Java Access Modifiers

Lab setup

1. Create a new Eclipse project.
2. Add the Bicycle class to the project.
3. Add a main class which we can use as a driver.

Access modifiers

Access modifiers specify the scope/accessibility of a field, method, constructor, or class.

The four access modifiers are the following:

Private
Accessible only within class

package-private
Accessible only within package

Protected
Accessible only through a child class (*note that the* ***child*** *class and* ***parent*** *class could be in the same package or in two different packages*)

Public
Accessible everywhere

package-private

If one of the other modifiers is not specified, then it will default to package-private. This means the class, method, or field cannot be accessed outside the package.

We can demonstrate this by adding Bicycle and Main to different packages.

1. Add the line `package mainPackage;` to the top of the Main.java file.

2. Add the line `package bicyclePackage;` to the top of the Bicycle.java file.

3. Eclipse will probably throw an error about the package not matching. Accept the first option Eclipse suggests to fix both errors. It will create the required packages and add the classes to the appropriate one.

4. Declare a Bicycle variable inside the driver.

Eclipse will show an error: `Bicycle cannot be resolved to a type`. This is because the compiler cannot find a definition for a class Bicycle. We have placed it inside bicyclePackage so we need to use an `import` statement to get its definition.

5. Add the following line to the beginning of Main.java:

   ```
   import bicyclePackage.Bicycle;
   ```

   The error should now be different: `The type Bicycle is not visible`. This is due to the default access modifier.

6. Make Bicycle public and the error should now be gone. We can only work with references of Bicycle, however, and we still cannot access any of the fields or methods.

Public and Private

At this point we have already made the class public. Try to create a Bicycle object using the `new` operator and you should get a familiar error, `the constructor Bicycle(int,int,int)` is not visible.

Go ahead and fix the cause of this error.

Graded Submission Task 1:

Make the following changes to the Bicycle class:
- Change all fields to be private.
- Add public getters and setters to access the private fields.

** 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.

Protected

Protected can only be applied to fields, methods, and constructors. It cannot be applied to a class. Protected fields, methods, and constructors can be accessed within a package, and can only be accessed outside of a package through a subclass.

```java
package mainPackage;
import bicyclePackage.Bicycle;
public class SpeedBicycle extends Bicycle {

    public SpeedBicycle(int cadence, int speed, int gear) {
        super(cadence, speed, gear);
    }

}
```

Figure 1 SpeedBicycle class in mainPackage

Graded Submission Task 2:

1. Change the `speedUp()` method to protected, then confirm for yourself that you cannot call this method with a Bicycle object in main.
2. Add a new class to the project call it SpeedBicycle.

   This class will be a child of the Bicycle class and be added to mainPackage. You can set all of this up through the class wizard, or you can create a SpeedBicycle class as normal then change the code to look as shown in Figure 1.

   Note that SpeedBicycle is not in the same package as Bicycle.

3. Create a SpeedBicycle object in the main method and try to call `speedUp()` by accessing it through the SpeedBicycle object.

** 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.

Which access modifier should be used?

The choice of access modifier depends on design. The most common you will find is that fields are generally private. Constructors and methods are generally public.

Methods can be private if they perform a function which is not necessary outside of the class. Methods can be protected in a base class if its children should have access to those methods, or it is not expected for the method to be used outside of the package it belongs to.

Constructors can be protected to prevent instantiation of a class outside of the package it belongs to. They can also be private in what is known as a singleton pattern, only one instance of the class can ever exist.

Classes can be package-private if they are not used or should not be exposed outside of the package it belongs to.

Drone Application

You have been given another driver that has worked with aspects of each class as either public or default. Run this driver once to see its output.

Graded Submission Task 3:

Make the appropriate access modifier changes to the classes you have written for the simulation, most importantly change all fields to private.

Add any additional functionality to the classes to get the same output from the driver now that the fields are private.

** 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 access modifiers (in order of level of visibility) private, package-private, protected, and public. The package-private modifier is applied when none of the other modifiers are specified. When a class, method, or field is package-private, it cannot be accessed outside of the package it resides in. When a class (nested class), method, or field is declared to be private it is only accessible in the class it resides in. The protected modifier can only be applied to methods or fields. Protected methods or fields can be accessed within the same package. They can also be accessed outside of a package, but only through inheritance i.e., some outside class must extend the class with the protected method or field. Public is the least restrictive access modifier. Classes, methods, or fields declared public can be accessed anywhere.

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 “Lab6_firstName_lastName”.
- Submit the zipped file to the respective Canvas link.

Rubrics		
Task 1 (allotted points) - Compilation (2) - All fields private (4) - Getters implemented for each field (2) - Setters implemented for each field (2)	Task 2 (allotted points) - Compilation (2) - Bicycle method speedUp() is protected (2) - SpeedBicycle class extends Bicycle (2) - SpeedBicycle in different package than Bicycle (2) - Driver shows SpeedBicycle object can call speedUp() (2)	Task 3 (allotted points) - Compilation (2) - All fields private (2) - Getters and setters implemented where needed (2) - Driver outputs as expected (4)