COMP 1409

Intro to Software Development 1

Java

Assignment 2

Assignment 2

The purpose of this assignment is to exercise your knowledge from lessons 1 – 8.

Vehicle.java

Use the `Vehicle` class from the solution for Assignment 1.

Customer.java

The `Customer` class has the following instance variables

- `firstName`, `lastName`, `driversLicense`, `address`, `phoneNumber`

Provide a default constructor.

The overloaded constructor will receive data to initialize all the instance variables listed above.

The class provides both accessor and mutator methods for each instance variable, and also a method that returns the full name. The first letter of each name component must be uppercase and the rest of the letters must be lowercase, no matter how the names are passed to the constructor.

You must ensure that `firstName`, `lastName`, `address` and `driversLicense` fields are not null and are at least one character in length, otherwise the fields will not be "mutated".

PurchaseDate.java

The PurcahseDate class has the following instance variables

- `year`, `month`, `day`

The constructor looks like this

```
    public PurchaseDate(int theYear, int theMonth, int theDay)
```

The constructor must ensure

- that the year is no later than `CURRENT_YEAR`
- that months are `JANUARY` (1) to `DECEMBER` (12)
- and that days are `FIRST_DAY` (1) to `LAST_DAY` (31).

Use constants. If any of the parameters passed to the constructor are incorrect, use `CURRENT_YEAR`, `JANUARY`, and `FIRST_DAY` as the default settings. Do not worry about months with fewer than 31 days; assume all months have 31 days.

Provide both accessor and mutator methods for every instance variable, and also a method that returns the full date as a `String` in the exact format of `yyyy-mm-dd` (for example 2016-03-30).

The mutator methods must ensure that the year is no later than `CURRENT_YEAR`, that months are `JANUARY` to `DECEMBER` and that days are `FIRST_DAY` to `LAST_DAY`; otherwise the fields will not be “mutated”.

VehiclePurchase.java

The `VehiclePurchase` class has the following definitions for instance variables

```
private Customer customer
private PurchaseDate purchaseDate
private Vehicle vehiclePurchased
private boolean servicePackage
public static final double SERVICE_FEE = 500.00
```

The single `VehiclePurchase` constructor looks like this

```
public VehiclePurchase(Customer renter,
PurchaseDate purchaseDate, Vehicle vehiclePurchased,
boolean servicePackage)
```

Provide accessor methods for each instance variable in the class. Note that the accessors for `customer`, `purchaseDate` and `vehiclePurchased` will return the relevant reference types. There should be NO mutator methods for `customer`, `purchaseDate` or `vehiclePurchased`.

Provide a mutator method for `servicePackage`.

Provide the method `calculatePurchasePrice(double purchasePrice)` that first uses the existing `Vehicle` method to validate the parameter. If the `servicePackage` field is true then the `SERVICE_FEE` is added to the parameter and the Vehicle `sellingPrice` is reset to the new value.

Provide a method named `displayDetails()` that displays all information to the console for a purchase agreement, eg

```
Customer: Darby Dog
Purchase Date: 2014-05-20
Vehicle Description: Jalopies Are Us Vehicle Summary:
Vehicle: 1974 Chevrolet Monte Carlo
Stock Code:1974CevMC
Dealer Cost: $250.00
Selling Price: $895.95
```

```
Profit Margin: 72%
Dollar Profit: $645.95
SERVICE PACKAGE INCLUDED
```

Running the Application

First create 3 instances, one of each `Customer`, `PurchaseDate` and `Vehicle`.

Next, create a `VehiclePurchase` instance and pass the 3 instances into the constructor, along with the boolean value for `servicePackage`.

Be sure to comment your code with appropriate JavaDoc.

Be sure to use proper camelCasing or PascalCasing.

Be sure to use reasonable data types/reference types.

Submission

Compress and submit your source code to the Dropbox in D2L.