

Exercise 1

A printout showing the problem, solution method, codes developed, and outputs produced for the tests indicated is due during and before the end of the class on Monday, 1 October 2018. 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 MyShape is the hierarchy's superclass and inherits Java class Object. An implementation of the class defines a point (x, y) and the color of the shape. The class includes appropriate class constructors and methods that perform the following operations:

- a. *getX, getY, getColor* – returns the point (x, y) and color of the MyShape object;
- b. *setX, setY, setColor* – sets the point (x, y) and color for the MyShape object;
- c. *shiftXY* – moves point (x, y) by $(\Delta x, \Delta y)$;
- d. *toString* – returns the object's description as a String. This method must be overridden in each subclass in the hierarchy.
- e. *draw* – This method must be overridden in each subclass in the hierarchy. For the MyShape object, it paints the drawing canvas in color.

Class MyLine:

Class MyLine inherits class Shape. The MyLine object is a straight line defined by its two 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: length and angle with the x-axis;
- d. *draw* – draws a MyLine object whose end points are (x, y) (x_1, y_1) and (x_2, y_2) .

Class MyPolygon:

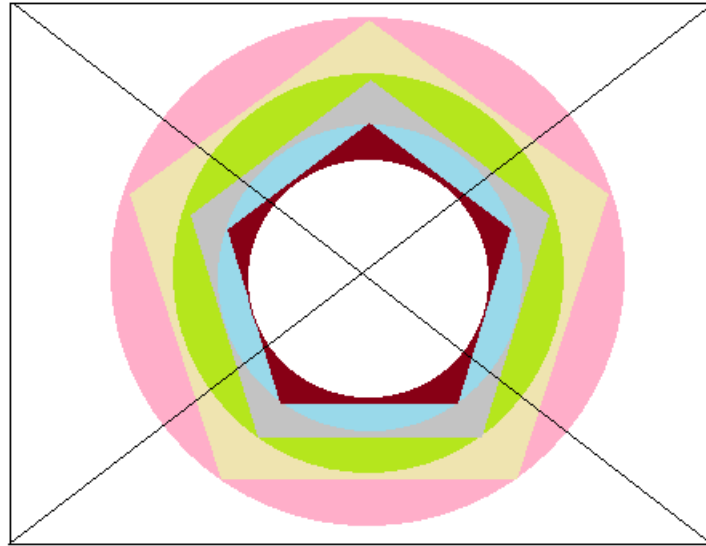
Class `MyPolygon` inherits class `MyShape`. The `MyPolygon` object is a *regular* polygon defined by the integer parameter N — the number of the polygon's equal side lengths and equal interior angles. The `MyPolygon` object may be filled with a color. The class includes appropriate class constructors 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` — return 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: side length, interior angle, perimeter, and area;
- j.* `draw` — draws a `MyPolygon` object whose center point (x, y) is defined in class `MyShape` and inscribed in a circle of radius *radius*.

Class `MyCircle`:

Class `MyCircle` inherits class `Shape`. The `MyCircle` object is defined by its radius, *radius*, and center (x, y) , 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: radius, perimeter, and area;
 - e.* `draw` — draws a `MyCircle` object of radius *radius*. The center point (x, y) of the circle is defined in class `MyCircle Shape`.
- 2- Use JavaFX graphics and the class hierarchy to draw a geometric configuration comprised of a sequence of alternating concentric pentagons and circles 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 pentagons and circles are filled with different colors of your choice.



Hesham A Auda
17 Sptember 2018