Lab 20
Bunch of Lines

Objective:

Write a bunch of classes that will draw a bunch of lines! Don't worry you don't have to write any graphics, as that part is provided in the driver. Each line is drawn based on a math function that takes in a given x coordinate and will return its y coordinate.

- First download the driver and put it in your project
  - DO NOT ALTER THE DRIVER!

Write an interface called **Line**

- Create the following method definition
  - getYPoint: This takes in a decimal value and returns a decimal value depending on the type of line.

Write a class called **SlopedLine**

- This should implement Line
- Instance variable
  - slope: a decimal value corresponding to the line's slope
- Create the following Constructors
  - Default
  - Parameterized Constructor
- Accessors and Mutators for each variable
- Create the following Methods
  - getYPoint – this method takes in a decimal value corresponding to a x-coordinate and returns the y-coordinate based on the slope equation (y = slope*x)

Write a class called **ExponentialLine**

- This should implement Line
- Instance variable
  - exponent: a decimal value corresponding to the line's exponent
- Create the following Constructors
  - Default
  - Parameterized Constructor
- Accessors and Mutators for each variable
- Create the following Methods
  - getYPoint – this method takes in a decimal value corresponding to a x-coordinate and returns the y-coordinate based on the slope equation (y = x^exponent)

Write a class called **SineLine**

- This should implement Line
- Instance variable
  - amplitude: a decimal value corresponding to the line's amplitude
  - frequency: a decimal value corresponding to the line's frequency
- Create the following Constructors
  - Default
  - Parameterized Constructor
- Accessors and Mutators for each variable
- Create the following Methods
  - getYPoint – this method takes in a decimal value corresponding to a x-coordinate and returns the y-coordinate based on a sine wave equation (y = amplitude*sin(x*frequency))

Write a class called **SawLine**

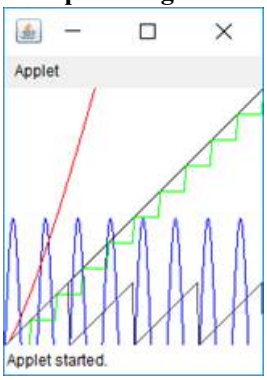
- This should implement Line
- Instance variable
  - modValue: a decimal value corresponding to the modulo peak of the wave
- Create the following Constructors
  - Default
  - Parameterized Constructor
- Accessors and Mutators for each variable
- Create the following Methods
  - getYPoint – this method takes in a decimal value corresponding to a x-coordinate and returns the y-coordinate based on the equation (y = x mod modValue)

Write a class called **StaircaseLine**

- This should implement Line
- Instance variable
  - width: a decimal value corresponding to the stair's width
  - height: a decimal value corresponding to the stair's height
- Create the following Constructors
  - Default
  - Parameterized Constructor
- Accessors and Mutators for each variable
- Create the following Methods
  - getYPoint – this method takes in a decimal value corresponding to a x-coordinate and returns the y-coordinate based the width and height. HINT(using integer division and multiplying that by the height will achieve the effect).

HINT: If the lines are looking weird it may be a good idea to print out each of the coordinates and observing what is going on in the method getYPoint

Example Dialog:



Lab Report Questions

1. Draw a UML class diagram for this project
2. Describe polymorphism

Finally:

Upload the files to the dropbox