Java Fundamentals 7-3: The Static Modifier and Nested Classes Practice Activities

Vocabulary Definitions

1. **Static Method**: Is a method that is available for use without first creating an instance of the class. It is declared by preceding its definition with the static modifier.
2. **Nested Class**: Is any class implemented as a nested class within another class. By definition, all inner classes are members of the container class by composition.
3. **Static Variable**: Any Java class-level variable that is declared with the static modifier. This means only one instance of the class variable can exist in the JVM regardless of the number of class instances.
4. **Static Keyword**: Is a keyword that makes a variable, method, or inner class available without first creating an instance of a variable.
5. **Inner Class**: Is an inner class. Inner classes are defined within a parent or container class and are members of the container class by composition. In fact, inner classes are the only way you can create class instances through composition.
6. **Static Inner Class**: Is an inner class that is available for use without first creating an instance of the container class. It is declared by preceding its definition with the static modifier.
7. **Class Method**: Any Java method defined with a static modifier. It is accessible outside the class when a public, protected, or default access specifier precedes it. It is private and inaccessible outside of the class when a private specifier precedes it. Class methods are available without first creating an instance of the class.
8. **Static Variable**: Is a variable that may be available outside of a class without first creating an instance of a class. It is declared by preceding the variable name with the static modifier.

Tasks

Task 1: Creating the Vehicle Class

```java
package vehicles;
public class Vehicle {
    public static String MAKE = "Augur";
    public static int numVehicles = 0;
    private String chassisNo;
    private String model;
    public Vehicle(String model) {
        numVehicles++;
        this.chassisNo = "ch" + numVehicles;
        this.model = model;
        System.out.println("Vehicle manufactured");
    }
    public String getChassisNo() {
        return chassisNo;
    }
    public void setChassisNo(String chassisNo) {
        this.chassisNo = chassisNo;
    }
    public String getModel() {
        return model;
    }
    public void setModel(String model) {
        this.model = model;
    }
    @Override
    public String toString() {
        return "The vehicle is manufactured by: " + MAKE + "\n" +
                "The model type is " + model + "\n" +
                "The chassis number is " + chassisNo;
    }
    public static class Engine {
        private static final String MAKE = "Predicter";
        private static final int CAPACITY = 1600;
        public static String getMake() {
            return MAKE;}
        public static int getCapacity() {
            return CAPACITY;
```

Task 2: Creating the TestVehicle Class

Main.java

```java
package vehicles;

public class TestVehicle {
    public static void main(String[] args) {
        // Output static variables
        System.out.println("Manufacturer: " + Vehicle.MAKE);
        System.out.println("Number of vehicles manufactured: " + Vehicle.numVehicles);

        // Create first vehicle
        Vehicle vehicle1 = new Vehicle("Vision");
        System.out.println(vehicle1.toString());

        // Create second vehicle
        Vehicle vehicle2 = new Vehicle("Edict");
        System.out.println(vehicle2.toString());

        // Display the total number of cars manufactured
        System.out.println("Number of vehicles manufactured: " + Vehicle.numVehicles);

        // Modify the MAKE using one of the instances
        vehicle2.MAKE = "Seer";

        // Display updated details
        System.out.println(vehicle1.toString());
        System.out.println(vehicle2.toString());

        // Create an Engine instance through nested static class
        Vehicle.Engine vehicle3 = new Vehicle.Engine();
        System.out.println("Vehicle number ch3 is a Fortune model and has an engine capacity of " + Vehicle
            .Engine.getCapacity() + "cc");
    }
}
```

Task Outputs

The expected outputs after running the provided Java classes are as follows:

Task 1 Output

plaintext Copy code

```plaintext
Vehicle manufactured
The vehicle is manufactured by: Augur
The model type is Vision
The chassis number is ch1
Vehicle manufactured
The vehicle is manufactured by: Augur
The model type is Edict
The chassis number is ch2
Number of vehicles manufactured: 2
The vehicle is manufactured by: Seer
The model type is Vision
The chassis number is ch1
The vehicle is manufactured by: Seer
The model type is Edict
The chassis number is ch2
Number of vehicles manufactured: 2
Vehicle number ch3 is a Fortune model and has an engine capacity of 1600cc
```