Assignment 4 – Connect Four

1. The program

This time you will be making the game Connect Four. You will be utilizing the **2-Dimensional array** to help you store the Connect Four grid information. You will still apply Classes and Methods in this, but it will be much smaller scale. This version of the Connect Four is slightly different as it will keep track of each player's score. Make sure you do apply all the knowledge you have learnt to write an efficient and modular program.

2. Basic Rules

The link will explain the rules.
https://en.wikipedia.org/wiki/Connect_Four

Play online.
https://www.mathsisfun.com/games/connect4.html

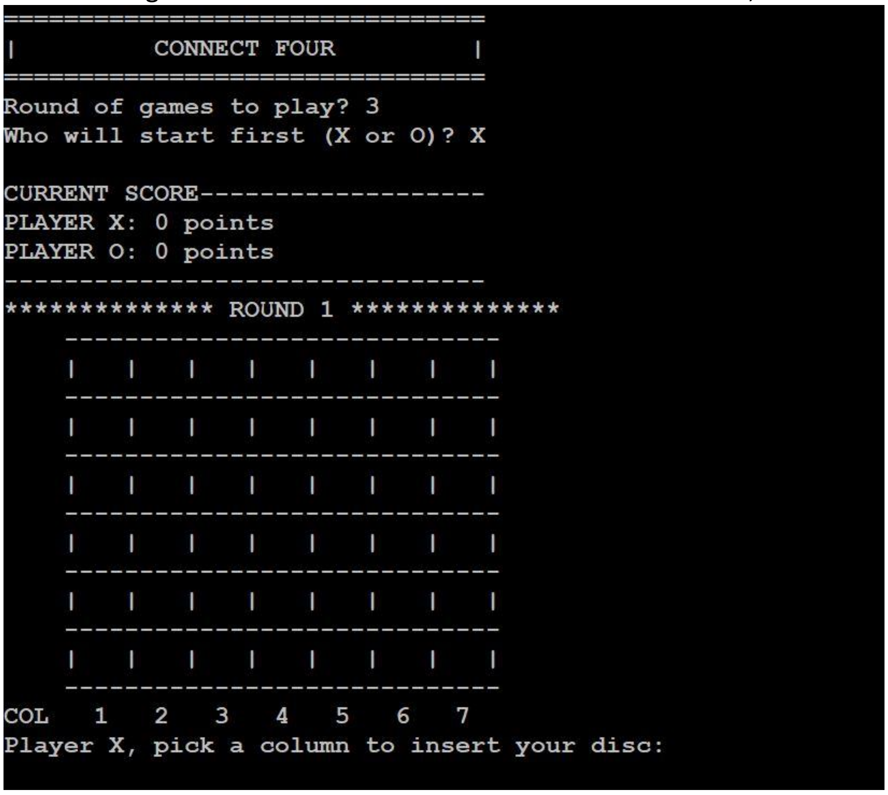
Video showing how it works
https://www.youtube.com/watch?v=utXzIFEVPjA

3. Extra Rules

- The game lets player decide how many round of games to play.
- Each player has a score.
- When a player wins in a single round, he/she will received points are base on the number of empty blocks remaining on the grid. This means if a player can win quickly, his/she will get more points. Example, if a player X wins and there's 15 empty blocks left on the grid, then player X will be awarded 15 points.
- After all the rounds, the player with the most points will win.

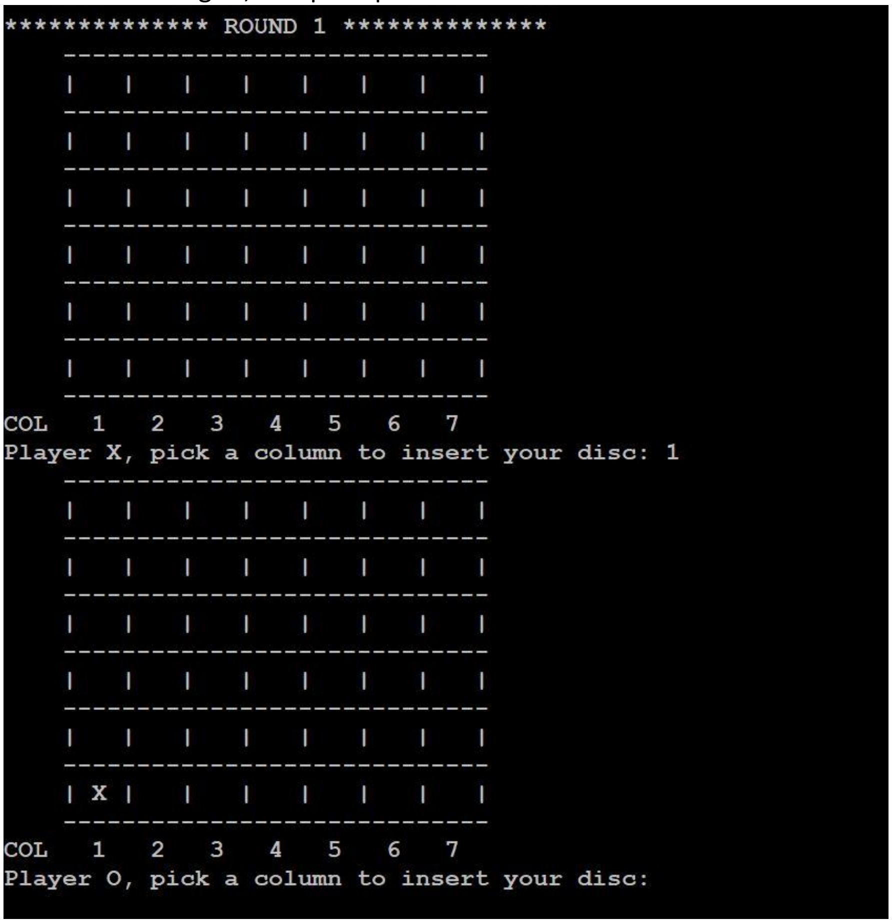
4. User Interface

a) Start of the game. Allow users to decide the number of rounds, and who will go first.



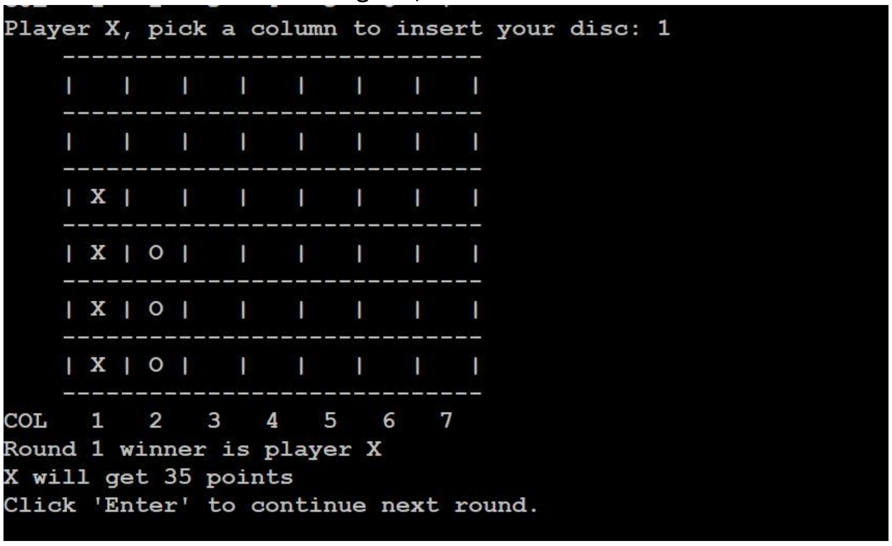
```
================================
|          CONNECT FOUR        |
================================
Round of games to play? 3
Who will start first (X or O)? X

CURRENT SCORE-------------------
PLAYER X: 0 points
PLAYER O: 0 points
--------------------------------
************** ROUND 1 **************
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
COL   1   2   3   4   5   6   7
Player X, pick a column to insert your disc:
```

b) Then shows the grid, and prompt for which column to insert a disc



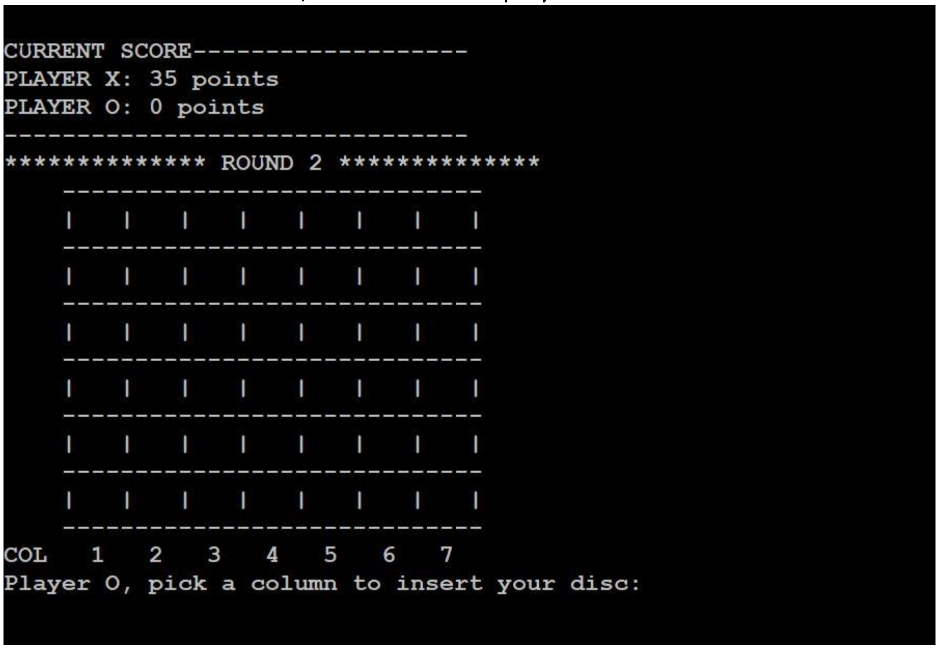
```
************** ROUND 1 **************
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
COL  1   2   3   4   5   6   7
Player X, pick a column to insert your disc: 1
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    | X |   |   |   |   |   |   |
    -----------------------------
COL  1   2   3   4   5   6   7
Player O, pick a column to insert your disc:
```

c) When 4 of the same disc is aligned, there’s a winner. It will also calculate the points.



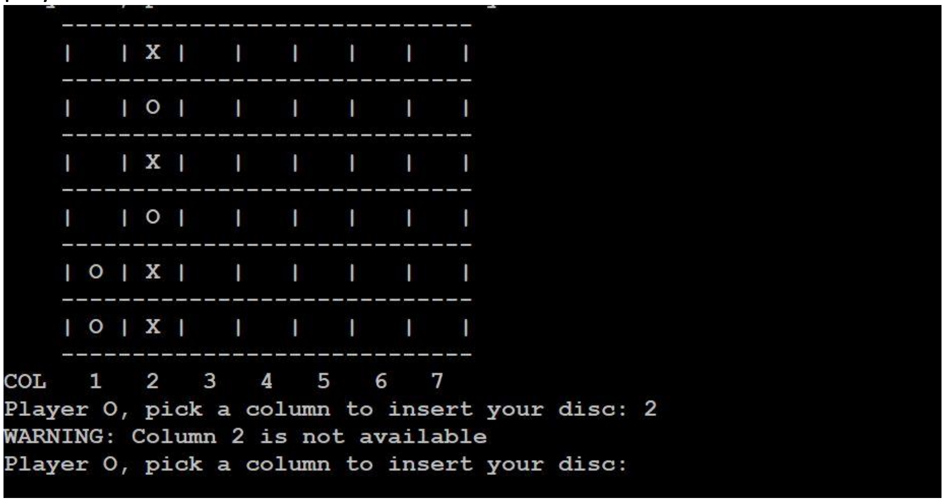
```
Player X, pick a column to insert your disc: 1
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    | X |   |   |   |   |   |   |
    -----------------------------
    | X | O |   |   |   |   |   |
    -----------------------------
    | X | O |   |   |   |   |   |
    -----------------------------
    | X | O |   |   |   |   |   |
    -----------------------------
COL  1   2   3   4   5   6   7
Round 1 winner is player X
X will get 35 points
Click 'Enter' to continue next round.
```

d) When a round is finished, it will show the player's scores and the round number (in this case round 2).



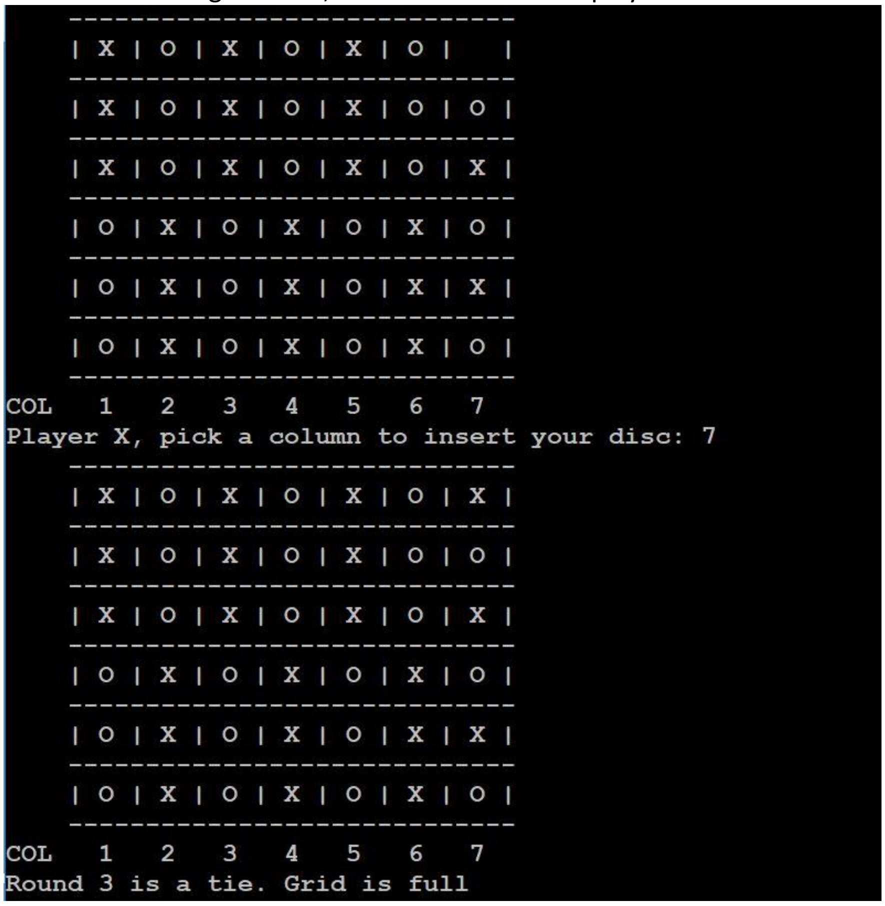
```
CURRENT SCORE--------------------
PLAYER X: 35 points
PLAYER O: 0 points
--------------------------------
************** ROUND 2 **************
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
    |   |   |   |   |   |   |   |
    -----------------------------
COL   1   2   3   4   5   6   7
Player O, pick a column to insert your disc:
```

e) When a column is full, it will warn the player and ask them to enter another column. Do not skip this player's turn.



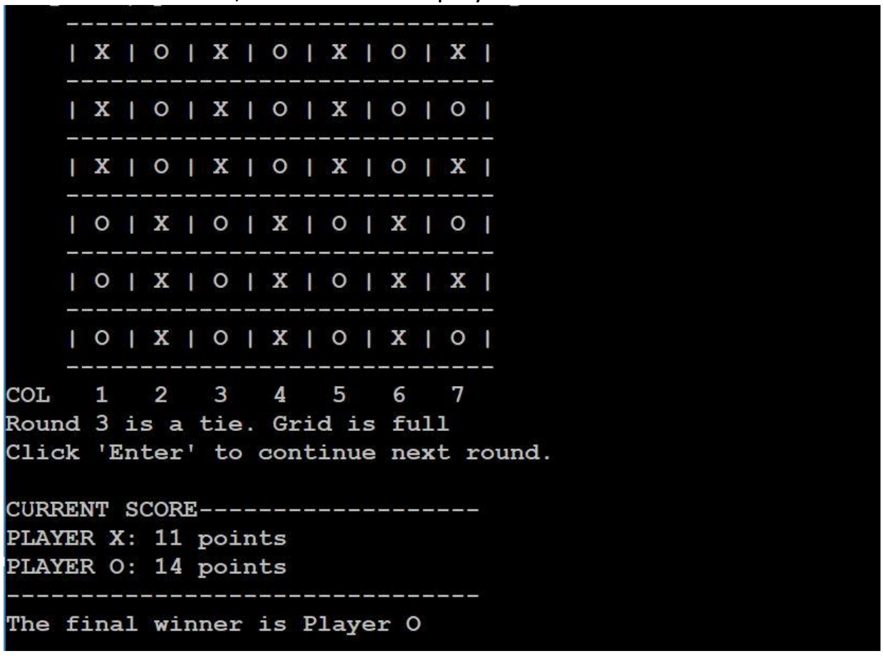
```
    -----------------------------
    |   | X |   |   |   |   |   |
    -----------------------------
    |   | O |   |   |   |   |   |
    -----------------------------
    |   | X |   |   |   |   |   |
    -----------------------------
    |   | O |   |   |   |   |   |
    -----------------------------
    | O | X |   |   |   |   |   |
    -----------------------------
    | O | X |   |   |   |   |   |
    -----------------------------
COL   1   2   3   4   5   6   7
Player O, pick a column to insert your disc: 2
WARNING: Column 2 is not available
Player O, pick a column to insert your disc:
```

f) When the whole grid is full, it will also notice the players. The round is a tie/draw.



```
    -----------------------------
    | X | O | X | O | X | O |   |
    -----------------------------
    | X | O | X | O | X | O | O |
    -----------------------------
    | X | O | X | O | X | O | X |
    -----------------------------
    | O | X | O | X | O | X | O |
    -----------------------------
    | O | X | O | X | O | X | X |
    -----------------------------
    | O | X | O | X | O | X | O |
    -----------------------------
COL   1   2   3   4   5   6   7
Player X, pick a column to insert your disc: 7
    -----------------------------
    | X | O | X | O | X | O | X |
    -----------------------------
    | X | O | X | O | X | O | O |
    -----------------------------
    | X | O | X | O | X | O | X |
    -----------------------------
    | O | X | O | X | O | X | O |
    -----------------------------
    | O | X | O | X | O | X | X |
    -----------------------------
    | O | X | O | X | O | X | O |
    -----------------------------
COL   1   2   3   4   5   6   7
Round 3 is a tie. Grid is full
```

h) After all the rounds, it will show the player's score and determine who is the final winner.



```
    -----------------------------
    | X | O | X | O | X | O | X |
    -----------------------------
    | X | O | X | O | X | O | O |
    -----------------------------
    | X | O | X | O | X | O | X |
    -----------------------------
    | O | X | O | X | O | X | O |
    -----------------------------
    | O | X | O | X | O | X | X |
    -----------------------------
    | O | X | O | X | O | X | O |
    -----------------------------
COL   1   2   3   4   5   6   7
Round 3 is a tie. Grid is full
Click 'Enter' to continue next round.

CURRENT SCORE-------------------
PLAYER X: 11 points
PLAYER O: 14 points
--------------------------------
The final winner is Player O
```

5. Things to check

- Check if the starting player is valid.
- Check if the user entered an invalid column number.
- Check if each column is full. If it is full, it will not allow the player to insert a disc there.
- Check if the grid is full, meaning no empty slot to insert a disc. Then it is a draw.

6. Requirements

- Program must follow the UML. Make sure you implement your program by using methods and classes.
- Each class must be in its separate file to improve readability and maintainability.
- Do not create extra classes, public methods or instance variables. However you may create Helper methods, and Global Constant variables.
- EXCEPTION: You may create more public method inside the Displayer class.
- Do not use anything that's not taught in class. (*Eg: ArrayList, LinkedList, 2D-Array, etc.*)
- Must complete all the methods in the template, and use them throughout your program.
- **IMPORTANT 1**: If a specific position doesn't have any disc, you must not insert a "fake" disc object to represent empty block. Should not waste memory and create something that's not useful. So must keep an empty position as **`null`**.
- **IMPORTANT 2**: You must not hardcode the checking of 4 connected pieces as you will automatically receive zero. Can't use actual value in the position of the array. Must use variables.

Hardcoding Example

```
int[] table = new int[3];
if( table[0] == 1 && table[0] == table[1] && table[1] == table[2] )
{
}
```

Good Example

```
int[] table = new int[3];
int count = 0;
for( int i = 0; i < table.length; i++ ) {
   if( table[i] == 1 ) {
      count++;
   }
}
```

7. Provided Templates and Files to Submit

Files to submit:	Files to submit:
● **`ConnectFourMain.java`** ● **`ConnectFour.java`** ● **`Disc.java`** ● **`Displayer.java`** ● **`Player.java`**	● **`ConnectFourMain.java`** ● **`ConnectFour.java`** ● **`Disc.java`** ● **`Displayer.java`** ● **`Player.java`**

Note:

- My whole program has 580 lines of code (Including comments and spaces).
- Aim for the following number of lines in your program. My sample program has:
  - There are **580 lines** of code in all of my files. (Including comments and empty lines)
  - There are **192 lines** of code in the templates.
  - So you only have to write **388 lines** of code
  - Do not stress yourself with the # of lines of code. It's just to let you know how big the program has to be.

- Working with friends is highly encourage, and may share ideas. But no sharing of code. There are programs that can check the program's similarity, and I can see who are copying programs.

Rubric (Checklist) for Knowledge

	Mark
KNOWLEDGE	
☐ Program will loop correctly ☐ Program will be able to exit ☐ Program will be able to print the game grid nicely ☐ Check invalid inputs	/4

Rubric for Application, Thinking and Communication

Items	Level 1	Level 2	Level 3	Level 4	Mark
APPLICATION					
Inserting disc onto the grid	Minimal functions work as intended	Some functions work as intended	Most functions work as intended	All functions work as intended	
Able to switch players	Minimal functions work as intended	Some functions work as intended	Most functions work as intended	All functions work as intended	
Check Winning by Horizontal	Minimal functions work as intended	Some functions work as intended	Most functions work as intended	All functions work as intended	
Check Winning by Vertical	Minimal functions work as intended	Some functions work as intended	Most functions work as intended	All functions work as intended	
Check Winning by Diagonal	Minimal functions work as intended	Some functions work as intended	Most functions work as intended	All functions work as intended	
Keep Track of Player Score & Calculate Score	Minimal functions work as intended	Some functions work as intended	Most functions work as intended	All functions work as intended	
Check Grid is Full	Minimal functions work as intended	Some functions work as intended	Most functions work as intended	All functions work as intended	
Check Final Winner	Minimal functions work as intended	Some functions work as intended	Most functions work as intended	All functions work as intended	
Error Handling.	Have error checking is in minimal places and displays little appropriate message	Have error checking is in some places and displays some appropriate message	Have error checking is in most places and displays mostly appropriate message	Have error checking is in all places and displays all appropriate message	/36
THINKING					
Use Displayer object to handle most of display	Minimal display functions are handled in Displayer	Some display functions are handled in Displayer	Most display functions are handled in Displayer	All display functions are handled in Displayer	
Classes and Objects implementation	Minimal classes are relevant structures and little methods belong to their classes.	Some classes are relevant structures and some methods belong to their classes.	Most classes are relevant structures and most methods belong to their classes.	All classes are relevant structures and all methods belong to their classes.	
Used static, final, and final static correctly	Minimal variables are used correctly.	Some variables are used correctly.	Most variables are used correctly.	All variables are used correctly.	
Program modularization (Creating helper methods)	Minimum required methods are modularized and efficiently used	Some required methods are modularized and efficiently used	Most required methods are modularized and efficiently used	All required methods are modularized and efficiently used	
Grid Management	Managed grid inefficiently and ineffectively	Managed grid somewhat efficiently and effectively	Managed grid mostly efficiently and effectively	Managed grid efficiently and effectively	
Followed UML Diagram	Minimum structures follow the UML diagram	Some structures follow the UML diagram	Most structures follow the UML diagram	All structures follow the UML diagram	/24

COMMUNICATION					
Variables/Methods Naming	Minimal variable names are clear and easy to understand	Some variable names are clear and easy to understand	Most variable names are clear and easy to understand	All variable names are clear and easy to understand	
Use of comments	Minimal amount of comments were used	Some amount of comments were used	Acceptable amount of comments were used	Extensive amount of comments were used	
User Interface	Difficult to understand the program, and minimal user prompts	Somewhat able to understand the program, and some user prompts	Somewhat easy to understand the program, and good user prompts	Easy to understand the program, and excellent user prompts	
Code Indentation	Indentations are minimal and readability is low	Indentations are somewhat correct and readability is average	Indentations are mostly correct and readability is mostly high	Indentations are all correct and readability is high	/16
TOTAL					