|  |
| --- |
| Tollbooth Program |

Project Details

|  |  |
| --- | --- |
| Project Title | Tollbooth Program |
| Registration Number | **2020-CS-139** |
| Summary | *This project first get all the information about the vehicle and the in the end calculate the toll-tax of the vehicle.*  *If the vehicle is Govt. Authorized then no toll-tax on that vehicle.* |
| Features | * *User can create the* ***New*** *Receipt* * *See the details of* ***Previous*** *receipts* |

Used Classes Details

|  |  |
| --- | --- |
| Classes | * Car * Receipt * Tollbooth * Driver (Main Class) |

|  |
| --- |
| Class Driver (Main Class) |

|  |  |
| --- | --- |
| Methods Used in the Class | * public static void main(String[] args) * tollOb.display(); * tollOb.finalize(); |
| Explanation | * public static void main(String[] args)   Is used to Execute the Program.   * tollOb.display();   Is used to **display** the list of the **Receipts**.   * tollOb.finalize();   Is used to Save the Data on the **File.** |
| Variables used | *int start = -1;*  *String input="";*  *String option="";* |
| Libraries Used | *import java.util.\*;*  *import javax.swing.JOptionPane;*  *import java.sql.Timestamp;* |
| Objects Used | *Car carOb = new Car();*  *Receipt receiptOb = new Receipt();*  *Tollbooth tollOb = new Tollbooth("****GRW Toll Plaza****");* |
| Use Of loop and Switch | * ***Do While*** *loop is used to control the execution of the* ***Main Menu.*** * *Also Switch Statement is also used in Main Menu.*   *Also on pressing case 2*  *A* ***Sub-Menu*** *is opened****,*** *and do while and Switch also used there.*  *Case 1:To view Recent One*  *Case 2:ToView all*  *Case 3:To delete one*  *Case 4: To Go Back to main Menu.* |

|  |
| --- |
| Class Car |

|  |  |  |  |
| --- | --- | --- | --- |
| Class-Fields | private String model;  private int make;  private String carNumber;  private int engineCapicity;  private int seatingCapicity;  private String driverName;  private boolean isGovtVechicle; | | |
| Constructors | /\*\*Constructor for carDriverName  \* @param model  \* @param make  \* @param carNumber  \* @param engineCapicity  \* @param seatingCapicity  \* @param driverName  \* @param isGovtVechicle  \*/  public Car(String modell,int make,String carNumber,int engineCapicity,  int seatingCapicity,String driverName,boolean isGovtVechile)  {  model = modell;  this.make = make;  this.carNumber = carNumber;  this.engineCapicity = engineCapicity;  this.seatingCapicity = seatingCapicity;  this.driverName = driverName;  this.isGovtVechicle = isGovtVechile;  }  This Constructor is used to initialize the related information about Car and Driver.  /\*\* Default Constructor  \*/  public Car()  {    } | | |
| Method Used | /\*\*  \* To Calculate Toll of The Car  \* @return It return total toll of Car  \*/  public double calculateToll()  {  double toll = 10;  if(isGovtVechicle == true)  {  // No toll for Govt vechile  return 0;  }  if(engineCapicity < 1000)  {  return toll;  }  else if(engineCapicity >= 1000 && engineCapicity <=1300)  {  toll = (engineCapicity / 100) + toll;  }  else if(engineCapicity >1300 && engineCapicity <=1800)  {  toll = (engineCapicity / 65) + toll;  }  else if(engineCapicity > 1800)  {  toll = (engineCapicity / 50) +toll;  }  // Seating capicity  if(seatingCapicity >4)  {  for(int i=4;i<seatingCapicity;i++)  {  toll = toll + 5;  }  }  // make  if(make < 2018)  {  for(int i =2018;i<make;i++)  {  toll = toll+ 2;  }  }  return toll;  } | | |
| Setter and Getter | /\*\*  \* Setter is for carDriverName  \* @param name  \* @return method return String  \*/  public void setName(String name)  {  driverName = name;  }  // Getter  /\*\* Getter for model  \* @return method return String  \*/  public String getModel()  {  return model;  }  /\*\* getter for make  \* @return method will return int  \*/  public int getMake()  {  return make;  }  /\*\* getter for carNumber  \* @return String  \*/  public String getcarNumber() {  return carNumber;  }  /\*\* getter for engineCapacity  \*@return method return int  \*/  public int getengineCapacity() {  return engineCapicity;  }  /\*\* getter for seatingCapacity  \*@return method return int  \*/  public int getseatingCapacity() {  return seatingCapicity;  }  /\*\* getter for DriverName  \*@return method return String  \*/  public String getdriverName() {  return driverName;  }  /\*getter for isGovtVehicle  \*@return method return Boolean  \*/  public boolean getisGovtVehicle() {  return isGovtVechicle;} | | |
| Validators | | /\*\* Validator for CarNAme  \* @param carName  \*@return method return Boolean  \*/  public boolean validCarName(String carName)  {  boolean flag = false;  int size = carName.length();  if (size <= 28) {  for (int i = 0; i < size; i++)  {  if ((carName.charAt(i) >= 'A' && carName.charAt(i) <= 'Z')  || (carName.charAt(i) >= 'a' && carName.charAt(i) <= 'z') || carName.charAt(i) == ' '  || carName.charAt(i) == '\_' || carName.charAt(i) == '-')  {  flag = true;  }  else  {  return false;  }  }  }  return flag;  }  /\*\* Validator for make  \*@return method return Boolean  \*@param year  \*/  public boolean validMake(int year)  {  boolean flag;  String yearr = String.valueOf(year);  if (yearr.length() == 4)  {  flag = true;  }  else  {  return false;  }  return flag;  }  /\*\* Validator for carNumber  \*@return method return Boolean  \*@param carNumber  \*/  public boolean validCarNumber(String carNumber)  {  boolean flag = false;  if (carNumber.length() == 11)  {  if (carNumber.charAt(3) == '-' && carNumber.charAt(6) == '-')  {  for (int i = 0; i < 3; i++)  {  if (carNumber.charAt(i) >= 'A' && carNumber.charAt(i) <= 'Z')  {  flag = true;  } else {  return false;  }  }  for (int i = 4; i < 6; i++) {  if (carNumber.charAt(i) >= '0' && carNumber.charAt(i) <= '9') {  flag = true;  } else {  return false;  }  }  for (int i = 7; i < 11; i++) {  if (carNumber.charAt(i) >= '0' && carNumber.charAt(i) <= '9') {  flag = true;  } else {  return false;  }  }  }  }  return flag;  }  /\*\* Validator for DriverName  \*@return method return Boolean  \*@param name  \*/  public boolean validDriverName(String name)  {  boolean flag = false;  for (int i = 0; i < name.length(); i++)  {  if ((name.charAt(i) >= 'a' && name.charAt(i) <= 'z')  || (name.charAt(i) >= 'A' && name.charAt(i) <= 'Z') || (name.charAt(i) == ' '))  {  flag = true;  } else  {  return false;  }  }  return flag;  }    /\*\* Validator for GovtCar  \*@return method return Boolean  \*@param govt  \*/  public boolean validGovtCar(String govt)  {  boolean flag = false;  if (govt.equals("Yes"))  {  return true;  } else  {  return false;  }  }  /\*\* Validator for EngineCapacity  \*@return method return Boolean  \*@param capicaty  \*/  public boolean validatengCap(int capicaty)  {  boolean flag = false;  if (capicaty >= 600 && capicaty <= 6599)  {  return true;  }  return flag;  }  /\*\* Validator for CarSeats  \*@return method return Boolean  \*@param seats  \*/  public boolean validatCarseats(int seats)  {  boolean flag = false;  if (seats >= 2 && seats <= 10)  {  return true;  }  return flag;  } |

|  |
| --- |
| Class Receipt |

|  |  |
| --- | --- |
| Class Fields | Car me = new Car();  private Timestamp timeStamp;  private double tollTex; |
| Setters | /\*\* Setter For tollTex  \* @param toll  \*/  public void setTollTex(double toll)  {  tollTex = toll;  }  /\*\* Setter for TimeStamp  \* @param date  \*/  public void setTimeStamp(Timestamp date)  {  timeStamp = date;  } |
| Getters | /\*\* getter for tollTex  \* @return double will return  \*/  public double getTollTex()  {  return tollTex;  }  /\*\* getter for TimeStamp  \*@return String is return type  \*/  public Timestamp getTimeStamp()  {  return timeStamp;  }  } |
| Libraries | import java.sql.Timestamp; |

|  |
| --- |
| Class Tollbooth |

|  |  |
| --- | --- |
| Fields | private String location;  List<Receipt> list; |
| Constructors | /\*\* Default Constructor for Tollbooth  \*/  public Tollbooth()  {    }  /\*\* Constructor for Tollbooth And to Read Data from the file  \* @param Loction  \*/  public Tollbooth(String Location)  {  this.location = Location;  this.list = new ArrayList<>();  Receipt recok = new Receipt();  try  {  FileReader obj = new FileReader("receipt.txt");  BufferedReader buobj= new BufferedReader(obj);  String line = "0";  while(line!=null)  {  recok = new Receipt();  line = buobj.readLine();  String []arr = line.split("/");  String name = arr[0];  String model = arr[1];  String make = arr[2];  int intMake = Integer.parseInt(make);  String carNo = arr[3];  int seatsCap = Integer.parseInt(arr[4]);  int engineCap = Integer.parseInt(arr[5]);  boolean govt = Boolean.parseBoolean(arr[6]);  Timestamp time = Timestamp.valueOf(arr[7]);  double toll = Double.parseDouble(arr[8]);  recok.me = new Car(model,intMake,carNo,engineCap,seatsCap,name,govt);  recok.setTimeStamp(time);  recok.setTollTex(toll);  list.add(recok);  }  buobj.close();  obj.close();  }  catch(Exception ex)  {    }  } |
| Getters | /\*\*Getter  \* @return String will return  \*/  public String getlocation()  {  return location;  } |
| Methods | /\*\* To Write the Data on the file  \*/  protected void finalize() throws Throwable  {  try  {  FileWriter obj = new FileWriter("receipt.txt");  String line = "";  boolean flag = false;  if(list.isEmpty())  {  JOptionPane.showMessageDialog(null,"File is Empty...");  }  else  {  for(int i=0;i<list.size();i++)  {  line = line+ list.get(i).me.getdriverName() + "/";  line = line+ list.get(i).me.getModel()+"/";  line = line+ list.get(i).me.getMake()+"/";  line = line+ list.get(i).me.getcarNumber()+"/";  line = line+ list.get(i).me.getseatingCapacity()+"/";  line = line+ list.get(i).me.getengineCapacity()+"/";  line = line+ list.get(i).me.getisGovtVehicle()+"/";  line = line+ list.get(i).getTimeStamp()+"/";  line = line+ list.get(i).getTollTex()+"\n";  }  obj.write(line);  obj.flush();  obj.close();    JOptionPane.showMessageDialog(null,"File saved successfuly!!!");  super.finalize();    }  }  catch(Throwable e)  {  throw e;  }  }  /\*\* To output  \* @param option to decide whether to show All list or Recent One.  \*/  public void display(int option)  {  String display = "";    if(list.isEmpty())  {  JOptionPane.showMessageDialog(null,"Plz Enter Receipt First.....");    }  else  {  if(option==1)  {  for(int i=0;i<list.size();i++)  {  display = display + "Receipt "+" #"+(i+1)+"\n";  display = display + getlocation()+"\n" + "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"+"\n" ;  display = display + "Name of the Driver : "+ list.get(i).me.getdriverName()+"\n";  display = display + "Car Details : "+"\n";  display = display + "Car Model :" + list.get(i).me.getModel()+"\n"+  "Car Year : " + list.get(i).me.getMake() + "\n";  display = display + "Car Number : " + list.get(i).me.getcarNumber()+"\n"+  "Car's Seating Capacity : "+list.get(i).me.getseatingCapacity() + "\n";  display = display + "Engine Capacity : "+list.get(i).me.getengineCapacity()+"\n";  display = display + "Tex details : "+"\n";  display = display + "Time :"+list.get(i).getTimeStamp()+"\n";  if(list.get(i).me.getisGovtVehicle())  {  display = display + "Car is Government,No toll tex"+"\n";  }  else  {  display = display + "Total Toll tex : "+list.get(i).getTollTex()+"\n"+"\t\n";  }    }  JOptionPane.showMessageDialog(null, display);  }  else if(option==2)  {  int size=list.size();  size=size-1;  display = display + "Receipt "+" #"+(size)+"\n";  display = display + getlocation()+"\n" + "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_"+"\n" ;  display = display + "Name of the Driver : "+ list.get(size).me.getdriverName()+"\n";  display = display + "Car Details : "+"\n";  display = display + "Car Model :" + list.get(size).me.getModel()+"\n"+  "Car Year : " + list.get(size).me.getMake() + "\n";  display = display + "Car Number : " + list.get(size).me.getcarNumber()+"\n"+  "Car's Seating Capacity : "+list.get(size).me.getseatingCapacity() + "\n";  display = display + "Engine Capacity : "+list.get(size).me.getengineCapacity()+"\n";  display = display + "Tex details : "+"\n";  display = display + "Time :"+list.get(size).getTimeStamp()+"\n";  if(list.get(size).me.getisGovtVehicle())  {  display = display + "Car is Government,No toll tex"+"\n";  }  else  {  display = display + "Total Toll tex : "+list.get(size).getTollTex()+"\n"+"\t\n";  }  JOptionPane.showMessageDialog(null, display);  }  }    } |
| Libraries | import java.io.FileWriter;  import java.io.FileReader;  import java.io.BufferedReader;  import java.sql.Timestamp;  import java.util.\*;  import javax.swing.JOptionPane; |