ITSC 1212 Introduction to Computer Science 1 - Lab 2

The point of this lab is for you to familiarize yourself with Java classes, objects and methods. You can use any lab or personal machine that has been set up with Java and DrJava.

OVERVIEW

1. Part A: Creating a new Java class
2. Part B – The main Method
3. Part C – Creating a class that uses another class

Part A – Creating a new Java class

1. Create a folder called Lab_2 inside the ITSC_1212 folder (remember this is the folder you downloaded in Lab 1).
2. In DrJava go to File → New to create a new file.
3. Type the code below replacing the text "Your Name" with your first and last name

```
/*
  Your Name
  Lab 2 - Hello World
  ITSC1212 - A Java Class
*/

class HelloWorld
{ //class body starts here
    public static void main(String[] args)
    { // method body starts here
       System.out.println("Hello World!");
    } //end of main method body
} //end of class definition
```

4. The commented lines at the top are very important; you should add them at the top of every program you write. The basic unit of a Java program is a class. A class called "HelloWorld" is defined via the keyword "class". The braces {.....} encloses the body of the class.
5. Save the file you have just created inside the Lab_2 folder you created in step 1. Make sure the file is called "HelloWorld.java".

Important Note: The filename you choose must be the same name as the class name. If you save it to a different name, the compiler will yell at you and not let you run the program.

6. Compile the program. If you receive any errors, you need to fix them. If all goes well, it will report no errors and generate a "HelloWorld.class" file in the same folder the HelloWorld.java is saved in.
7. Now run your program. What does the program write to the screen?

_____

8. Show your output to the professor or one of the TAs. If you are completing this on your own outside the lab take a screenshot of your screen and save the file as lab2_partA to include with your submission.

Part B – The main method

1. Download Circle.java from Module 1 in Canvas. (We will often provide code for you to start with.)
2. Save Circle.java in the Lab_2 folder. **Do not change the filename.** If it is not called Circle.java, you will have problems.

1) In DrJava go to File → Open. Navigate to your Lab_2 folder. Locate and double-click on Circle.java or select Circle.java and click Open.
2) Read through the Circle.java file. DON'T CHANGE ANYTHING. Compile "Circle.java" and Answer this question:
   Can you run the Circle class? Why?

   _____

   _____

   _____

This Circle class does not have a main() method. Classes without a main method cannot be run directly. The main method is the entry point of any java program. Its syntax is always

```
public static void main(String[] args)
```

You can only change the name of the String array argument, for example you can change `args` to `myStringArgs` or any valid identifier name. We will learn more about valid naming in Java next week. Programs that don't have a main method are often referred to as "building blocks". Later on, we will see more examples of how these building blocks can be used in other programs. For now, let's add a main method to the Circle class. Inside the Circle class body definition (the curly brace after the class name and at the end of the file encloses the body of the class) add the following:

```
   public static void main(String[] args) {
     //Declare an instance of Circle class called c1
     //Construct the instance c1 by invoking the "default" constructor
     //which sets its radius and color to their default value
     Circle c1 = new Circle();

     //Invoke (call) methods on instance c1, via dot operator
     double radius = c1.getRadius();
     double area = c1.calculateArea();

     //Display (print) information on the screen
     System.out.println("The circle has radius "
                        + radius + " and area of " + area);

     //Declare an instance of Circle class called c2
     //Construct the instance c2 by invoking the second constructor
     //which given radius and default color
     Circle c2 = new Circle(2.0);

     //Invoke (call) methods on instance c2, via dot operator
     double radius2 = c2.getRadius();
     double area2 = c2.calculateArea();

     //Display (print) information on the screen
     System.out.println("The circle has radius "
                        + radius2 + " and area of " + area2);
}
```

4. Now compile and run the Circle.java and study the results. What does the program write to the screen?

_____

5. Show your output to the professor or one of the TAs. If you are completing this on your own outside the lab take a screen and save the file as lab2_partB to include with your submission.

Part C – Creating a class that uses another class

In part B we created a main method inside the Circle class to be use that class and insatiate objects from it. As I mentioned earlier, classes are often written to define building blocks that can be used in various programs to create a solution.

1. Let us write a *test program* called Shapes (in another source file called Shapes.java) which uses the Circle class. Create a new file called Shapes.java inside the Lab_2 folder that contains the same code that is inside the main method. Make sure your code looks like the following:

```
public class Shapes{ //save file as Shapes.java

  public static void main(String[ ] args) {
     // Declare an instance of Circle class called c1
     // Construct the instance c1 by invoking the "default" constructor
     // which sets its radius and color to their default value
     Circle c1 = new Circle();

     // Invoke (call) methods on instance c1, via dot operator
     double radius  = c1.getRadius();
     double area =  c1.calculateArea();

     // Display (print) information on the screen
     System.out.println("The circle has radius of "
                          + radius + " and area of " + area);

     // Declare an instance of class circle called c2
     // Construct the instance c2 by invoking the second constructor
     // with the given radius and default color
     Circle c2 = new Circle(2.0);

     // Invoke (call) methods on instance c1, via dot operator
     double radius2  = c2.getRadius();
     double area2 =  c2.calculateArea();

     // Invoke public methods on instance c2, via dot operator
     // Display (print) information on the screen
     System.out.println("The circle has radius of "
                          + radius2 + " and area of " + area2);
  }

}
```

2. Save and compile Shapes. On no! This code won't compile. Why do you think this code did not compile?

_____

3. In part B we were able to reference the calculateArea( ) method inside of the main method because that method was in the same class. calculateArea( ) method has a private access control modifier. Think of private as like your diary. Only you should have direct access to it. In this case private means that only the code in this class can directly access this method. To be able to fix this we need to reference a public method from the Circle class that can give us the value of the area. Do you see a method in the Circle class that can be used instead?

4. The Circle class has a public method called getArea() that calls the private method calculateArea and is able to retrieve the value of the area calculated. Let's update our code to use this method instead. Change the two method calls to calculateArea() to call getArea() instead. These method calls should look like: c1.getArea() and c2.getArea()

5. Save and compile your updated Shapes class. If your code compiles with no errors run the program and compare the output to the output you received in part B. Is it the same?

6. You're done! Show your output to the professor or one of the TAs. If you are completing this on your own outside the lab take a screen and save the file as lab2_partC to include with your submission.

Don't forget to submit your work!

Lab2 Canvas submission

To submit the lab

1. Navigate to the course Canvas page.
2. Click on Module 2
3. In this module locate and click on the Lab 2 Completion assignment and submit the following files
    a. HelloWorld.java
    b. Circle.java
    c. Shapes.java
    d. Screenshots for parts A, B and C if you were not present in the lab when you completed them