Department of Computer Science
The City College of CUNY

CSc 22100 [P 34721]: Software Design Laboratory [Fall 2021]

Assignment 1

A report uploaded on the Bloackboard's course page for the section showing:

[1] *the problem,*
[2] *solution methods,*
[3] *codes developed, and*
[4] *outputs produced for the tasks indicated*

is due by 11:00 pm on Tuesday, 5 October 2021. ***The deadline is strictly observed***.

1- Create a hierarchy of Java classes as follows:

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

Class MyPoint:

Class **MyPoint** is used by class **MyShape** to define the reference point **p**(x, y) of the Java display coordinate system, as well as by all subclasses in the class hierarchy to define the points stipulated in the class definition. The class utilizes a color of enum reference type **MyColor**, and includes appropriate class constructors and methods, including methods that perform point related operations.

Class MyShape:

Class **MyShape** is the hierarchy's superclass and extends the Java class Object. An implementation of the class defines a reference point **p**(x, y), an object of type **MyPoint**, and the color of the shape of enum reference type **MyColor**. The class includes appropriate class constructors and methods, including methods, including methods that perform the following operations:

a. *area, perimeter* – return the area and perimeter of the object. These methods must be overridden in each subclass in the hierarchy. For the **MyShape** object, the methods return zero.
b. *toString* – returns the object's description as a String. This method must be overridden in each subclass in the hierarchy;
c. *draw* – draws the object shape. This method must be overridden in each subclass in the hierarchy. For the **MyShape** object, it paints the drawing canvas in the color specified.

Class MyLine:

Class **MyLine** extends class MyShape. The **MyLine** object is a straight line segment defined by the endpoints $\mathbf{p}_1(x_1, y_1)$ and $\mathbf{p}_2(x_2, y_2)$. The **MyLine** object may be of any color of enum reference type **MyColor**. The class includes appropriate class constructors and methods, including methods that perform the following operations:

a. *length* — returns the length of the **MyLine** object;
b. *xAngle*— returns the angle (in degrees) of the **MyLine** object with the x-axis;
c. *toString* — returns a string representation of the **MyLine** object, including the line's endpoints, length, and angle with the x-axis;
d. *draw* — draws a **MyLine** object.

Class MyRectangle:

Class **MyRectangle** extends 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 of enum reference type **MyColor**. The class includes appropriate class constructors and methods, including methods that perform the following operations:

e. *getX, getY, getWidth, getHeight* — return the width, height of the **MyRectangle** object
f. *toString*— returns a string representation of the **MyRectangle** object: top left corner point, width, height, perimeter, and area;
g. *draw*— draws a **MyRectangle** object.

Class MyOval:

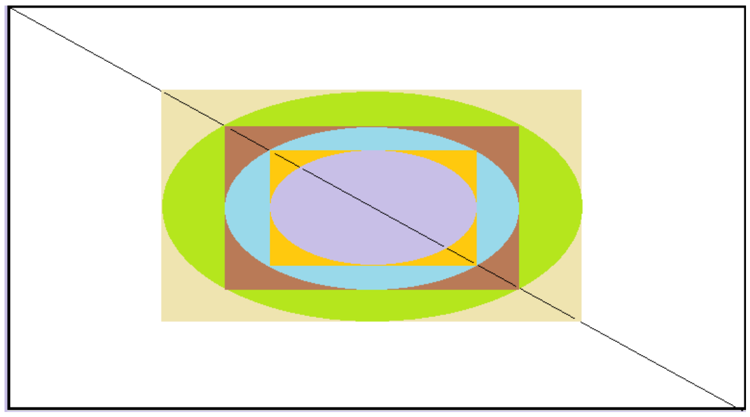
Class **MyOval** extends class **MyShape**. The **MyOval** object is defined by an ellipse within a rectangle of height *h* and width *w*, and a center point **p**(*x*, *y*). The **MyOval** object may be filled with a color of enum reference type **MyColor**. The class includes appropriate class constructors and methods, including methods that perform the following operations:

a. *getX, getY, getA, getB* — return the x- and y-coordinates of the center point and abscissa of the **MyOval** object;
b. *toString*— returns a string representation of the **MyOval** object: axes lengths, perimeter, and area;
c. *draw*— draws a **MyOval** object.

2- Use JavaFX graphics and the class hierarchy to draw the geometric configuration comprised of a sequence of alternating concentric ovals and their inscribed rectangles shown 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 ovals and rectangles are filled with different colors of your choice, specified through an enum reference type **MyColor**.

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



Best wishes

Hesham A. Auda
09-22-2021