Assignment 3

Due date: as indicated on Moodle

You may work alone or in a group of 2.

Objectives:

- Create a fully functioning Java application
- Get familiar with the IDE (IntelliJ **Community**)
- Create classes
- Use inheritance
- Apply polymorphism

Instructions:

1. This is going to be an on going project that we will keep adding functionality to it. We will keep expanding it as we progress.

2. Create a GeometricShape class. A geometric shape has an outline color attribute that is defined as a string.

3. Every GeometricShape has an area. That is why we need to define the **abstract** method calcArea that returns the area of the GeometricShape as a double value. Should calcArea take any parameters? Think about it.

4. As always, the GeometricShape MUST override the toString method.

5. Develop a Circle concrete class. What are you going to do with the method calcArea that you inherit from GeometricShape?

6. A circle is defined by its radius which is a double value.

7. Develop a calcPerimeter method in the Circle class. Should it have any parameters?

8. Develop a Rectangle class. A rectangle is defined by its height and width. Both are double values.

9. You must override the equals method in all classes. The system should allow me to compare two objects from the created ones and display the result on the screen.

10. A square is a special rectangle where the width is equal to the length. Create a square class and add it to the system.

11. Develop a driver class that displays the following menu. The program should keep running until the user chooses option 4.
    1. Create a circle
    2. Create a rectangle
    3. Create a square
    4. Compare two objects
    5. Display all objects in the system (MUST use the toString method of the different classes)
    6. Display all object and their corresponding areas.
    7. Exit

12. Use one array to store all the GeometricShapes in it. Polymoric calls are expected when you loop on objects.

What to submit:

The whole project Zipped + UML class diagram as a PDF in the zip

A demo must be given to the instructor. 0 credit if no demo is given to the instructor. You will be notified of the demo times.

Grading Rubric

Criteria	**Marks**
Internal documentation	1
Code quality (Meaningful names, indentation, spacing, etc..)	1
Polymorphic calls	3
Overriding the equals method	2
Class design (UML class diagram)	1
Functionality	2
Total	**10**

Have fun ☺