## Project #3 [10%]– Putting it all together.

**CLR 1**: Analyze a problem and define a solution in the form of an algorithm.

**CLR 2**: Document an algorithm using flowcharts, pseudocode, and UML class diagrams.

**CLR 3**: Test algorithms and / or associated flowcharts, using a test plan.

**CLR 4**: Describe and manipulate data as represented in a computer using the mathematics operations.

**CLR 5**: Implement the solution of a given problem by writing the appropriate code in a high-level language (Java).

**CLR 6**: Describe the representation of data in Java manipulating the data using mathematics operations.

**CLR 7**: Install and use a Java Development Kit (JDK), documentation libraries, Integrated Development Environment (IDE) and version control software.

**CLR 8**: Create programmer-oriented test plans and memory maps using an integrated debugger to evaluate and document Java program execution.

**CLR 9**: Create a local Git repository using Git commands to perform basic version control tasks on a Java program.

### Objective of this Assignment:

The objective of this assignment is to take everything you have learned in this course and put it altogether. This project is purposely more vague than previous it’s also in general much harder. This is by design, do not be discouraged by this you can still receive extremely high marks even for an only semi functional program. As stated before this is your time to showcase all that you have learned this term.

### Pre-Assignment Instructions:

To complete this assignment successfully, you should:

* Read the module contents and follow learning activities.
* Read the textbook chapters assigned.
* Read, follow links, and watch videos in announcements related to Project 3.
* Understand all previous Assignments.

### Assignment Description:

In this assignment you will create an interactive Connect 4 game if you are unfamiliar here is the Wikipedia article https://en.wikipedia.org/wiki/Connect\_Four.

### Assignment Tasks:

1. Create a Java project for your Connect 4 game.
2. Initialize and use a local git repository for your Project. (HINT: recall Assignments 1 and 2)
3. Create an interactive Connect 4 game for two players. (HINT: a lot of the concepts used in Assignment 8 Tic-tac-toe would also apply to Connect 4)
4. Some additional hints:
   1. Plan first! It’s much easier to start when you have planned out how to approach the problem beforehand.
   2. Break the project down into smaller parts. How do you check if there is a winner? How does a player make a move? How do you store the board representation? How do you display the current game state? etc. By thinking about smaller and smaller pieces, what seems like an impossible task becomes much simpler.
   3. Git is your friend! I encourage you to make use of git!
   4. The debugger is your friend! Use it as often as possible to step through errors in your code.
   5. Something close is better than something perfect. If you are struggling to think of how to get it to entirely work, try and get something semi working and work from there. You may find it easier to make modifications to an existing codebase as opposed to attempting to make something perfect right from the start.

### Submission:

Submit your entire Project folder (including the .git folder which exists within it) likely easiest to zip the entire thing and attach that to Brightspace.

### Marking Rubric:

Take note of the following, it is possible to still receive a near perfect score even if your code is not perfect.

For this assignment you will be marked as follows:

* 2 points for code that compiles and runs.
* 1 point for code that produces any output at all.
* 1 point for using git in anyway at all.
* 2 points for code that at least prints out a connect 4 game.
* 4 points for completeness and interactivity of connect 4.

Of the 4 points for completeness and interactivity. The first 2 points will be given out more easily for large design choices and attempts and completion. The final 2 points will be for polish and correctness.

You may lose partial points on discretion for the following reasons:

* Code does not compile and run.
* Supplied code has been changed in any way.
* Code is not formatted nicely or clean.
* Code has clear to see simpler solutions.
* Code has bugs or missed edge cases.
* Over reliance on the main method, you should be breaking your logic up.