1. **Create an abstract class named Vehicle with an abstract method startEngine(). Add an attribute String model.**
2. **Add a constructor to the Vehicle class that takes the model attribute as a parameter. Ensure subclasses should call this constructor.**
3. **Create a method showInfo() in the Vehicle class that prints the model of the vehicle. This method should be inherited by all subclasses.**
4. **Create a class Car that extends Vehicle. Implement the startEngine() method in Car and display the car's model when the engine starts.**
5. **Create a class Motorcycle that extends Vehicle. Implement the startEngine() method in Motorcycle and display the motorcycle's model when the engine starts.**
6. **Create a class Truck that extends Vehicle. Implement the startEngine() method in Truck and display the truck's model when the engine starts.**
7. **In the Main class, create an array that can store four Vehicle objects: two cars, one motorcycle, and one truck.**
8. **Initialize the array with the appropriate vehicle objects and iterate over the array to call startEngine() and showInfo() for each vehicle.**

**Corresponding Code:**

java

// Step 1, 2: Abstract class Vehicle with an attribute and constructor

abstract class Vehicle {

protected String model;

// Constructor

public Vehicle(String model) {

this.model = model;

}

// Abstract method that must be implemented by subclasses

public abstract void startEngine();

// Method to show vehicle information

public void showInfo() {

System.out.println("Vehicle model: " + model);

}

}

// Class Car extending Vehicle with constructor and overriding startEngine()

class Car extends Vehicle {

public Car(String model) {

super(model); // Call the parent constructor

}

@Override

public void startEngine() {

System.out.println("Starting car's engine. Model: " + model);

}

}

// Class Motorcycle extending Vehicle with constructor and overriding startEngine()

class Motorcycle extends Vehicle {

public Motorcycle(String model) {

super(model); // Call the parent constructor

}

@Override

public void startEngine() {

System.out.println("Starting motorcycle's engine. Model: " + model);

}

}

// Class Truck extending Vehicle with constructor and overriding startEngine()

class Truck extends Vehicle {

public Truck(String model) {

super(model); // Call the parent constructor

}

@Override

public void startEngine() {

System.out.println("Starting truck's engine. Model: " + model);

}

}

// Main class demonstrating polymorphism, arrays, constructors, and attributes

public class Main {

public static void main(String[] args) {

// Step 7: Create an array of Vehicle objects (2 Cars, 1 Motorcycle, 1 Truck)

Vehicle[] vehicles = new Vehicle[4];

// Initializing the array

vehicles[0] = new Car("Toyota Camry");

vehicles[1] = new Car("Honda Civic");

vehicles[2] = new Motorcycle("Harley Davidson");

vehicles[3] = new Truck("Ford F-150");

// Step 8: Iterate over the array and call startEngine() and showInfo() for each vehicle

for (Vehicle vehicle : vehicles) {

vehicle.startEngine();

vehicle.showInfo();

}

}

}