In-Class Exercise: ShapeHeritance II

# Concepts:

* Inheritance, Abstract, Overrides, GUI with JOptionPane, & ArrayLists

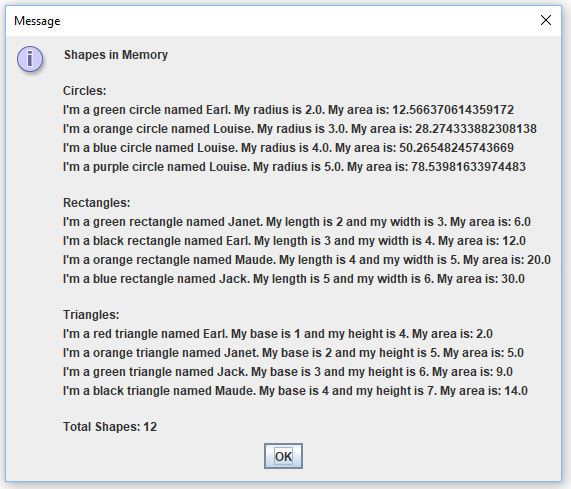
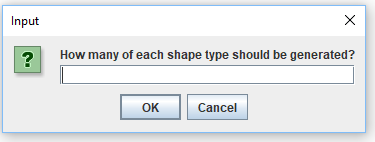
# Instructions

* This program is an extension of the first Shape-heritance exercise. Be sure to commit/push to GitHub before starting (or make a file backup), so you’ll have a copy of both exercises to refer to in the future
* Change your superclass to be an abstract class.
* Change the superclass’ getArea() method to be an abstract method.
* Each subclass must include an overridden implementation of the getArea() method, in which the actual math to calculate the area will be used.
* In Main(), include code to do the following:
  + Create a simple String array that holds several color names, which will be used to get a random color for each shape (use Random class).
  + Create ArrayLists for each subclass type of your shapes.
  + In GUI fashion, ask the user to enter a number (x).
  + Create x new objects of each subclass shape and add it to its respective arraylist.
  + Each new shape should have a random color assigned. Put whatever shape dimensions you want… either randomly generated or hard-coded is fine for now.
  + Build and display a GUI report by retrieving all saved shape objects and reporting its properties and calculating its area.
* **Bonus!:** Include a total count of all shapes in the report, without putting code to do so in any of the subclasses.

# Exercise Notes

* Don’t worry about input validation unless you want to.
* For the time being, you can ignore all Cancel buttons in the popups.

# Screenshots for Ideas below



**Note:** My version included a personal name for each shape. Not required! =)