<u>csTask53 Assignment</u>

The activities in this assignment require you to create and modify Java programs using BlueJ. The BlueJ project csTask53 already contains various classes for you to work on.

Follow the directions given for each part. At the end of class, all modified programs (class files) must be in your project, and the project must be copied to your shared Dropbox folder.

When questions are posed, write your responses as full sentences in the Readme.txt file in your project.

There are two parts to this assignment:

I. Another Type of Employee
II. Painting Shapes

I. Another Type of Employee

The classes Firm, Staff, StaffMember, Volunteer, Employee, Executive, and Hourly are from the text. This application illustrates inheritance and polymorphism. In this exercise you will add one more employee type to the class hierarchy (see Figure 9.1 in the text). The employee will be one that is an hourly employee but also earns a commission on sales. Hence the class, which we'll name Commission, will be derived from the Hourly class.

Write a class named Commission with the following features:

- It extends the Hourly class.
- It has two instance variables (in addition to those inherited):
    - the total sales the employee has made (type double)
    - the commission rate for the employee
        - The commission rate will be type double and will represent the percent (in decimal form) commission the employee earns on sales. For example, 0.2 would mean the employee earns 20% commission on sales.
- The constructor takes 6 parameters: the first 5 are the same as for Hourly (name, address, phone number, social security number, hourly pay rate) and the 6th is the commission rate for the employee. The constructor should call the constructor of the parent class with the first 5 parameters then use the 6th to set the commission rate.
- One additional method is needed: `public void addSales (double totalSales)` that adds the parameter to the instance variable representing total sales.
- The pay method must call the pay method of the parent class to compute the pay for hours worked then add to that the pay from commission on sales. (See the pay method in the Executive class as an example.) The total sales should be set back to 0 (Note: you don't need to set the hoursWorked back to 0—why not?).
- The toString method needs to call the toString method of the parent class then add the total sales to that.

To test your class, update the Staff class as follows:

- Increase the size of the array to 8.
- Add two commissioned employees to the staffList —make up your own names, addresses, phone numbers and social security numbers. Have one of the employees earn $6.25 per hour and 20% commission and the other one earn $9.75 per hour and 15% commission.
- For the first additional employee you added, put the hours worked at 35 and the total sales $400; for the second, put the hours at 40 and the sales at $950.

Compile and run the program. Make sure it is working properly.

II. Painting Shapes

In this exercise you will develop a class hierarchy of shapes and write a program that computes the amount of paint needed to paint different objects. The hierarchy will consist of a parent class Shape with three derived classes - Sphere, Rectangle, and Cylinder. For the purposes of this exercise, the only attribute a shape will have is a name and the method of interest will be one that computes the area of the shape (surface area in the case of three-dimensional shapes).

Do the following:

1. Write an abstract class Shape with the following properties:
   - An instance variable shapeName of type String
   - An abstract method area()
   - A toString method that returns the name of the shape

2. The Sphere class is a descendant of Shape. A sphere has a radius and its area (surface area) is given by the formula $4\pi r^2$. Define similar classes for a rectangle and a cylinder. Both the Rectangle class and the Cylinder class are descendants of the Shape class. A rectangle is defined by its length and width and its area is length times width. A cylinder is defined by a radius and height, and its area (surface area) is $2\pi rh + 2\pi r^2$. Define the toString method in a way similar to that for the Sphere class.

3. The Paint class represents a type of paint (which has a "coverage" and a method to compute the amount of paint needed to paint a shape). Correct the return statement in the amount method so the correct amount will be returned. Use the fact that the amount of paint needed is the area of the shape divided by the coverage for the paint. (NOTE: Leave the print statement - it is there for illustration purposes, so you can see the method operating on different types of Shape objects.)

4. The PaintThings class computes the amount of paint needed to paint various shapes. A Paint object has been instantiated. Add the following to complete the program:
   - Instantiate the three shape objects: `deck` to be a 20 by 35 foot rectangle, `bigBall` to be a sphere of radius 15, and `tank` to be a cylinder of radius 10 and height 30.
   - Make the appropriate method calls to assign the correct values to the three amount variables.
   - Run the program and test it. You should see polymorphism in action as the amount method computes the amount of paint for various shapes.