Department of Computer Science
The City College of CUNY

CSc 22100 [P 51666]: Software Design Laboratory [Fall 2020]

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 assignment indicated is due by* 2:00 *pm on Thursday*, 1 October 2020. ***The deadline is strictly observed***.

1- Create a hierarchy of Java classes as follows:

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

Class MyShape:

Class MyShapeis the hierarchy's superclass and inherits the Java class Object. An implementation of the class defines a reference point (*x*, *y*) and the color of the shape. The class includes appropriate class constructors and methods, including methods that perform the following operations:

a. *getX, getY, getColor* – returns the x- and y-coordinates of the reference point and color of the MyShape object;
b. *setX, setY, setColor*– sets the x- and y-coordinates of the reference point and color for the MyShape object;
c. *toString*– returns the object's description as a String. This method must be overridden in each subclass in the hierarchy;
d. *draw*– draws a MyShape object. 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 inherits class MyShape. The MyLine object is a straight line defined by the endpoints (x_1, y_1) and (x_2, y_2). The MyLine object may be of any color. The class includes appropriate class constructors and methods that perform the following operations:

a. *getLength* — returns the length of the MyLine object;
b. *get_xAngle*— return 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 whose end points are (x_1, y_1) and (x_2, y_2).

Class MyPolygon:

Class MyPolygon inherits class MyShape. The MyPolygon object is a *regular* polygon defined by the integer parameter, *N*— the number of the polygon's equal side lengths and equal interior angles, and the center (x, y) and radius, r, of the circle in which it is inscribed. The MyPolygon object may be filled with a color. The class includes appropriate class constructors and methods that perform the following operations:

e. *getArea*— returns the area of the MyPolygon object;
f. *getPerimeter*— returns the perimeter of the MyPolygon object;
g. *getAngle*— returns the interior angle (in degrees) of the MyPolygon object;
h. *getSide*— returns the side length of the MyPolygon object;
i. *toString*— returns a string representation of the MyPolygon object, including the number of sides, side length, interior angle, perimeter, and area;
j. *draw*— draws a MyPolygon object inscribed in a circle centered at (x, y) and of radius r.

Class MyCircle:

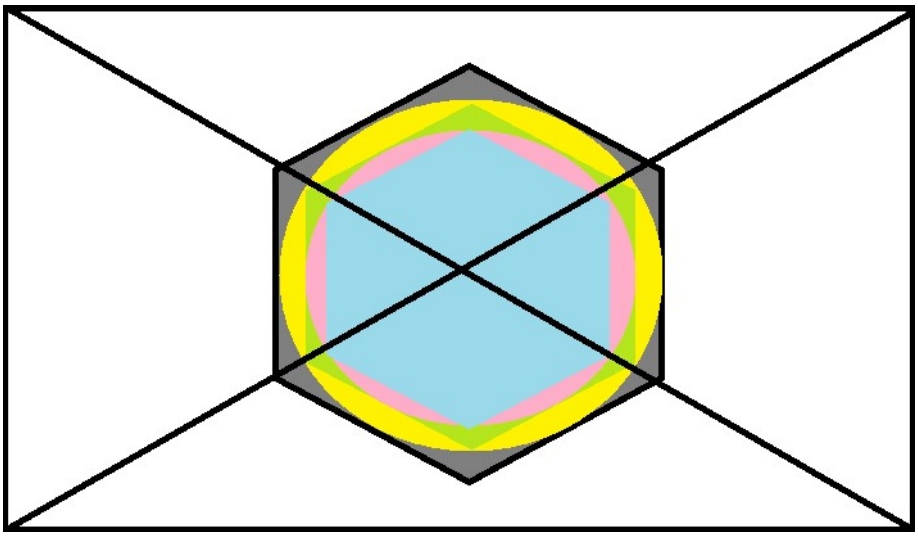
Class MyCircle inherits class MyShape. The MyCircle object is defined by its center (x, y) and radius r, and may be filled with a color. The MyCircle class includes appropriate class constructors and methods that perform the following operations:

a. *getArea*— returns the area of the MyCircle object;
b. *getPerimeter*— returns the perimeter of the MyCircle object;
c. *getRadius*— returns the radius of the MyCircle object;
d. *toString*— returns a string representation of the MyCircle object, including its center point, radius, perimeter, and area;
e. *draw*— draws a MyCircle object centered at (x, y) and of radius r.

2- Use JavaFX graphics and the class hierarchy to draw a geometric configuration comprised of a sequence of alternating concentric circles and their inscribed hexagons 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.

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



Best wishes

Hesham A. Auda
9-17-2020