United International University (UIU)

Dept. of Computer Science & Engineering (CSE)
Object-Oriented Programming Laboratory Assignment 1

The assignment questions are typically more challenging than your CTs. Doing this sincerely will benefit you in becoming a Java Pro.
Anyone adopting any unfair means will get a straight ZERO.

No.	Problem
1	You are trying to design a graphics software that lets you render 2D and 3D graphics. In the first phase, the software would support 2D and 3D points, squares/rectangles & cubes, and circles & spheres. You are to design a Java codebase for these shapes using the principles of OOP. The first class is the **Point2D** class, which has the attributes: *x*(int) and *y*(int). These are also private attributes. A constructor is used to initialize them both. The *x* attribute renders the point at a particular x-coordinate, and the y attribute does the same for the y-coordinate. The Point2D has a method **double findEuclidean(Point2D point)** that lets you find the Euclidean distance between another point, passed as a parameter to this method. [Use Math.sqrt() and the Math library in Java for this]. The second class is the **Point3D** class, which inherits from the Point2D class, but has an extra attribute *z*(int) that renders the point at the z-coordinate. This is also private. A constructor is used to initialize all three attributes. This class also has the **double findEuclidean(Point3D point)** but with a different implementation than Point2D. See here for 3D Euclidean distance calculation. The shapes squares, rectangles, cubes and so on need to be united under a single class. This is why you're going to create a class named **Shape2D**. Shape2D will have no attributes but will contain two methods, **double findPerimeter()** and **double findArea()**. The **findPerimeter()** method calculates the perimeter of the shape and the **findArea()** calculates the area of the shape. The **Rectangle** and **Circle** classes inherit from the Shape2D class. The Rectangle class has two attributes *length*(int) and *width*(int). These are private. A constructor is used to initialize them both. The Circle class has an attribute *radius*(int). This is private. A constructor is used to initialize the radius. You are to implement the perimeter and area methods for both the Rectangle and Circle class. The shape **Cuboid** inherits from the Rectangle class and has one extra attribute: *height*(int). You will need to implement the **double getVolume()** class using the **getArea()** class inside the implementation of the method. The shape **Sphere** inherits from the Circle class and has no extra attributes. You will need to implement the **double getVolume()** class using the **getArea()** class inside the implementation of the method. PTO

Once done, create a Main.java class to create and test out the objects. The Main.java class should be like this:

```
public static void main(String[] args) {
        Point2D point1 = new Point2D(13,13);
        Point2D point2 = new Point2D(19,19);
        System.out.println(point1.findEuclidean(point2)); // 8.48528

        Point3D point3 = new Point3D(13,13,13);
        Point3D point4 = new Point3D(19,19,19);
        System.out.println(point3.findEuclidean(point3)); // 10.3923

        Rectangle rect = new Rectangle(13,13);
        System.out.println(rect.findPerimeter()); // 26
        System.out.println(rect.findArea()); // 169

        Circle circle = new Circle(13);
        System.out.println(circle.findPerimeter()); // 81.6814089933
        System.out.println(circle.findArea()); // 530.929158457

        // write the code for the Cuboid and Sphere. Marks will be deducted if there are no test cases
    }
```

NOTE: ALL CLASSES SHOULD BE IN SEPARATE .java FILES.

2 You are working at a ride-sharing startup. Your task is to implement 5 classes: **User**, **Driver**, **Ride**, **Location**, and **RideSharingService**.

The **Location** class only has one attribute: *place*(String). This attribute is private. A constructor will initialize this attribute.

The **User** class has the attributes: *name*(String), *id*(int) and two methods: one **Ride requestRide(Location from, Location to)**, and **String getUserInfo()**. The **requestRide()** method will return an object of **Ride** and the **getUserInfo()** will return the name and id of the user as a String. The constructor will initialize all the attributes.

The **Ride** class has the attributes *rideId*, *fromLocation*(**Location**), *toLocation*(**Location**), and *isCompleted*(boolean). The class will have two methods: **completeRide(),** and **getRideInfo()**. The **completeRide()** method will set the isCompleted to true. The getRideInfo will return the rideId as a String. The constructor will initialize all the attributes.

The **Driver** class inherits from the **User** with some extra attributes: *vehicleType*(String), *isAvailable* (boolean). The class will have two methods: **acceptRide(Ride ride)**, **completeRide(Ride ride)**, and **toggleAvailability()**. The **acceptRide()** method will check if the driver is available or not (can you use isAvailable in this case?). If yes, then print “Ride Accepted” and change the isAvailable to false. If the driver is not available, then print "Driver is not available". The **completeRide()** method will complete the ride. If the ride is not completed (can you use the isCompleted attribute of the Ride class over here?), then set the ride as completed and print “Ride completed for <rideId>”. Finally, the method **toggleAvailability()** will toggle the value of the isAvailable attribute i.e. if it is false then it will become true and vice versa. The constructor will initialize all the attributes.

The **RideSharingService** has three attributes: rides (an array of **Rides**), users (an array of **Users**), and drivers (an array of **Drivers**). The class has the following methods: **registerUser(User user)**, **registerDriver(Driver driver)**, **requestRide(User user, Location from, Location to)**, and **completeRide(Driver driver, Ride ride)**. The method **registerUser(User user)** will add a new user to the array, **registerDriver(Driver driver)** will add a new driver to the array, and **requestRide(User user, Location from, Location to)** will add a ride to the array, making sure that the *isCompleted* value of the **Ride** object added to the array is false**.** The **completeRide(Driver driver, Ride ride)** will take the objects of the **Driver** and the **Ride** and simply change the *isCompleted* value of the **Ride** object to true and the *isAvailable* value of the **Driver** object to true**.** The constructor will take no parameters. It will be a blank constructor.

Implement the scenario. Create 3 objects for each of the classes except for the **RideSharingService**. Create only one object for it. Test out all the methods/implement them in the main method at least once i.e. call it once.

NOTE: ALL CLASSES SHOULD BE IN SEPARATE .java FILES.