Finals Laboratory Activity 4

Name: Abenes, Enrico O.

Subject Code & Schedule: CC1, 1: 30 pm – 5: 20 pm MT

Course and Section: BSIT, INTL1

TITLE: Java Methods

LEARNING OBJECTIVES:

At the end of this activity, the students should be able to:

1. Identify the types of methods in the Java programming language.

2. Identify what portions of the main program can be subdivided into sub-programs.

3. Create methods or sub-programs that can be called in the main program.

4. Distinguish the different parameters used in creating methods.

5. Create a complete Java program that utilizes methods.

INSTRUCTIONS:

1. Make sure you have your own individual account.

2. Always keep your account secret from others to avoid unauthorized access to your files.

3. Always save your work and log off when not using the computer.

4. By now you should have been familiarized with using your text editor.

5. By now you should know how to create, save, compile, execute, and debug programs in Java.

DURATION: Two to Three Meetings

HANDS-ON:

1. Log on using your own individual account. Use your own username and password.

2. Open your text editor.

3. Write your next Java program:

    3.1. Write your next program by copying the source code shown below to your text editor.

```
/* Programmed by: <write your name here>
   Program title: PrintHello.java
   Program Date: <write the date today here> */

public class PrintHello{
     public static void printHello(){
          System.out.print("HELLO");
     }
     public static void printSpace(){
          System.out.print(" ");
     }
     public static void printWorld(){
          System.out.println("WORLD!");
     }
     public static void allTogetherNow(){
          printHello();
          printSpace();
          printWorld();
     }

     public static void main(String[] args){
          AllTogetherNow();
     }
}
```

3.2. Save the program as PrintHello.java then compile your program until no errors and warnings are reported.

3.3. Run your program.

3.4. Simulate and write what will be displayed on the screen.

```
Hello WORLD!
```

3.5. Enumerate all user-defined methods:

printHello() – "Hello" is printed to the console using this way.

printSpace() – This technique prints the character "space" to the console.

printWorld() – "WORLD!" is printed to the console by this way.

allTogetherNow() – The printHello(), printSpace(), and printWorld() methods are sequentially invoked by this method.

3.6. Enumerate all pre-defined methods:

System,out print() – Text can be printed on the console using this technique.

String[] args – This is the input for the main method, a built-in method that acts as the starting point for Java programs.

> ***HINT:*** To call a method, indicate the name of the method preceded by the **()** symbol. User-define methods are methods you created , while pre-defined functions are functions already inherent to the compiler.

4. Is the method parameter always void? No

```
public class OneParam{
     public static void oneParam(int x){
          System.out.println("Number is " + x);
     }

     public static void main(String[] args){
          int x;
          x = 100;
          oneParam(x);
     }
}
```

4.1 To answer that, let's simulate the following program

4.2 Save your program as OneParam.java

4.3 Run and write what will be displayed on the screen?

```
Number is 100
```

4.4. Insert the statement oneParam(75);

What can you conclude? That is, is it possible to use a constant value as a parameter when a method is called?

When you wish to send a method a certain value that is known at compile time and doesn't need to be dynamically computed during runtime, using constant values as method parameters **might be helpful.**

4.5 Declare another integer variable in your main method as follows:

int y; Initialize the variable y to 150. Thereafter, write another statement OneParam(y);

What can you conclude? That is, is it possible to use a variable name that is not the same as that used in the method definition?

Using a variable name other than the one given in the method description is technically **conceivable**. As long as the types of the arguments and parameters match, various variable names can be used to pass values when invoking a method.

4.6 Insert the statement OneParam(x+y);

What can you conclude? That is, is it possible to use an expression (that will evaluate an integer value) as a parameter when the method is called?

When invoking a method, it is **possible** to pass an expression that evaluates to an integer value as a parameter.

4.7 Insert the statement OneParam();

What can you conclude? That is, can you call a method requiring a parameter with a passing one? What error is reported if any?

A method that requires a parameter **cannot be called** without one being sent in. There is a **compilation problem** when OneParam() is called without any parameters.

5. Let us create another program and save it as TwoParams.java

5.1. Copy the source code

```
public class TwoParams{
  public static void twoParams(int x, int y){
     System.out.println("First parameter is " + x);
     System.out.println("Second parameter is " + y);
  }

  public static void main(String[] args){
     int x = 5;
     int y = 10;

     int a;
     int b;

     twoParams(10, 20);
     twoparams(x + 2, y * 10);

     a = -2;
     b = 22;
     twoParams(a, b);
  }
}
```

5.2. Save and then compile your program until no errors and warnings are reported.

5.3. Run your program.

5.4. Simulate and write what will be displayed on the screen.

```
First parameter is 10
Second parameter is 20
First parameter is 7
Second parameter is 100
First parameter is -2
Second parameter is 22
```

5.5. Insert the statement twoParams(y, x);

What can you conclude? Is the result the same as of twoParams(x,y)?

Both "First parameter is 10" and "Second parameter is 5" will be printed. TwoParams(x, y); prints "First parameter is 5" and "Second parameter is 10", however this outcome **is different.**

5.5.1 What will happen if we forgot to type the comma between parameter? For example, will twoParams(x y) work? Java will encounter a **syntax error** as a result. The code won't successfully compile.

5.6. What will happen if we only supplied one parameter? For example, will twoParams(x) work? No, compilation error.

5.7. What will happen if we forgot all the parameters?

For example, will twoParams() work? No, compilation error.

5.8. What will happen if we used more than two parameters? For example, will twoParams(x, y, a) work? No, compilation error.

6. Are all methods of return type void? No

6.1. To answer that let's simulate the following program:

```
//Return value

public class WithReturnValue{
  public static int printSum(int x, int y){
     z = x + y;
     return z;
  }

  public static void main(String[] args){
     int a, b, c;

     System.out.println("Sum = " + printSum(5, 10));
     c = printSum(100, 300);
     System.out.println("c = " + c);

     a = 25;
     b = 75;

     System.out.println("Sum = " + printSum(a, b));
     System.out.println("Sum = " + Sum(a+2, b - 3));

  }
}
```

6.2. Save your program as WithReturnValue.java then compile and run and simulate your program. Write what will be displayed on the screen:

```
Compilation Error

Command execution failed.

BUILD FAILURE
```

6.3. What will happen if the statement return z; was omitted?

When the code attempts to delete the method's return value, a **compilation error** will still result.

6.4. Write a method named printDifference that accepts two integer parameters named x and y then computes the difference (x – y). Call this method inside the main().

```
/* Programmed by: Abenes, Enrico O.
   Program Title: Difference.java
   Program Date: July 27, 2023*/

package intl.cc1;

public class Difference {
    public static int printDifference(int x, int y) {
        int negatibo = x - y;
        return negatibo;
    }

    public static void main(String[] args) {
        int a = 10;
        int b = 5;

        int resulta = printDifference(a, b);
        System.out.println("Difference = " + resulta);
    }
}
```

6.5. Write a method named printProduct that accepts two double-type parameters named x and y. The method should compute and return the value of the product of x and y. Use double as return type. Call this method inside main();

```
/* Programmed by: Abenes, Enrico O.
   Program Title: Product.java
   Program Date: July 27, 2023*/

package intl.cc1;

public class Product {
    public static double printProduct(double x, double y) {
        double producto = x * y;
        return producto;
    }

    public static void main(String[] args) {
        double a = 4.6;
        double b = 5.9;

        double resulta = printProduct(a, b);
        System.out.println("Product = " + resulta);
    }
}
```

Rubrics:

Code Content	The source code submitted covers all the items specified in the activity and satisfactorily meets all these requirements. It also makes use of the specific features of Java specified in the activity.	20 pts
Code Function	The source code submitted works completely with no errors and provides the correct expected output when run.	20 pts
Code Syntax	The source code submitted has sound logic and follows proper syntax for Java, with no unnecessarily complicated code.	10 pts
Total		**100 pts**