Assignment 5

Description

There is a store that allows consumers to rent different types of vehicles. The consumer only provide the range of money he can spend and how many days he wants to rent, then the program can return the information of vehicles that match his requirements.

General describe your task

Your job is to design those different types of vehicles including **OldVehicle**, **NewReleasedVehicle** and **Motorbike**, which are the subclasses of the **Vehicle** class.

A. Create classes: (15 points)

Create three subclasses of **Vehicle** including **OldVehicle**, **NewReleasedVehicle** and **Motorbike**.

B. Adding attributes: (10 points)

- **NewReleasedVehicle**: adding a private data field named **energy**, and the type of which is an enum type (**Energy**).
- **Motorbike**: adding a private data field named **deposit**, which is an integer and it means the deposit of motorbike.

C. Constructor. (15 points)

- **OldVehicle:** Only one constructor with two parameters including **name** and **basicPrice**.
- **NewReleasedVehicle:** Only one constructor with four parameters including **name**, **basicPrice**, **chargeable**(boolean) and **oiling**(boolean). In constructor, we need to give **energy** an initial value as follows

	HYBRID	ELECTRIC	GAS
chargeable	true	false	true
oiling	true	true	false

- **Motorbike:** Only one constructor with three parameters including **name**, **basicPrice** and **deposit**.

D. Override abstract method. (30 points)

You need to implement the abstract method **getRental()** in those three subclasses. The algorithm for calculating the rental value is as follows.

	Algorithm	Example
OldVehicle	If you rent vehicle lower than 3 days, the rental would be basicPrice, otherwise, the rental would be basicPrice+(days-3)*basicPrice* 20%	If you want to rent 5 days and the basic price is 400, the rental is 560
NewReleasedVehicle	If you rent vehicle lower than 4 days, the rental would be basicPrice, otherwise, the rental would be basicPrice+(days-4)*basicPrice* 30%	If you want to rent 5 days and the basic price is 400, the rental is 520
Motorbike	If you rent motorbike between 5(k-1)+1 and 5k days, the rental would be k*basicPrice+deposit.	If you want to rent 8 days, and the basic price is 400, and the deposit is 100, the rental is 900.

E. Override toString() method. (20 points)

The **toString()** method in all subclasses must invoke the **toString()** method in its superclass.

- **OldVehicle**: The return value of toString() method consists of two parts.
  1. A String "Old ".
  2. The return value of toString() method from its superclass.

- **NewReleaseVehicle**: The return value of toString() method consists of three parts:
  1. A String "New ".
  2. The return value of toString() method from its superclass.
  3. The value of **desc**, which is an attribute of Enum type **energy**.

- **Motorbike**: The return value of toString() method consists of two parts.
  1. A String "Motorbike ".
  2. The return value of toString() method from its superclass.

F. Implement interface. (20 points)

Only **OldVehicle** and **NewReleasedVehicle** need to implement the interface "**Insurance**".
In override method InsuanceDescription,
For **OldVehicle**, it needs to print "Purchased insurance"
For **NewReleasedVehicle**, it needs to print "Purchased high-value insurance"

Here is a Sample Test

```
public static void main(String [] args) {
        ArrayList<Vehicle> list=new ArrayList<Vehicle>();
        list.add(new OldVehicle("FORD-maverick",400));
        list.add(new NewReleasedVehicle("Tesla-Model2",1500,true,false));
        list.add(new Motorbike("YAMAHA-ys150",600,200));

        for(Vehicle v:list) {
            System.out.println(v);
            if(v instanceof Insurance) {
                ((Insurance) v).InsuanceDescription();
            }
        }

    }
```

Sample Output:

```
Old Vehicle [FORD-maverick, 400.0]
Purchased insurance
New Vehicle [Tesla-Model2, 1500.0](Electric only)
Purchased high-value insurance
Motorbike Vehicle [YAMAHA-ys150, 600.0]
```

What to submit

- Compress all following files into one folder, and then submit them into sakai.
  - Customer.java
  - Energy.java
  - Insurance.java
  - Vehicle.java
  - OldVehicle.java
  - NewReleasedVehicle.java
  - Motorbike.java
  - vehicle.txt

- Do not do any modification of those files given to you. (Customer.java, Energy.java, Insurance.java, Vehicle.java, vehicle.txt)
- No package information included.