CSCI-C212: Assignment 5 - Tom's Automobile Adventure

Purpose:

This assignment will help you understand and apply the concepts of inheritance, polymorphism, composition, method overloading, and method overriding in Java. You are required to write a program that implements a small system demonstrating these concepts.

Description:

Tom is an automobile enthusiast who collects a wide variety of vehicles in his garage. He loves everything from sports cars to trucks and bikes. Tom has a deep understanding of how these vehicles work and enjoys categorizing and organizing them.

In his programming journey, Tom decides to create a system that will help him manage his collection using object-oriented programming principles in Java. He wants to use **inheritance**, **polymorphism**, and **composition** to model the relationships between different types of vehicles. Tom maintains a garage to keep track of all his vehicles using an array. The garage will store different types of vehicles, and Tom can perform operations like starting all vehicles or refueling them.

Your Task

Help Tom by writing a Java program that models his collection. You will be required to:

1. **Create a base class** called `Vehicle`, which represents any vehicle in general. It will have properties such as the vehicle's make, model, year, and fuel capacity. You will define a few **methods** that apply to all vehicles, such as `start()`, `stop()`, and `refuel()`. Add other properties and methods as you feel necessary.
2. **Composition**: Some vehicles, like trucks, have engines that are **composed** of smaller parts, such as cylinders and pistons. Add an `Engine` class that will represent the engine of a vehicle. This will demonstrate the concept of composition, where the `Vehicle` class will have an `Engine` object as one of its properties. Add properties and methods as you feel necessary.
3. **Inheritance**: Tom has different types of vehicles like **cars** and **bikes**. Extend the `Vehicle` class to create specialized subclasses like `Car`, `Truck`, and `Bike`. Each subclass should have unique properties or methods that differentiate them from the `Vehicle` super class. For example, a truck may have a method for `loadCargo()`.
4. **Method Overriding**: In some cases, the `start()` method of a bike might be different from that of a car. You need to **override** the `start()` method in the appropriate subclasses to reflect this.
5. **Method Overloading**: Tom wants to have different ways to **refuel** his vehicles. Sometimes, he may refuel by specifying a specific amount of fuel, and at other times, he may simply want to fully fill the tank. Use **method overloading** to give the `refuel()` method multiple forms, depending on how much fuel is being added.

6. **Polymorphism**: Tom should be able to interact with all his vehicles in a uniform way. For example, he should be able to call `start()` on any vehicle, whether it is a car, a bike, or a truck. Use **polymorphism** to allow Tom to treat all vehicles uniformly while keeping their individual behaviors intact.
7. **The Main class**: In the main class, there will be an array of vehicles to store 100 vehicles. The program will ask Tom if he wants to add any vehicle. If Tom responses as 'Yes', the program will ask different information regarding the vehicle and add it to the array. The program will continue adding the vehicles until Tom responses "No". Then Tom will be able to execute different operations on all the vehicles polymorphically.

Output Requirements:

- The program must demonstrate all the methods declared in different classes.
- You must demonstrate the use of inheritance, method overriding, method overloading, and composition in your solution.

Submission Instructions:

- Create a Java project using an IDE, such as NetBeans, Eclipse, or any other IDE that you like.
- Include comments explaining how inheritance, polymorphism, method overloading, and composition are used in your program.
- Provide screenshots of example input and output to show how your program works.
- Include all your work in a single zip file and submit that.

Sample:

```
Welcome to Tom's Garage!

Enter details for a new vehicle.
Enter vehicle type (Car/Truck/Bike): Car
Enter make: Toyota
Enter model: Camry
Enter year: 2025
Enter fuel capacity: 14
Do you want to enter another vehicle? (yes/no): yes

Enter details for a new vehicle.
Enter vehicle type (Car/Truck/Bike): Truck
Enter make: Ford
Enter model: F-350
Enter year: 2015
Enter fuel capacity: 20
Do you want to enter another vehicle? (yes/no): yes

Enter details for a new vehicle.
Enter vehicle type (Car/Truck/Bike): Bike
Enter make: Kawasaki
Enter model: Ninja
Enter year: 2020
Enter fuel capacity: 8
Do you want to enter another vehicle? (yes/no): no

--- Vehicle Information in Tom's Garage ---

---Vehicle #1---
Make: Toyota
Model: Camry
Year: 2025
Engine Type: 6 Cylinder Engine

---Vehicle #2---
Make: Ford
Model: F-350
Year: 2015
Engine Type: 8 Cylinder Engine

---Vehicle #3---
Make: Kawasaki
Model: Ninja
Year: 2020
Engine Type: 2 Cylinder Engine


---Starting All Vehicles---
Car starts smoothly.
Toyota Camry starts.
Engine with 6 cylinders starts.

Truck  starts with a heavy roar.
Ford F-350 starts.
Engine with 8 cylinders starts.

Bike  starts with a vroom.
Kawasaki Ninja starts.
Engine with 2 cylinders starts.

---Refueling All Vehicles---
Toyota Camry Refueling done - Fuel tank full.
Ford F-350 Refueling done - Fuel tank full.
Kawasaki Ninja Refueling done - Fuel tank full.

---Stopping All Vehicles---

Toyota Camry stopped.
Ford F-350 stopped.
Kawasaki Ninja stopped.
```

Good luck helping Tom manage his amazing collection of vehicles!

Useful links:

https://www.javatpoint.com/how-to-create-array-of-objects-in-java

https://www.tutorialspoint.com/java/java_polymorphism.htm