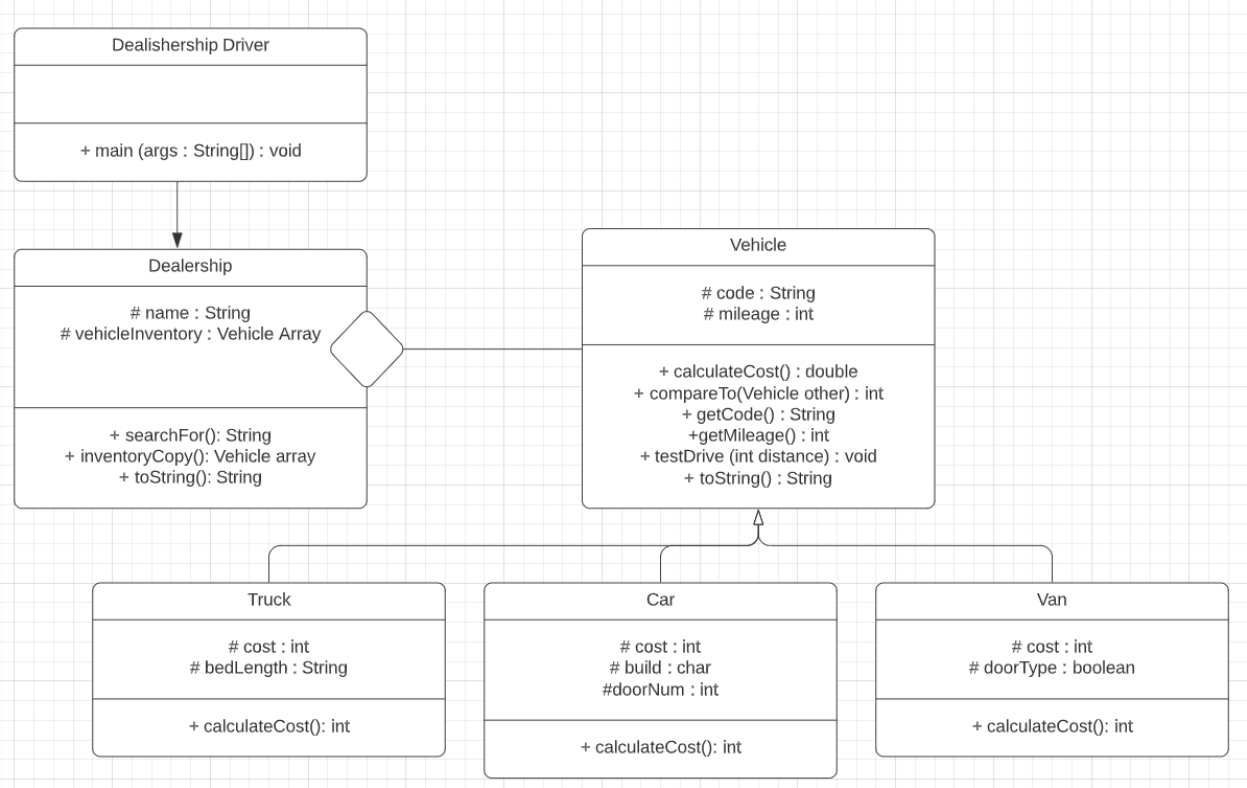
CS1083

Ms. Leah Bidlake

October 14, 2020

Kisenge Mbaga/3680552

1.



2.

Vehicle.java

```
/**
This class represents a Vehicle.

@author Kisenge Mbaga
*/

import java.util.Scanner;
import java.text.NumberFormat;
abstract class Vehicle{

        /**
        The vehicle code.
```

```
*/
private String code;
/**
The mileage of the vehicle
*/
private int mileage;
NumberFormat costFormat = NumberFormat.getCurrencyInstance();



/**
Constructs a Vehicle object with a vehicle code and mileage.
@param codeIn The vehicle code
@param mileageIn The vehicles mileage.
*/
public Vehicle( String codeIn, int mileageIn){
        code=codeIn;
        mileage=mileageIn;
}

public abstract double calculateCost();

/*public int compareTo(Vehicle other){
        if(this.code.compareTo(other.getCode())==0){

        }

}*/



/**
 Returns the vehicle code
@return The vehicle code.
*/
public String getCode(){
        return code;
}

/**
 Returns the vehicle mileage
@return The vehicle mileage.
*/
public int getMileage(){
        return mileage;
}
```

```
	/**
	 Returns whether the car was found.
	@param distance The distance the vehicle has driven.
	*/
	public void testDrive(int distance){
		mileage+=distance;
	}


	/**
	 Returns the vehicle code, mileage and cost in a string
	@return A string of the vehicle code, mileage and cost.
	*/
	public String toString(){
		String output="";
		output+=("\n"+code+"\tMileage: "+mileage+"km\n");
		output+=("\t\tCost: "+costFormat.format(this.calculateCost()));


		return output;
	}

	}
```

Car.java

```
/**
This class represents a Car.

@author Kisenge Mbaga
*/
public class Car extends Vehicle{

	/**
	Base cost of the car.
	*/
	private double cost= 10000;
	/**
	Number of doors.
	*/
	private int doorNum=2;
	/**
	The type indiciating whether hatchback or trunk.
	*/
	private char build;
```

```
	/**
	Constructs a Vehicle object with a vehicle code and mileage.
	@param codeIn The vehicle code
	@param mileageIn The vehicles mileage.
	@param doorNumIn The number of doors.
	@param buildIn The type of build. Hatchback or trunk.
	*/
	public Car( String codeIn, int mileageIn, int doorNumIn, char buildIn){
		super(codeIn, mileageIn);
		doorNum= doorNumIn;
		build= buildIn;
	}


	/**
	 Returns the cost of the vehicle based on the number of doors
	 and the type.
	@return The cost of the vehicle.
	*/
	public double calculateCost(){

	if (doorNum==4){
		cost+= 0.05*(cost);
	}

	if (build=='H'){
		cost+= 1000;
	}

	return cost;

	}

}
```

Truck.java

```
/**
This class represents a Truck.

@author Kisenge Mbaga
*/
public class Truck extends Vehicle{

	/**
	The type indiciating whether hatchback or trunk.
	*/
```

```
	private String bedLength;

	/**
	The base cost of a truck.
	*/
	private double cost= 50000;


	/**
	Constructs a Vehicle object with a vehicle code and mileage.
	@param codeIn The vehicle code
	@param mileageIn The vehicles mileage.
	@param bedLenthIn The type of truck length
	*/
	public Truck( String codeIn, int mileageIn, String bedLengthIn){
		super(codeIn, mileageIn);
		bedLength= bedLengthIn;
	}

	/**
	 Returns the cost of the vehicle based on type of truck bed.
	@return The cost of the vehicle.
	*/
	public double calculateCost(){

	if (bedLength.equals("short")){
		cost-= 0.1*(cost);
	}

	if (bedLength.equals("long")){
		cost+= 0.1*(cost);
	}

	return cost;

	}


}
```

Van.java

```
/**
This class represents a Van.

@author Kisenge Mbaga
*/
public class Van extends Vehicle{


	/**
	Base cost of the truck.
	*/
	private double cost= 25000;

	/**
	Boolean indicating whether door is electric or not.
	*/
	private boolean doorType= false;


	/**
	Constructs a Vehicle object with a vehicle code and mileage.
	@param codeIn The vehicle code
	@param mileageIn The vehicles mileage.
	@param doorTypeIn The type of door. Manual or electric.
	*/
	public Van( String codeIn, int mileageIn, boolean doorTypeIn){
		super(codeIn, mileageIn);
		doorType= doorTypeIn;
	}


	/**
	 Returns the cost of the vehicle based on the door type.
	@return The cost of the vehicle.
	*/
	public double calculateCost(){

	if (doorType==true){
		cost+= 0.15*(cost);
	}


	return cost;

	}
```

}

Dealership.java

```
/**
This class represents a Dealership.

@author Kisenge Mbaga
*/
import java.util.Scanner;

public class Dealership{

	/**
	Name of dealership.
	*/
	private String name;
	/**
	Array of vehicles
	*/
	private Vehicle [] vehicleInventory;
	/**
	Number of vehicles dealership.
	*/
	private int vehicleNum;
	Scanner sc= new Scanner(System.in);



	private String extraInfo;
	private String vCode;
	private String in;


	/**
	Constructs a Dealership object by reading the name of the dealership, and
	reading in a String array of vehicle information. It creates vehicles and puts them
	into its Vehicle array.
	@param nameIn The dealership name
	@param vehicleInventoryIn An array with all vehicle information.
	*/
	public Dealership(String nameIn, String[] vehicleInventoryIn){
		name= nameIn;


		for(int i=0; i<vehicleNum; i++){ //convert String Array to Vehicle Array
```

```
                String in= vehicleInventoryIn[i];
                Scanner sc2= new Scanner(in);
                String vCode= sc2.next();
                int mileage= Integer.parseInt(sc2.next());


                Scanner sc3= new Scanner(vCode);
                char vType= vCode.charAt(0);


                if(vType=='C'){
                        String extraInfo= sc2.next() + sc2.next();
                        Scanner sc4= new Scanner(extraInfo);
                        int doorNum= sc4.nextInt();
                        char build= sc4.next().charAt(0);
                        Car car1= new Car(vCode, mileage, doorNum, build);
                        vehicleInventory[i] = car1;
                }

                if(vType=='T'){
                        String extraInfo= sc2.next();
                        String bedLength= extraInfo;
                        Truck truck1= new Truck(vCode, mileage, bedLength);
                        vehicleInventory[i] = truck1;
                }

                if(vType=='V'){
                        String extraInfo= sc2.next();
                        boolean doorType= Boolean.parseBoolean(extraInfo);
                        Van van1= new Van(vCode, mileage, doorType);
                        vehicleInventory[i] = van1;
                }

        }


        }
```

```
/**
 Returns whether the car was found.
@param codeIn The code being searched for
@return A string indicating whether the vehicle was found or not.
*/
public String searchFor(String codeIn){
```

```
		String result="";
		for(int i=0; i<vehicleInventory.length; i++){
			Vehicle ref= vehicleInventory[i];
			if(ref.getCode().equals(codeIn)){
				result="Vehicle found at dealership.";
			}
			else{
				result="Vehicle not found at dealership.";
			}
		}

		return result;
	}


	/**
	 Returns a copy of the array of vehicles.
	@return A vehicle Array with the dealership vehicles
	*/
	public Vehicle[] inventoryCopy(){
		return vehicleInventory;

	}

	/**
	 Returns a string with the dealership name and vehicle information.
	@return A string
	*/
	public String toString(){
		String output="";
		output+=(name+"\n");
		for(int i=0; i>vehicleInventory.length; i++){
			Vehicle vehicleIn= vehicleInventory[i];
			output+= vehicleIn.toString();
		}

		return output;
	}



}
```

dealershipDriver.java

```
import java.util.Scanner;

public class DealershipDriver{
	public static void main(String[] args){

	Scanner sc = new Scanner(System.in);


	String name= sc.nextLine();
	int vehicleNum= sc.nextInt();
	int counter=0;


	//scan each vehicle record into a String array
	String [] vehicleInventory= new String[vehicleNum];
	for (int i=0; i<vehicleInventory.length; i++){
		vehicleInventory[i]=sc.nextLine();
	}


	//scan the Vcodes to search for into an array
	boolean check= true;
	while(check=true){
		if (sc.hasNextLine()){
			counter++;
		}
		else{
			check=false;
		}
	}
	String [] testVCode= new String[counter];
	for(int i=0; i<testVCode.length; i++){
		testVCode[i]=sc.nextLine();
	}



	//create a dealership object
	Dealership honda= new Dealership(name,vehicleInventory);
	System.out.print(honda.toString());
	// System.out.print(selectionSort((honda.inventoryCopy())));


	//search for the vehicle
	for(int i=0; i<vehicleInventory.length; i++){
		String vCode;
```

```
            vCode=vehicleInventory[i];
            honda.searchFor(vCode);
        }



        }
}
```

3. Output

Was unable to fix problem to test. The driver and all other classes compiled but the driver couldn't run. Aswell, I commented out the compareTo() method as I was unable to get it.