Project 2

Attiqa Sheikh

CMIS 242

April 11, 2020

Java Code

Automobile Class

```
/*Filename: Automobile
Developer: Attiqa A. Sheikh
Date: April 11, 2020
Purpose: Contains automobile's make, model, and purchase price in whole dollars.
*/
public class Automobile {
  private String makeAndmodel;
  private double purchasePrice;
//constructor that initializes intance variables makeAndmodel and purchasePrice
  public Automobile(String makeAndmodel, double purchasePrice){
    super();
    this.makeAndmodel = makeAndmodel;
    this.purchasePrice = purchasePrice;
    }
  public String getMakeAndmodel(){
    return makeAndmodel;
    }
  public double getPurchasePrice(){
    return purchasePrice;
    }
  public void setPurchasePrice(){
    this.purchasePrice = purchasePrice;
    }
  public double salesTax(){
  //calculate sales tax by multiplying purchase price by 5%
    return this.purchasePrice * 0.05;
    }
  public String toString(){
  //prints out make and model, sales price, and sales tax of vehicle
    return ("Make and Model: " + makeAndmodel + "\n" + "Sales Price: " + purchasePrice + "\n" + "Sales
Tax: " + this.salesTax() + "\n");
    }
    }
```

Electric subclass

```
/*Filename: Electric
Developer: Attiqa A. Sheikh
Date: April 11, 2020
Purpose: Stores weight of electric vehicle, calculates discount of  $200 into sale price
if vehicle is more than 3000 pounds, if not then calculates discount of $150
*/
class Electric extends Automobile{
  private int weight;
//constructor to initialize make and model, purchase price, and weight
  public Electric(String makeAndmodel, double purchasePrice, int weight){
    super(makeAndmodel, purchasePrice);
    this.weight = weight;
    }
  @Override
  //overriden salesTax method
  public double salesTax(){
  /*calculates sales tax and applies discount for
  vehicle according to weight*/
    double salesTaxPrice = super.salesTax();
      if(weight > 3000){
        if(salesTaxPrice - 200 >= 0){
        return salesTaxPrice - 200;
        }
        else {
          return 0.0;
          }
        }else{
          if(salesTaxPrice - 150 >= 0){
          return salesTaxPrice - 150;
          }else {
            return 0.0;
            }}}
  public int getWeight(){
    return weight;
    }
  public void setWeight(){
    this.weight = weight;
    }
  @Override
  //overriden toString method that prints make and model, sales price, sales tax, weight, and vehice type
  public String toString(){
    return "Make and Model: " + this.getMakeAndmodel() + "\n" + "Sales Price: " + this.getPurchasePrice() +
"\n" + "Sales Tax: " + this.salesTax() + "\n" + "Weight: " + this.getWeight() + "\n" + "Electric Vehicle\n";
    }}
```

Hybrid subclass

```
/*Filename: Hybrid
Developer: Attiqa A. Sheikh
Date: April 11, 2020
Purpose: Stores the mile per gallon of vehicle. If vehicle mpg is
less than 40, then a discount of $100 is applied. If mpg is more
than 40, then an additional discount of $2 every mpg is applied.
*/

class Hybrid extends Automobile {
  private int mpg;
//constructor that initializes make and model, purchase price, and mpg
  public Hybrid(String makeAndmodel, double purchasePrice, int mpg){
    super(makeAndmodel, purchasePrice);
    this.mpg = mpg;
    }
  @Override
//overriden method that applies disount according to vehicle mpg
  public double salesTax(){
    double salesTaxPrice = super.salesTax();

       if(mpg < 40) {
        if(salesTaxPrice - 100 >=0) {
           return salesTaxPrice - 100;
        }else {
           return 0.0;
        }
     }else {
        int disct = (this.mpg - 40) * 2;
        if(salesTaxPrice - disct-100 >= 0) {
           return salesTaxPrice - disct -100;
        }else {
           return 0.0;
        }
     }
  }
  public int getMpg(){
    return mpg;
    }
  public void setMpg(){
    this.mpg = mpg;
    }

  @Override
//overriden toString method that prints make and model, sales price, sales tax, hybrid vehicle and mpg
  public String toString(){
    return "Make and Model: " + this.getMakeAndmodel() + "\n" + "Sales Price: " + this.getPurchasePrice()
+"\n" + "Sales Tax: " + this.salesTax() +"\n" + "Hybrid Vehicle \n" + "MPG: " + this.getMpg() +"\n";
}}
```

Project2 and main method

```
/*Filename: Project2
Developer: Attiqa A. Sheikh
Date: April 11, 2020
Purpose: Contains the main method. Generates the GUI for Automobile Sales
Tax Calculator
*/
```

```
import java.awt.Color;
import java.awt.Font;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;
import java.util.List;

import javax.swing.BorderFactory;
import javax.swing.ButtonGroup;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JRadioButton;
import javax.swing.JTextField;

public class Project2 extends JFrame implements ActionListener{
//to display fields in GUI
  JLabel makeAndModelLabel,salesPriceLabel;
  JTextField makeAndModel, salesPrice;

  JPanel up,middle,down;

  JRadioButton hybrid,electric,other;
  ButtonGroup group;

  JLabel mpgLabel,weightLabel;
  JTextField mpg,weight;

  JButton computeSalesTax,clearFields,displayReport;
  JLabel output;


  List<Automobile> autoMobiles;

  Project2(){
     setTitle("Automobile Sales Tax Calculator");
//initializes all components of the program
     autoMobiles = new ArrayList<>();
     setLayout(null);
     setSize(600,450);
     up = new JPanel(new GridLayout(2,2,10,10));
     middle = new JPanel(new GridLayout(3,3,10,10));
     middle.setBorder(BorderFactory.createTitledBorder("Automobile Type"));
     down = new JPanel(new GridLayout(2,2,10,10));
```

```
makeAndModelLabel = new JLabel("Make and Model");
salesPriceLabel = new JLabel("Sales Price");
makeAndModel = new JTextField(20);
salesPrice = new JTextField(20);
up.add(makeAndModelLabel);
up.add(makeAndModel);
up.add(salesPriceLabel);
up.add(salesPrice);

hybrid = new JRadioButton("Hybrid");
electric = new JRadioButton("Electric");
other = new JRadioButton("Other");
ButtonGroup group = new ButtonGroup();
group.add(hybrid);
group.add(electric);
group.add(other);

mpgLabel = new JLabel("Miles per Gallon");
weightLabel = new JLabel("Weight in Pounds");
mpg = new JTextField(20);
weight = new JTextField(20);

middle.add(hybrid);
middle.add(mpgLabel);
middle.add(mpg);

middle.add(electric);
middle.add(weightLabel);
middle.add(weight);

middle.add(other);

computeSalesTax = new JButton("Compute Sales Tax");
clearFields = new JButton("Clear Fields");
displayReport = new JButton("Display Report");
output = new JLabel("");
output.setBorder(BorderFactory.createLineBorder(new Color(132,141,149),1));

down.add(computeSalesTax);
down.add(output);
down.add(clearFields);
down.add(displayReport);

up.setBounds(80,30,400,50);
middle.setBounds(10,100,550,120);
down.setBounds(60,250,400,80);
add(up);
add(middle);
add(down);
```

```
    computeSalesTax.addActionListener(this);
    clearFields.addActionListener(this);
    displayReport.addActionListener(this);

    other.addActionListener(this);
    hybrid.addActionListener(this);
    electric.addActionListener(this);

    output.setEnabled(false);
    other.doClick();
    output.setForeground(Color.BLUE);
    output.setFont(new Font(Font.SANS_SERIF, Font.BOLD, 15));
  }
/*Method that returns converted double from price if it is a double
or returns -1.0 for price value*/
  public Double isValidPrice(String price) {
    try {
      Double priceValue = Double.parseDouble(price.trim());
      if(priceValue <= 0) {
        priceValue = -1.0;
      }
      return priceValue;
    }catch(Exception e) {
      return -1.0;
    }
  }
/*Returns integer from num if value is an integer or
it return -1*/
  public Integer isValidInteger(String num) {
    try {
      Integer intValue = Integer.parseInt(num.trim());
      if(intValue <= 0) {
        intValue = -1;
      }
      return intValue;
    }catch(Exception e) {
      return -1;
    }
  }
/*Method that adds automobile object to the array*/
  public void addToList(Automobile mobile) {

    if(autoMobiles.size() < 5) {
      autoMobiles.add(mobile);
    }else {
      autoMobiles.remove(0);
      autoMobiles.add(mobile);
    }
  }
/*Checks if data is valid  for Other class
calculates the tax and adds it to the array
sets the sales tax to the output label*/
  public void saveOtherReport() {
     Double price = isValidPrice(salesPrice.getText());
     if(price != -1.0) {
        Automobile mobile = new Automobile(makeAndModel.getText(),price);
        output.setText(String.format("%.2f",mobile.salesTax()));
        addToList(mobile);
     }else {
        JOptionPane.showMessageDialog(this, "Invalid sales price, sales price must be greater than
0","ERROR",JOptionPane.ERROR_MESSAGE);
     }
  }
/*Method checks if data is valid for Hybrid class and calculates
then adds tax to the array
sets the tax to output label*/
  public void saveHybridReport() {
     Double price = isValidPrice(salesPrice.getText());
     if(price != -1.0) {
        Integer mpgValue = isValidInteger(mpg.getText());
        if(mpgValue != -1) {
           Hybrid mobile = new Hybrid(makeAndModel.getText(),price,mpgValue);
           output.setText(String.format("%.2f",mobile.salesTax()));
           addToList(mobile);
        }else {
           JOptionPane.showMessageDialog(this, "Invalid MPG, MPG must be greater than
0","ERROR",JOptionPane.ERROR_MESSAGE);
        }
     }else {
        JOptionPane.showMessageDialog(this, "Invalid sales price, sales price must be greater than
0","ERROR",JOptionPane.ERROR_MESSAGE);
     }
  }
/*Method checks if entered data is valid,
calculates and adds tax to the array
sets tax to output label*/
  public void saveElectricReport() {
     Double price = isValidPrice(salesPrice.getText());
     if(price != -1.0) {
        Integer weightValue = isValidInteger(weight.getText());
        if(weightValue != -1) {
           Electric mobile = new Electric(makeAndModel.getText(),price,weightValue);
           output.setText(String.format("%.2f",mobile.salesTax()));
           addToList(mobile);
        }else {
           JOptionPane.showMessageDialog(this, "Invalid weight, weight must be greater than
0","ERROR",JOptionPane.ERROR_MESSAGE);
```

```
        }
     }else {
        JOptionPane.showMessageDialog(this, "Invalid sales price, sales price must be greater than
0","ERROR",JOptionPane.ERROR_MESSAGE);
     }
  }

  @Override
  public void actionPerformed(ActionEvent ae) {
     if(ae.getSource() == computeSalesTax) {

        if(other.isSelected()) {
           saveOtherReport();
        }else if(hybrid.isSelected()) {
           saveHybridReport();
        }else {
           saveElectricReport();
        }

     }else if(ae.getSource() == clearFields) {
        resetForm();

     }else if(ae.getSource() == displayReport) {
        displayReports();
     }else if(ae.getSource() == other) {
        mpg.setEnabled(false);
        weight.setEnabled(false);
        output.setText("");
        mpg.setText("");
        weight.setText("");
     }
     else if(ae.getSource() == hybrid) {
        mpg.setEnabled(true);
        weight.setEnabled(false);
        output.setText("");
        weight.setText("");
     }
     else if(ae.getSource() == electric) {
        mpg.setEnabled(false);
        weight.setEnabled(true);
        mpg.setText("");
        output.setText("");
     }
   }
  public void resetForm() {
     makeAndModel.setText("");
     salesPrice.setText("");
     mpg.setText("");
     weight.setText("");
```

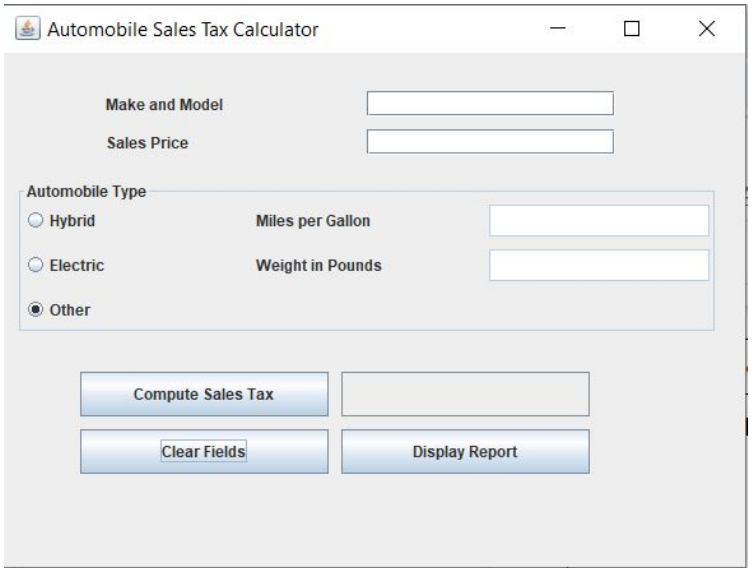
```
      other.setSelected(true);
      output.setText("");

      other.doClick();
   }
   public void displayReports() {
      String result = "";
      for(Automobile mobile:autoMobiles) {
         result += mobile+"";
      }
      JOptionPane.showMessageDialog(this, result, "Automobile Report",
JOptionPane.INFORMATION_MESSAGE);
      System.out.println(result);
   }
   public static void main(String[] args) {
      Project2 mainFrame = new Project2();
      mainFrame.setVisible(true);
      mainFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
   }
}
```

Test Cases

	Test 1	**Test 2**	**Test 3**
Make & Model	Lexus LX570	Mercedes Benz E-Class	Tesla Model S
Sales Price	$86,480	$81,650	$119,000
Hybrid (MPG)	No	Yes (20 mpg)	No
Electric(Weight)	No	No	Yes(4647 lb)
Other	Yes	No	No
Compute Sales Tax	4324.00	3982.50	5750.00
Display Report	Make and Model: Lexus LX570 Sale Price: 86480.0 Sales Tax: 4324.0	Make and Model: Mercedes Benz E-Class Sale Price: 81650.0 Sales Tax: 3982.5 Hybrid Vehicle MPG: 20	Make and Model: Tesla Model S Sale Price: 81650.0 Sales Tax: 5750.0 Weight: 4647 Electric Vehicle

Screen Captures



Make and Model

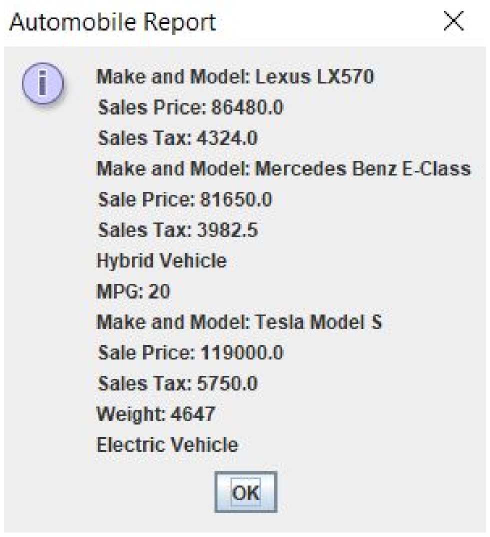
Sales Price

Automobile Type

( ) Hybrid Miles per Gallon

( ) Electric Weight in Pounds

(●) Other



```
Make and Model: Lexus LX570
Sales Price: 86480.0
Sales Tax: 4324.0
Make and Model: Mercedes Benz E-Class
Sale Price: 81650.0
Sales Tax: 3982.5
Hybrid Vehicle
MPG: 20
Make and Model: Tesla Model S
Sale Price: 119000.0
Sales Tax: 5750.0
Weight: 4647
Electric Vehicle
```