**Programming Project 2**

The second programming project involves writing a program that computes the sales tax for a collection of automobiles of different types and the total cost for each of those automobiles. This program consists of three classes and the Program Core.

The first class is the ***Automobile*** class, which contains the automobile’s Make and Model [one data element], purchase price, and total cost which are specified in whole US dollars. It should have three methods:

1. A ***constructor*** that allows the Make and Model and purchase price to be initialized.
2. A method named ***salesTax*** that returns the base sales tax computed as 5% of the sales price.
3. A ***toString*** method that returns a string containing the Make and Model of the automobile, the sales price, the adjusted sales tax, and total cost appropriately labeled.

The ***Automobile*** class has two subclasses. The first subclass is ***Electric***. It has an additional instance variable that contains its weight in pounds stored as an integer. It should have the same three methods:

1. A ***constructor*** that allows the automobile’s Make and Model, purchase price, and weight to be initialized.
2. An overridden method ***salesTax*** that returns the total sales tax. The sales tax for an Electric Automobile consists of the base sales tax of 5% that applies to all automobiles minus a discount. If the weight of the automobile is 3,000 pounds or less the discount is $200 off the sales tax; otherwise it is $150 off the sales tax.
3. An overridden ***toString*** method that returns a string containing the Make and Model of the automobile, the sales price, adjusted sales tax, the weight of the automobile, and the total cost appropriately labeled.

The second subclass is ***Hybrid***. It has an additional instance variable that contains the number of miles per gallon stored as an integer. It should have the same three methods:

1. A ***constructor*** that allows the automobile’s Make and Model, purchase price, and miles per gallon to be initialized.
2. An overridden method ***salesTax*** that returns the total sales tax. The sales tax for a Hybrid Automobile consists of the base sales tax of 5% minus a discount. If the miles per gallon is 40 mpg or less, the discount is $100 off the sales tax; otherwise there is an additional discount above the $100 of $2.00 for every mile per gallon in excess of 40 mpg.
3. An overridden ***toString*** method that returns a string containing the Make and Model of the automobile, the sales price, adjusted sales tax, the number of miles per gallon, and total cost appropriately labeled.

No additional public methods should be included in any of the above three classes.

Finally, the Program Core contains the ***main*** method. It should generate one of the GUI shown below in three versions:

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

After entering the information about the Make and Model and its sales price, select the type of the automobile. If another entry box displays, enter the appropriate information. Once all the information is added to the *Automobile Sales Tax Calculator* Window, click the *Compute Sales Tax* button. The appropriate amount of sales tax displays in the TextField to its right. The default Automobile Type is ***Other*** so that radio button in the radio group should be selected. When ***Other*** Type is selected the Labels and TextFields for ***Hybrid*** and ***Electric*** are hidden. [Hints: [Component].setVisible(false).];[Component].setEditable(false)] When ***Hybrid*** Type is selected the *Miles per Gallon* Label and the its TextField display but are hidden if not selected. When ***Electric*** Type is selected the *Weight in Pounds* Label and its TextField display but are hidden if not selected.

If non-integer values are entered in any of the fields that require integers, an error message should be displayed in a JOptionPane window and after clicking OK the cursor returns to the same TextField for another entry.

In addition, the object should be stored in an array of type Automobile. That array should contain at least five automobiles entries/records/rows.

Clicking the *Clear Fields* button should clear all TextFields and hide the Labels and TextField for the Hybrid and Electric Types.

Clicking the *Display Report* button should produce in the IDE Output Window the information about all the automobiles stored in the array during the processing of the program. An example of the report that should be displayed is shown below with three entries in the Automobile array:

Make and Model: Toyota Prius

Sales Price: 30000.00

Sales Tax: 1390.00

Vehicle Type: Hybrid

Vehicle Unique (MPG): 45

Total Cost: 31390.00

Make and Model: Ford Fusion

Sales Price: 21000

Sales Tax: 1050.00

Vehicle Type: Other

Vehicle Unique (None): 0

Total Cost: 22050.00

Make and Model: Chevrolet Bolt

Sales Price: 32000

Sales Tax: 1450.00

Vehicle Type: Electric

Vehicle Unique (Weight): 3250

Total Cost: 33450.00

Be sure to follow good programming practices, which means making all instance variables private, naming all constants and avoiding the duplication of code. Furthermore, you must select enough different kinds of automobiles in your Test Plan to completely test the program.