LAB EXERCISE

RectangleMethods

Background:

1. In this lab exercise, you will continue your work on the `Rectangle` class created in *L.A. 6.2 Rectangle* by adding additional attributes and behaviors. These enhancements will include the ability to construct a `Rectangle` from and existing one, get and set the x coordinate, y coordinate, width and height parameter, set the orientation in which the rectangle will be drawn, and display textual information on the drawing surface.

2. The specifications of a class that models a rectangular shape would be:

   Variables

```
private double myX;      // the x coordinate of the rectangle
private double myY;      // the y coordinate of the rectangle
private double myWidth;  // the width of the rectangle
private double myHeight; // the height of the rectangle

// saves the direction the pen is pointing
// (0 = horizontal, pointing right)
private double myDirection;

// Creates a 500 x 500 SketchPad with a DrawingTool, pen, that is used
// to display Rectangle objects. The Drawingtool is declared static
// so that multiple Rectangle objects can be drawn on the Sketchpad
// at the same time.
private static DrawingTool pen =
   new DrawingTool(new SketchPad(500, 500));
```

   Constructors

```
// Creates a default instance of a Rectangle object with all dimensions
// set to zero.
Rectangle()

// Creates a new instance of a Rectangle object with the left and right
// edges of the rectangle at x and x + width. The top and bottom edges
// are at y and y + height.
Rectangle(double x, double y, double width, double height)

// Creates a new instance of a Rectangle object that is a copy of an
// existing Rectangle object.
Rectangle(Rectangle rect)
```

   Methods

```
// Sets the x coordinate of this Rectangle
public void setXPos(double x)

// Sets the y coordinate of this Rectangle
public void setYPos(double Y)
```

```
// Sets the width this Rectangle
public void setWidth(double width)

// Sets the height of this Rectangle
public void setHeight(double height)

// Sets the direction the DrawingTool is pointing
// 0 = horizontal to the right
public void setDirection(double direction)


// Returns the x coordinate of this Rectangle
public double getXPos()

// Returns the y coordinate of this Rectangle
public double getYPos()

// Returns the width of this Rectangle
public double getWidth()

// Returns the height of this Rectangle
public double getHeight()

// Returns the direction the DrawingTool is pointing
// 0 = horizontal to the right
public double getDirection()

// calculates and returns the perimeter of the rectangle
public double getPerimeter()

// Calculates and returns the area of the rectangle.
public double getArea()


// Draws String str at the specified x and y coordinates
public void drawString(String str, double x, double y)

// Draws a new instance of a Rectangle object with the left and right
// edges of the rectangle at x and x + width. The top and bottom edges
// are at y and y + height.
public void draw()
```

Assignment:

1. Implement a `Rectangle` class with the following properties.

   a. A default `Rectangle` object is specified in the constructor with the `x`, `y`, `width` and `height` set to 0.

   b. A `Rectangle` object is specified in the constructor with the left and right edges of the rectangle at `x` and `x + width`. The top and bottom edges are at `y` and `y + height`.

   c. A `Rectangle` object is specified that is a copy of an existing `Rectangle`.

   d. Methods `getXPos`, `getYPos`, `getWidth`, and `getHeight`, return the `x`, `y`, `height` and `width` of the `Rectangle` respectively.

e. A method, `getDirection`, returns the current orientation of the `DrawingTool`.

f. Methods `setXPos`, `setYPos`, `setWidth`, and `setHeight`, sets the `x`, `y`, `height` and `width` of the `Rectangle` respectively to the value of each methods **double** parameter.

g. A method `setDirection`, sets the current orientation of the `DrawingTool`.

h. A method `getPerimeter` calculates and returns the perimeter of the `Rectangle`.

i. A method `getArea` calculates and returns the area of the `Rectangle`.

j. A method `draw` displays a new instance of a `Rectangle` object.

k. A method `drawString` displays String at the specified x and y coordinates of the drawing area.

2. The methods `draw`, `drawString`, and `setDirection` make use of existing `DrawingTool` methods. Refer to handout, *H.A.1.1 – DrawingTool*, for details on the `DrawingTool` methods.

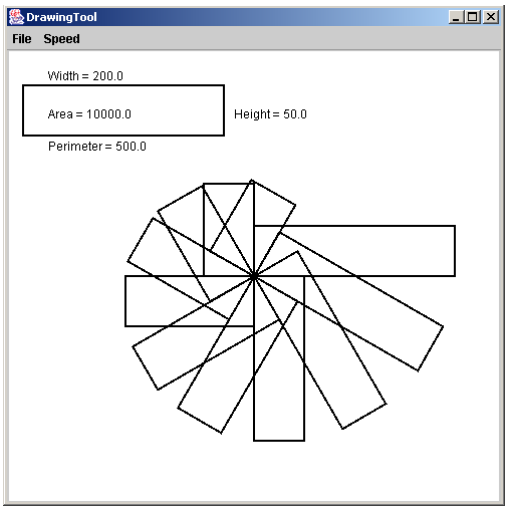
3. Write a testing class with a `main` method that constructs a `Rectangle`, `rectA`, and calls `setDirection`, `setWidth`, and `draw` for each `Rectangle` created. It is recommended that the changes in orientation and width of each successive rectangle in the spiral be calculated using the `getDirection` and `getWidth` methods. For example, if the increment for each turn is given by `turnInc` and the decrease in size of rectangle is given by `widthDec`, then successive calls to the following:

```
rectA.setDirection(rectA.getDirection() - turnInc);
rectA.setWidth(rectA.getWidth() - widthDec);
rectA.draw();
```

would draw each "spoke" of the rectangular spiral:

Construct another `Rectangle`, `rectB` that is a copy of the original `rectA`. Draw the rectangle in the upper left corner of drawing area. Label the rectangle with its width, height, perimeter and area.

The resulting image would be similar to the one shown below:



4. It is recommended that the `Rectangle` class and the testing class be in separate files. Call your instructor to your workstation for scoring