Lab - 10

Task: Write Java classes to represent a vehicle system.

First download the tester file *FleetOfVehicles.java* and put it in your project.

- DO NOT ALTER THE TESTER!

Write a Java class called **Vehicle**

- Some of the attributes are
  - Manufacturer's name
  - Number of Cylinders (must be greater than 0)
  - Owner's name
- Create the following Constructors
  - Default – sets everything to default values
  - Parameterized Constructor
    - Check for valid values
- Accessors and Mutators for each variable
  - MAKE SURE THE MUTATORS CHECK FOR VALID VALUES!
- Create the following Methods
  - equals – takes in another instance of a Vehicle and returns true only if all instance variables match
  - toString – returns a String that contains the Manufacturer's name, number of cylinders, and the owner's name

Write another Java class **Truck** which is a **Vehicle**

- Some of the attributes of Truck are
    - Load capacity: a nonnegative number of tons represented by a decimal number
    - Towing capacity: a nonnegative number of tons represented by a decimal
- Create the following constructors
    - Default – sets everything to default values
        - This includes calling the Vehicle's default constructor
    - Parameterized Constructor
        - This must also take in via parameter the manufacturer's name, number of cylinders, and the owner's name in addition to the load and towing capacity.
- Accessors and Mutators for each variable
    - MAKE SURE THE MUTATORS CHECK FOR VALID VALUES!
- Create the following methods
    - equals – takes in another instance of a Truck and returns true only if all the instance variables of vehicle and truck match.
    - toString – This should also override the vehicle's toString method and also return the Vehicle's toString along with the towing and load capacity

Write another class **Car** which is a **Vehicle**

- Some of the attributes of Car are
    - Gas Mileage: a nonnegative number of gallons represented by a decimal number

    - Number of passengers: a nonnegative number of passengers represented by a whole number

- Create the following constructors
    - Default – sets everything to default values
        - This includes calling the Vehicle’s default constructor
    - Parameterized Constructor
        - This must also take in via parameter the manufacturer’s name, number of cylinders, and the owner’s name in addition to the load and towing capacity.
- Accessors and Mutators for each variable
    - MAKE SURE THE MUTATORS CHECK FOR VALID VALUES!
- Create the following methods
    - equals – takes in another instance of a Car and returns true only if all the instance variables of vehicle and car match.
    - toString – This should also override the vehicle’s toString method and also return the Vehicle’s toString along with the gas mileage and number of passengers

Sample Output:

```
Welcome to the fleet manager

Enter 1: to add a Vehicle

Enter 2: to remove a Vehicle

Enter 9 to quit

1
```

```
Enter 1: if it is a car

Enter 2: if it is a truck

Enter 3: if it is unclassified

1

Enter the manufacturer's name

Nissan

Enter the number of cylinders

6

Enter the owner's name

Mark

Enter the car's gas mileage

29

Enter the number of passengers

5

The Fleet currently

Manufacturer's Name: Nissan

Number Of Cylinders: 6

Owner's Name: Mark

Gas Mileage: 29.0 gallons

Number of Passengers: 5
```

```
Enter 1: to add a Vehicle

Enter 2: to remove a Vehicle

Enter 9 to quit

1

Enter 1: if it is a car

Enter 2: if it is a truck

Enter 3: if it is unclassified

2

Enter the manufacturer's name

Chevy

Enter the number of cylinders

8

Enter the owner's name

Eddie

Enter the truck's load capacity

1

Enter the truck's towing capacity

2

The Fleet currently
```

```
Manufacturer's Name: Nissan

Number Of Cylinders: 6

Owner's Name: Mark

Gas Mileage: 29.0 gallons

Number of Passengers: 5


Manufacturer's Name: Chevy

Number Of Cylinders: 8

Owner's Name: Eddie

Load Capacity: 1.0

Towing Capacity: 2.0


Enter 1: to add a Vehicle

Enter 2: to remove a Vehicle

Enter 9 to quit

1

Enter 1: if it is a car

Enter 2: if it is a truck

Enter 3: if it is unclassified

3
```

```
Enter the manufacturer's name

Ford

Enter the number of cylinders

6

Enter the owner's name

Bob

The Fleet currently

Manufacturer's Name: Nissan

Number Of Cylinders: 6

Owner's Name: Mark

Gas Mileage: 29.0 gallons

Number of Passengers: 5



Manufacturer's Name: Chevy

Number Of Cylinders: 8

Owner's Name: Eddie

Load Capacity: 1.0

Towing Capacity: 2.0



Manufacturer's Name: Ford
```

```
Number Of Cylinders: 6
Owner's Name: Bob

Enter 1: to add a Vehicle
Enter 2: to remove a Vehicle
Enter 9 to quit
2
Enter 1: if it is a car
Enter 2: if it is a truck
Enter 3: if it is unclassified
2
Enter the manufacturer's name
Chevy
Enter the number of cylinders
8
Enter the owner's name
Eddie
Enter the truck's load capacity
1
Enter the truck's towing capacity
```

```
2
The Fleet currently
Manufacturer's Name: Nissan
Number Of Cylinders: 6
Owner's Name: Mark
Gas Mileage: 29.0 gallons
Number of Passengers: 5


Manufacturer's Name: Ford
Number Of Cylinders: 6
Owner's Name: Bob


Enter 1: to add a Vehicle
Enter 2: to remove a Vehicle
Enter 9 to quit
9
The Fleet currently
Manufacturer's Name: Nissan
Number Of Cylinders: 6
Owner's Name: Mark
```

```
Gas Mileage: 29.0 gallons

Number of Passengers: 5



Manufacturer's Name: Ford

Number Of Cylinders: 6

Owner's Name: Bob

Goodbye!
```

Lab Submission:

- At the beginning of your program, insert your full name as a comment.
- Include comments in your program wherever necessary.
- Upload all .java files on Dropbox.

Lab Report Submission:

- First, download the Lab report Template document on Dropbox.
- Use this template to complete your lab report.
- Additional Questions:

  1. Draw a UML Diagram to represent the classes *Vehicle, Truck* and *Car* in the proposed solution section.

  2. Explain the difference between method overloading and method overriding?