Department of Computer Science
The City College of CUNY

CSc 22100: Software Design Laboratory [Fall 2020]

Assignment 2

A report uploaded on the Blackboard's course page for the section showing [1] *the problem,* [2] *solution methods,* [3] *codes developed, and* [4] *outputs produced for the assignment indicated is due by* 2:00 *pm on Tuesday,* 20 October 2020. ***The deadline is strictly observed***.

1- Amend the hierarchy of Java classes in Assignment 1 as follows:

MyLine *is_a* **MyShape**;
MyRectangle *is_a* **MyShape**
MyOval *is_a* **MyShape**;
MyCircle *is_a* **MyOval;**
MyPolygon *is_a* **MyShape**.

2- Class **MyShape** is an **abstract** class; is the hierarchy's superclass; and inherits Java class Object. The *draw* method in class **MyShape** is an *abstract* method and hence must be overridden in each subclass in the hierarchy. Further, the reference point of **MyShape**, **p**(*x*, *y*), is an object of class **MyPoint**. Otherwise, the classes **MyShape**, **MyLine**, **MyPolygon**, and **MyCircle** are defined as in Assignment 1, but now utilize:

Class MyPoint:

Class **MyPoint** is used by class **MyShape** to define the reference point **p**(*x*, *y*) of the Java display coordinate system. The class includes appropriate class constructors and methods, including methods that perform point related operations.

Class MyRectangle:

Class **MyRectangle** inherits class **MyShape**. The **MyRectangle** object is a rectangle of height *h* and width *w*, and a top left corner point **p**(*x*, *y*), and may be filled with a color. The class includes appropriate class constructors and methods, including methods that perform the following operations:

a. *getWidth, getHeight, getPerimeter, getArea*— return the width, height, perimeter, and area of the **MyRectangle** object;
b. *setWidth, setHeight* — set the width and height of the **MyRectangle** object;
c. *toString*— returns a string representation of the **MyRectangle** object: top left corner point, width, height, perimeter, and area;

d. *draw*— draws a **MyRectangle** object of height *h* and width *w*, and anchored at **p**(*x*, *y*).

Class MyOval:

Class **MyOval** inherits class **MyShape** and like other subclasses in hierarchy may use class **MyRectangle**. The **MyOval** object is defined by an ellipse inscribed in a rectangle of height *h* and width *w*, and a top left corner point **p**(*x*, *y*). The **MyOval** object may be filled with a color. The class includes appropriate class constructors and methods, including methods that perform the following operations:

a. *getPerimeter, get Area, getCenter*— returns the perimeter, area, and center point of the **MyOval** object;
b. *setAxes, setCenter*— sets the axes lengths and center point of the **MyOval** object;
c. *toString*— returns a string representation of the **MyOval** object: axes lengths, perimeter, and area;
d. *draw*— draws a **MyOval** object inscribed in a rectangle of height *h* and width *w*, and anchored at **p**(*x*, *y*).

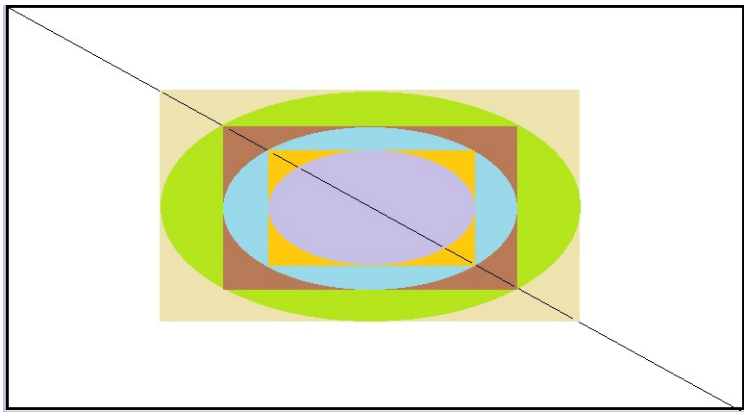
3- Interface **MyShapeInterface** is implemented by class **MyShape**. All classes of the hierarchy must be amended in accordance with the interface. The interface includes appropriate constants and abstract, static, and/or default methods that describe the functions and behaviors of the specific object types of the class hierarchy, including:

a. *getMyBoundingRectangle*— returns the bounding rectangle of an object in the class hierarchy;
b. *getMyArea*— returns the area of an object in the class hierarchy— i.e., the set of all points on and within the boundary of the object;
c. *overlapMyShapes*— returns the intersecting area of two objects in the class hierarchy if they do overlap; and **null** otherwise.

4- Use JavaFX graphics and the class hierarchy to draw a geometric configuration comprised of a sequence of alternating concentric ovals and their inscribed rectangles as illustrated below, subject to the following additional requirements:

a. The code is applicable to canvases of variable height and width;
b. The dimensions of the shapes are proportional to the smallest dimension of the canvas;
c. The hexagons and circles are filled with different colors of your choice, specified through a **MyColor** enum reference type.

5- Explicitly specify all the classes imported and used in your Java code.



Best wishes

Hesham A. Auda
8 October 2020