AP Computer Science Project I
Mr. Lew

"Never tell people how to do things. Tell them what you want them to achieve, and they will surprise you with their ingenuity."

- George Smith Patton, Jr.

This project will require you to design and implement a Java-based game or simulation. It will be completed in 4 phases. Each phase is described below:

Phase I - Proposal

Page 1:
Title of your project goes here (e.g. "Othello")

Project Summary – This section should be a description of your project. It should address things such as the operation of the game/simulation, rules, number of players, artificial intelligence, special features you want to implement, etc. How long should the summary be?

"Be simple as possible, but no simpler."
- Albert Einstein

Page 2:
UML Diagram – You will create a UML (Unified Modeling Language) diagram that shows each of the classes in your project along with its private data members and public and private (helper) methods. Arrows to superclasses in your inheritance hierarchy should be drawn as well. Use the Excel template to create an electronic form of this diagram in landscape orientation. You will edit the diagram as you develop your project and will present the final draft of it in your final presentation. Make sure to include:

a. Class Composition: "has-a" relationships
b. Inheritance: "is-a" relationships
c. Interface: "does-a" relationships

Page 3:
Screenshot(s) – Here you will sketch screenshot(s) for different views of your project. Your self-drawn screenshot (may be hand draw or electronically draw) should should not include downloaded picts or jpgs. Include multiple diagrams if your project will present the user with different screens. Include a brief description of each screenshot.

IMPORTANT: View this proposal as a way to thoughtfully plan out your project. The more thorough you are in the planning stages, the fewer roadblocks, programming issues, and bugs you will face when you actually begin coding. Starting to code from the ground up without a good plan in place is quintessential cowboy programming (i.e., not good).

Phase II – Java Program

The program must include the following items.

1. At least two relational (==, !=, >, >=, <, <=) and two logical operators (&&, | |, !)
2. At least two "if-then-else" statements.
3. At least one of **EACH** of the following: for loop, for-each loop, and while loop.
4. At least ONE **student-designed** interface and at least THREE **student-designed** classes (one of which MUST be abstract) must be included. In addition, your program shall have a driver class (e.g., "OthelloDriver.java", or subclass or the `World` class if using Greenfoot), bringing the total minimum number of classes to five.
5. At least one instance of Class Composition ("has-a" relationship) must be used in your project. Recall that "Class Composition" is using another class as an "instance variable" for your class. Recall `MyPhone` of `MySongs`, `MyPhone` "has-a" `MySong`.
6. Interaction between your student-designed classes/interfaces in your project must be implemented. Specifically, each class must call method(s) from another class in your project.
7. An Inheritance hierarchy ("is-a" relationship) must be implemented with the student-designed superclass (inheriting from the `World` and `Actor` classes in Greenfoot does not satisfy this requirement since `World` and `Actor` classes are not student designed). The inheritance hierarchy shall have at least three levels.
8. At least one student-designed interface must be implemented in the project.
9. Polymorphism must be implemented with the **student-designed** classes.
10. Class `ArrayList` must be used in at least ONE **student-designed** class and it MUST be traversed through AND accessed via a for loop OR a for-each loop.
11. Class `array` must be used in at least ONE **student-designed** class and it MUST be traversed through AND accessed via a for loop OR a for-each loop.
12. **Your project must demonstrate a high level of "algorithmic complexity." That is, your "brain" or "processing" methods should show a high degree of problem solving ability. See the grading Lewbric for the details on how algorithmic complexity will be assessed.**
13. Comments explaining logic and operation of program at "key points" (e.g. special algorithm to determine possible next moves in Chess, to check winners in Connect Four, to follow Pac-Man around the screen, etc.)
14. Meaningful and variable/class names throughout (class, methods, variable name, instance variables, etc.)
15. Including javadocs comments for each of your methods in at least one student-designed class. (see sample AP problems for the javadoc convention)

See the website for ideas on projects. Projects whose code has been released are not eligible project ideas unless SIGNIFICANT changes are proposed that will make it UNIQUE from the original.

A zip file containing the entire project will be due on the date shown on thecubscientist.com. This is typically the second or third day after you return to school in February.

THERE WILL BE A 15% deduction of points for each day that the project is late.

Phase III – Project Presentation Submission Guidelines

By 8am on the morning of your presentation, you should email your presentation IN PDF format to mlew@loyolahs.edu(no hardcopy printouts are necessary)

In addition, the **BODY** of the email should include a "Description", "How to play", and "Interesting Features" text that will be used to describe your project when it is posted to the website. See the example below.

Example:
Description:
My game is simple single player pong game with an AI controlled opponent and a scoreboard. It is similar to the original pong except for a few key features. The objective of the game is to get as many points as possible by getting the ball past the opposing player. The game ends when the total score has reached 9. The AI difficulty can be changed in order to change the difficulty level. The game starts with a simple window asking whether or not you would like to play; after that there is an eight second timer then the game begins.

How to play:
Click the "DOWN" button in order to get the paddle to move down, and click the "UP" button in order to get the ball to move up.

Interesting Features:
1. The AI's difficulty setting and the AI's algorithm are both set by only two numbers. The difficulty setting is a number between 60 - 99 and the higher the number the harder it is to defeat the AI. The second number is a random number between 1 - 100, should the difficulty setting be higher than this number than the AI is allowed to move towards the paddle, but if the random number be higher than the difficulty setting, the paddle remains in place.
2. Constant motion of the paddle is another key feature of my game. The paddle is designed to be in almost constant motion in order to add another level of difficulty to the game. The paddle only stops at the bottom and top of the screen. This prevents the paddle from leaving the game entirely.

Phase IV – Presentation

You will give your presentation on your specified lottery date. The presentation shall have, at minimum, the following slides. You will have approximately 35 minutes for your presentation.

a. Title
b. Description of program operation (or how game is played)
c. Demonstration of Program
d. UML Diagrams for each class
e. Use of classes/objects in project - elaborate on how classes represent **physical objects** in your program **(be prepared to justify class names, class data member names, class method names…)**
f. Description of class interaction **(be prepared to discuss how each class interacts with the other classes)**
g. Description of use of an inheritance hierarchy **(be prepared to justify structure)**
h. Description of use of an interface **(be prepared to justify its use with other classes)**
i. Description of use of polymorphism **(include a code snippet that demonstrates polymorphism)**
j. Special features implemented in program - elaborate on tricks/special things you did
k. Known bugs in program
l. Citation of "second-party" code used in program **(be able to explain code; not including Greenfoot GUI)**
m. Conclusion - Summary of what you thought of writing the program
    i. Difficulty level,
    ii. "Fun" level,
    iii. Your evaluation of the final product,
    iv. What you learned (be specific)
n. Questions? (this is simply a slide that says "Questions?" that keys the audience for any questions they might have)

Pointers for your final project:

1. **START EARLY!! Allow yourself enough time to plan your code, write it and debug it!**
2. **Test your program on the presentation computer before the presentation date.**
3. **Using System.out.println() statements to display values during the debugging process.**
4. **MAKE BACKUPS OF YOUR WORK!!**
5. **And, of course,…HAVE FUN :-) !!**

"The first 90% of code is responsible for the first 90% of debugging,
the last 10% of code is responsible for the other 90% of debugging"
- Unknown