Lab 3: Methods

Objectives

- To create methods, invoke methods, and pass arguments to a method.
- To determine the scope of variables.

Syntax

```
[modifiers] data-type method-name ([parameter-declaration]) {…}
```

Calling a Method

In creating a method, you give a definition of what the method is to do. To use a method, you have to call or invoke it. There are two ways to call a method; the choice is based on whether the method returns a value or not.

1. If the method returns a value, a call to the method is usually treated as a value. For example:

   ```
   data-type variable-name = method-name( passed-arguments);
   ```

   Calls **`method-name`** `(passed-arguments)` and assigns the result of the method to the variable `variable-name`. Another example of a call that is treated as a value is

   ```
   System.out.println(method-name( passed-arguments));
   ```

   which prints the return value of the method call **`method-name`**`( passed-arguments)`

2. If the method returns `void`, a call to the method must be a statement. For example, the method `println` returns `void`. The following call is a statement:

   ```
   System.out.println("Welcome to Java!");
   ```

Exercises

1. The factorial of non negative integer n is written n! and is defined as follows:

   $$n! = n \cdot (n-1).(n-2).\cdots.1$$

   and

   $$n! = 1 \ (for\ n = 0)$$

   While to compute the value of $\boldsymbol{e^x}$ by using the formula:

   $$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots$$

   Write a program that contains two methods ***factorial*** and ***exp.*** This program will ask the user to insert a number and then find $\boldsymbol{e^x}$ for this number.

2. An integer number is said to be *perfect number* if its factors including 1 (but not the number itself), sum to the number. For example 6 is perfect because 6 = 1 + 2 + 3. Write a method ***perfect*** that determine if parameter number is perfect number. Use this method main method that determine and prints all the perfect numbers between 1 and 1000.

3. Write a Java application that contains the following two methods:
   - ***binaryToDecimal***: This method accept a binary number and convert it to decimal.
   - ***DecimalToBinary***: This method accepts a decimal number and then converts it to its binary representation.

   The main method should ask the user to choose either to convert from binary to decimal or decimal to binary. Depending on the users choice one of the appropriate method will be chosen.