Practical-03: Object-Oriented Programming - Part II

Due 1 Sep by 22:00 **Points** 100 **Available** after 11 Aug at 22:00

Submission

Note: all class work must be submitted to your SVN repository. For this practice, you shall use:

```
https://version-control.adelaide.edu.au/svn/<YOUR-ID>/2019/s2/fcs/week-04/practical-03
```

Part II - Classes and Objects

Problem 01 - Basic Classes and Object

Object-Oriented Programming Concepts are very important. Without having an idea about OOPS concepts, you will not be able to design systems in the object-oriented programming model. In this part, you are required to develop several classes for different uses. For each class, you must develop accessors and mutators;

```
Base classes:

+ Person
++ attributes: givenName (String), lastName (String), age (int), gender (String), citizenship (String)

+ Shape
++ attributes: width (double), height (double), radius (double), length (double)

+ Book
++ attributes: title (String), year (int), publish (String), pages (int)

+ Animal
++ attributes: name (String), weight (double), favouriteFood (String), age (int)

Requirements:
+ Implement Accessors;
+ Implement Mutators;
+ Implement Basic Constructor;
+ Implement Parametric Constructor;
```

Problem 02 - Inheritance

By default, all classes in Java inherit from the Object class provided by Java. Therefore, the Object class is the superclass to all other classes and it defines methods that all its subclasses share. In this problem, you are required to extend from previous Base Classes the Children Classes.

```
Inheritance classes:

+ Person

++ Student
+++ attributes: marks (double []), disciplines (String [])
+++ methods: averageMarks(), displayDisciplines ()

```

```
++ Lecturer
+++ attributes: salary (double), disciplines (String [])
+++ methods: annualSalary(), displayDisciplines ()

++ Doctor
+++ attributes: speciality (String)

++ Patient
+++ attributes: diagnostic (String); inTime (String); prevTime (String);
--------------------------------------------------------------------------------

+ Shape
++ Rectangle
+++ methods: area()

++ Circle
+++ methods: area()

++ Triangle
+++ methods: area()

--------------------------------------------------------------------------------
+ Book

++ EBook
+++ attributes: url (String)

++ PhysicalBook
+++ attributes: shelf (int), aisle (int), floor (int)
+++ method: displayBookLocation()

--------------------------------------------------------------------------------
+ Animal

++ Lion
+++ methods: makeSound()

++ Turtle
+++ methods: makeSound()

++ Chameleon
+++ methods: makeSound()

Requirements:
+ Implement Accessors;
+ Implement Mutators;
+ Implement Basic Constructor;
+ Implement Parametric Constructor;
+ Implement methods required;
```

Problem 03 - Handling Constraints

Consider the Rectangle class you created in the previous exercise. Copy and change it such as you should have two data fields-width and height of int types. The class should have a display() method, to print the width and height of the rectangle separated by space.

DerivedClass

The RectangleArea class is derived from Rectangle class, i.e., it is the sub-class of the Rectangle class. The class should have a read_input() method, to read the values of width and height of the rectangle (from user input). The RectangleArea class should also overload the display() method to print the area (width*height) of the rectangle. The first and only line of input contains two space-separated integers denoting the width and height of the rectangle.

```
Constraints
1 <= width,height <= 10^3

Your code must handle these constraints;
```

The output should consist of exactly two lines:

1. In the first line, print the width and height of the rectangle separated by space.
2. In the second line, print the area of the rectangle.

```
Input:
12 9

Output:
12 9
108
```

Problem 04 - Abstract, Interface Classes, and Polymorphism

A local bike shop or local bicycle shop is a small business specializing in bicycle sale, maintenance, and parts. The expression distinguishes small bicycle shops from large chains and mail-orders.

In this problem, you are required to develop a Bicycle Repair Store. The system needs an Input/Output interface to keep track of the bicycles that have been serviced;

```
Interface Class:
+ Vehicle
++ methods:  changeGear (int), checkBrakes ()

Abstract Class

+ VehicleAbstract
++ attributes: color, year, numberGears

Base Class
+ Bicycle
++ attributes: is_serviced (bool), inDate (String), outDate (String), serviceResponsible (String)
++ methods: checkoutService()

Children Classes

+ Electric Bike
++ attributes: battery;
++ methods: chargeBike();

+ MountainBike
++ attributes: suspension (String), forks (string)
```

```

+ RoadBike
++ attributes: tyres (String)

+ Hibrid
++ attributes: suspension (String)



```

Basic Marking Scheme

Criteria	Ratings	Pts
Compilation In order to achieve full marks - your code must compile and run;		20.0 pts
Basic Functionality Your code (1) perform all the functions correctly, (2) use latest concepts learned in class, (3) has a clear, creative and sophisticated way of solving the problem.		40.0 pts
Functionality Extension Your code (1) perform all the functions correctly, (2) use latest concepts learned in class, (3) has a clear, creative and sophisticated way of solving the problem, and (4) you propose novel ways to solve the problems - or extra functionalities.		10.0 pts
Code Formatting Your code (1) has the right proportion of comments and place line and block comments correctly, (2) follow correct indentation every new indentation level, (3) has good variable naming, (4) has clear organization between tabs and is easy to read.		30.0 pts
		Total points: 100.0