, what column do you want to place your marker in?
6
| 0| 1| 2| 3| 4| 5| 6| 7| 8| 9|10|11|
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |O |  |  |  |  |
|  |  |X |  |X |X |O |O |O |X |  |  |
```

```
Player X, what column do you want to place your marker in?
1
| 0| 1| 2| 3| 4| 5| 6| 7| 8| 9|10|11|
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |O |  |  |  |  |
|  |X |X |  |  |X |X |O |O |O |X |  |  |

Player O, what column do you want to place your marker in?
7
| 0| 1| 2| 3| 4| 5| 6| 7| 8| 9|10|11|
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |O |  |  |  |  |
|  |  |  |  |  |  |  |O |  |  |  |  |
|  |X |X |  |  |X |X |O |O |O |X |  |  |

Player X, what column do you want to place your marker in?
3
| 0| 1| 2| 3| 4| 5| 6| 7| 8| 9|10|11|
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |O |  |  |  |  |
|  |  |  |  |  |  |  |O |  |  |  |  |
|  |X |X |X |X |X |O |O |O |X |  |  |

Player X Won!
Would you like to play again? Y/N
y
How many rows should be on the board?
5
How many columns should be on the board?
5
How many in a row to win?
3
| 0| 1| 2| 3| 4|
|  |  |  |  |  |
|  |  |  |  |  |
```

```
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Player X, what column do you want to place your marker in?
1
| 0| 1| 2| 3| 4|
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |X |  |  |  |  |

Player O, what column do you want to place your marker in?
3
| 0| 1| 2| 3| 4|
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |X |  |O |  |  |

Player X, what column do you want to place your marker in?
1
| 0| 1| 2| 3| 4|
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |X |  |  |  |  |
|  |X |  |O |  |  |

Player O, what column do you want to place your marker in?
1
| 0| 1| 2| 3| 4|
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |O |  |  |  |  |
|  |X |  |  |  |  |
|  |X |  |O |  |  |

Player X, what column do you want to place your marker in?
2
| 0| 1| 2| 3| 4|
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |O |  |  |  |  |
|  |X |  |  |  |  |
|  |X |X |O |  |  |
```

```
Player O, what column do you want to place your marker in?
2
| 0| 1| 2| 3| 4|
|  |  |  |  |  |
|  |  |  |  |  |
|  |O |  |  |  |
|  |X |O |  |  |
|  |X |X |O |  |

Player O Won!
Would you like to play again? Y/N
N
```

The Program Report

Along with your code you must submit a well formatted report with the following parts:

Requirements Analysis

Fully analyze the requirements of this program. Express all functional requirements as user stories. Remember to list all non-functional requirements of the program as well.

Design

Create a UML class diagram for each class and interface in the program. Also create UML Activity diagrams for every method in your Connect4Game class and every method in your GameBoard class (except the constructor). If a method is defined as a default method in the interface, you need to update it to remove any mention of the 2D array or other private data of the GameBoard class, and instead refer to primary methods.

Testing

No test cases are required for this assignment

Deployment

Provide instructions about how your program can be compiled and run. Since you are required to submit a makefile with your code, this should be pretty simple.

Additional Requirements

- You must include a makefile with your program. We should be able to run your program by unzipping your directory and using the make and make run commands.
- Remember, this class is about more than just functioning code. Your design and quality of code is very important. Remember the best practices we are discussing in class. Your code should be easy to change and maintain. You should not be using magic numbers. You should be considering Separation of Concerns, Information Hiding, Encapsulation, and Programming to the interface when designing your code. You should also be following the idea and rules of design by contract.
- A new game should always start with player X
- You class files should be in the cpsc2150.connectX package
- Your code should be well formatted and consistently formatted. This includes things like good variable names and consistent indenting style.
- You should not have any dead code in your program. Dead code is commented out code.

- Your code must be able to run on the school Unix machines. Usually there is no issue with moving code from intelliJ to Unix, but some times there can be issues, especially if you try to import any uncommon java libraries.
- Code that does not compile will receive a 0.
- Your input and output should match the example input and output provided. The TAs will prepare scripts to help with the grading process, and their scripts will rely on the input and output matching my example.
- Your UML Diagrams must be made electronically. I recommend the freely available program draw.io, but if you have another diagramming tool your prefer feel free to use that.
- While you need to validate all input from the user, you can assume that they are entering numbers or characters when appropriate. You do not need to check to see that they entered 6 instead of "six."

Tips and Reminders

- Do your design work before you write any code. It will help you work through the logic and help you avoid writing code just to delete it later.
- Carefully consider how you can use your available methods to solve problems both in your main function, and in methods in the GameBoard class.
- When writing your code, try to consider how things may change in the future. How can you design your code now to be prepared for future changes and requirements?

Submission

You will submit your program on Handin. You should have one zipped folder to submit. Inside of that folder you'll have your package directory (which will contain your code), your report (a pdf file) and your make file. Make sure your code is your java files and not your .class files.

Academic Dishonesty

This is an INDIVIDUAL assignment. You should not collaborate with others on this assignment. See the syllabus for further details on academic dishonesty.

Late Policy

Your assignment is due at 11:59 pm. Even a minute after that deadline will be considered late. Make sure you are prepared to submit earlier in the day so you are prepared to handle any last second issues with submission.

If your assignment is late, but turned in within 24 hours of the due date, you will automatically receive 25 points off. If it is more than 24 hours of the due date, but less than 48 hours of the due date, then you will receive 50 points off automatically. After 48 hours from the due date late assignments will no longer be accepted.

Checklist

Use this Checklist to ensure you are completing your assignment. Note: this list does not cover all possible reasons you could miss points, but should cover a majority of them. I am not intentionally leaving anything out, but I am constantly surprised by some of the submissions we see that are wildly off the mark. For example, I am not listing "Does my program play Connect 4?" but that does not mean that

if you turn in a program that plays Checkers you can say “But it wasn’t on the checklist!” The only complete list that would guarantee that no points would be lost would be an incredibly detailed and complete set of instructions that told you exactly what code to type, which wouldn’t be much of an assignment.

- Can I change the size of my game board?
- Can I set the number in a row needed to win?
- Did I create an interface for my Game Board?
- Did I provide my interface Specification?
- Did I move my contracts from my GameBoard class to my Interface?
- Did I implement all secondary methods as default methods in my interface?
- Does my game allow for players to play again?
- When players start a new game can they change the size of the board or the number to win?
- Does my game take turns with the players?
- Does my game correctly identify wins?
- Does my game correctly identify tie games?
- Does my game run without crashing?
- Does my game validate user input?
- Did I protect my data by keeping it private, except for public static final variables?
- Did I encapsulate my data and functionality and include them in the correct classes?
- Did I follow Design By Contract?
- Did I provide contracts for my methods?
- Did I provide correspondences to tie my private data representation to my interface?
- Did I follow programming to the Interface?
- Does my Game board still print correctly with changing board sizes?
- Did I comment my code?
- Did I avoid using magic numbers?
- Did I use good variable names?
- Did I follow best practices?
- Did I remove “Dead Code.” Dead Code is old code that is cancelled out.
- Did I make any additional helper functions I created private?
- Did I use the static keyword correctly?
- Did I add in new user stories that capture my new functionality?
- Did I update my activity diagrams to ensure any secondary methods don’t refer to private data?
- Did I create a class diagram for my new interface?
- Did I update my old class diagram to show that it inherits methods from the interface it implements
- Did I provide a working make file?
- Does my code compile and run on Unix?
- Is my program written in Java?