Java Static

Lab setup

We will continue with our Bicycle example.

1. Create a new Java project and call it "`JavaStatic`".
2. Add the Bicycle class to the project and add a new field `private int serialNo`
3. Add a new constructor to Bicycle which accepts only a parameter for the `serialNo`
4. Add the BicycleFactory class in Figure 1 to the project.

```
class BicycleFactory{

    private int serialNo;

    public BicycleFactory(){
        serialNo = 0;
    }

    public Bicycle createBicycle(){
        return new Bicycle(serialNo++);
    }
}
```

Figure 1 BicycleFactory class

Instance methods and fields

Instance methods and fields require an object of a class to access. There is no keyword which specifies an instance method or field. A method/field always requires an object of its class to access unless it is labeled `static`. Instance methods can be overridden, something which will be discussed in a later lab. Instance methods can access all methods/fields in a class.

Graded submission Task 1

1. Add the code for the following to the driver:
    - Create an object of the `BicycleFactory` class
    - Use the `BicycleFactory` method `CreateBicycle` to create two Bicycle objects
        - Print `serialNo` for each Bicycle
    - Repeat the above steps for a different `BicycleFactory` object

You should notice that each BicycleFactory will create bicycles starting from the same serial number. These bicycles are not the same objects but can be considered equal.

** STOP **

Ensure you have saved the source files in your project. Make a copy of your Eclipse project at this point (you can do this from your system's file explorer). Name the file labelling it as Task 1. You will submit this copy at the end of this lab.

Static methods and fields

Static methods and fields do not require an object of a class to access. They are accessed by using the name of the class. These methods/fields are labeled using the `static` keyword. Unlike instance methods/fields, static methods/fields are resolved at compile time. Because of this, static methods cannot be overridden. Static fields are shared with every instance of a class. Static methods cannot access instance methods/fields without an object of the class. Static methods also cannot use the keyword `this`.

We want the bicycle serial number to increment whenever a bicycle is created even if they are created by different factories.

Graded submission Task 2

1. Add code for the following in BicycleFactory:
    - Static field for the next serial number
        - Should be initialized to 0.
    - Static method to get the next serial number.
        - Should not increment the serial number.
    - Change CreateBicycle so that after it creates a new Bicycle, it increments the next serialNo by 1.
2. In the driver, between calls to the CreateBicycle method, print the next serialNo using the static method and confirm every Bicycle is created with a unique serialNo.

** STOP **

Ensure you have saved the source files in your project. Make a copy of your Eclipse project at this point (you can do this from your system's file explorer). Name the file labelling it as Task 2. You will submit this copy at the end of this lab.

Drone application

We would like to track drones from different drone depots as unique entities for the purposes of logging at the end of a simulation run.

Graded Submission Task 3

Modify the DroneDepot class to use a static field for the next drone ID.

Write a driver which shows that this works as intended.

In particular,

- The next drone ID should increment by one when a new drone is deployed.
- The IDs of newer drones should be greater than older drones regardless of which depot deployed them.
- Consider if you can verify this without manually inspecting the IDs of every drone.

** STOP **

Ensure you have saved the source files in your project. Make a copy of your Eclipse project at this point (you can do this from your system's file explorer). Name the file labelling it as Task 3. You will submit this copy at the end of this lab.

Lab wrap-up

In this lab we discussed the keyword static. Most likely, we are most familiar with instance methods and fields. Instance methods and fields require an object of a class to access. Anything that is not labeled static requires an object to access. When the keyword static is used, an object is no longer required. Static methods and fields are resolved at compile time, and they are shared with every instance of a class. We illustrated this using a static field to determine the next ID a created Bicycle should have.

Static methods cannot access non-static methods and fields or this because there is no instance for a static method to work with. Instance methods can access anything within a class.

Submission notes for graded tasks:

- Submit a zipped folder containing the Eclipse projects for each graded task.
- For each project, ensure you have included the src folder containing the .java file, the .classpath file, and the .project file.
- Ensure that each project is named appropriately to correspond with each task.
- Zip the projects together and name the zipped folder "Lab7_firstName_lastName".
- Submit the zipped file to the respective Canvas link.

Grading Rubrics		
Task 1 (points allotted) - Compilation (2) - Create more than one Bicycle from one BicyleFactory (2) - Create more than one BicycleFactory (2) - Demonstrate equality of serialNo between Bicycles from different BicycleFactories (4)	Task 2 (points allotted) - Compilation (2) - Static field for next serialNo (2) - Static method for getting next serialNo (2) - Demonstrates serialNo increases by one whenever a Bicycle is created from any BicycleFactory (4)	Task 3 (points allotted) - Compilation (2) - Static field in DroneDepot for next drone ID (2) - All drones have unique IDs regardless of the DroneDepot that deployed it (6)