Faculty : Mr. Tarek Mizan
Lab Instructor: Nazmul Alam Diptu
Email : nazmul.diptu@northsouth.edu
Class Timing: ST 1:00 PM – 2:30 PM (LIB-611)
Topic: Methods

Objective

1. To learn about methods in Java
2. To learn how to solve a problem using single or multiple methods

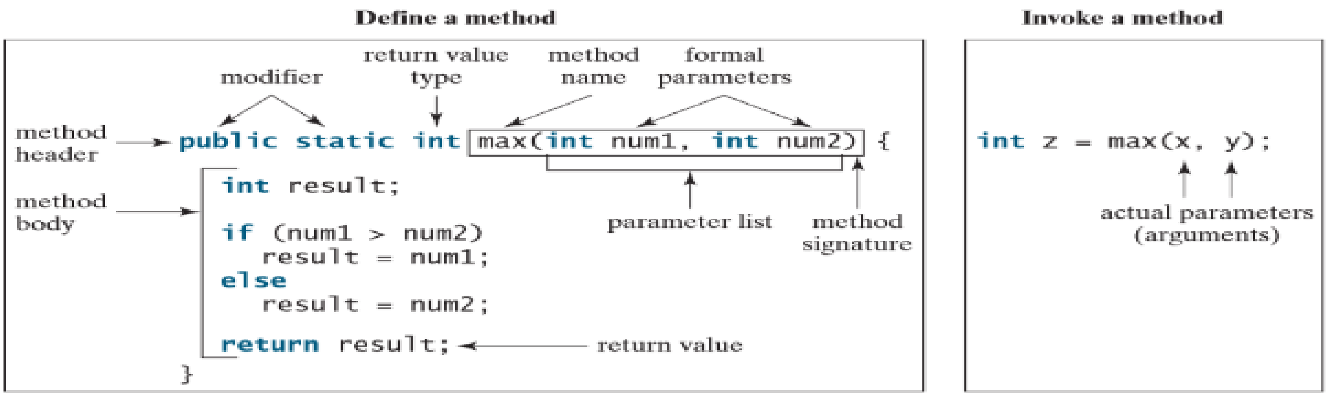
---

Methods: A Java method is a collection of statements that are grouped together to perform some specific task with or without returning anything to the caller. A method definition consists of its method name, parameters, return value type, and body. The syntax for defining a method is as follows:

```
Syntax :

modifier   returnValueType   methodName(list of parameters)  {
// Method body;
}
```

The following figure below illustrates each component of a method.



```
public static int max(int num1, int num2) {
    int result;

    if (num1 > num2)
       result = num1;
    else
       result = num2;

    return result;
}
```

```
int z = max(x, y);
```

Method01.java

```
public class Method01 {
    public static void main(String[] args) {
        int x = 10, y = 20;
        maxPrint(x, y); // calling method
        int m = max(x, y); // calling method
        System.out.println("Max Value: " + m);
    }
```

```java
    public static void maxPrint(int num1, int num2) {
        if (num1 > num2)
            System.out.println("Max: " + num1);
        else
            System.out.println("Max: " + num2);
    }

    // Method with return statement
    public static int max(int num1, int num2) {
        int result;
        if (num1 > num2)
            result = num1;
        else
            result = num2;
        return result;
    }
}
```

Passing Arguments by Values: When calling a method we need to provide arguments, which must be given in the same order as their respective parameters in the method signature. This is known as parameter order association. However, when we invoke a method with an argument, the value of the argument is passed to the parameter. This is referred to as pass-by-value. Java always maintains the rule of pass-by-value. The following code shows how java maintains pass-by-value rules.

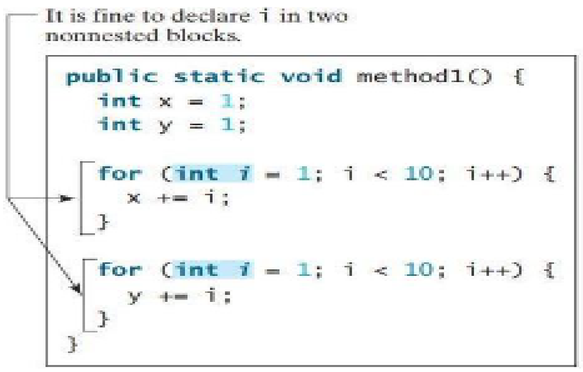
`Method02.java`

```java
public class Method02 {
    public static void main(String[] args) {

        int x = 5, y = 10;
        swap(x, y);
        System.out.println("Outside the method: " + x + ", " + y);
    }

    public static void swap(int a, int b) {
        int temp = a;
        a = b;
        b = temp;
        System.out.println("Inside the method: " + a + ", " + b);
    }
}
```

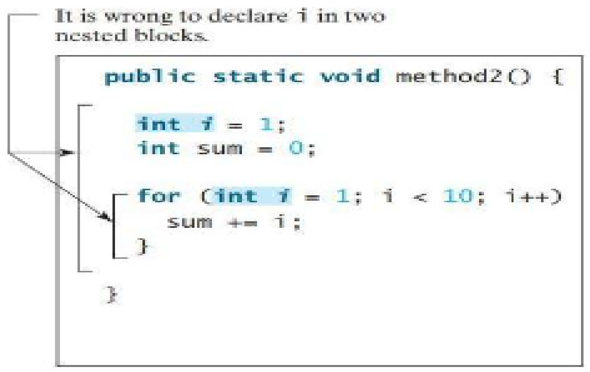
The Scope of Variables: A variable defined inside a method is referred to as a local variable. The scope of a local variable starts from its declaration and continues to the end of the block that contains the variable. A local variable must be declared and assigned a value before it can be used. However, a parameter is actually a local variable. The scope of a method parameter covers the entire method. The following figure shows that a variable can be declared multiple times in non-nested blocks, but only once in the nested blocks.



```
public static void method1() {
  int x = 1;
  int y = 1;

  for (int i = 1; i < 10; i++) {
    x += i;
  }

  for (int i = 1; i < 10; i++) {
    y += i;
  }
}
```



```
  public static void method2() {

    int i = 1;
    int sum = 0;

    for (int i = 1; i < 10; i++)
      sum += i;
    }

  }
```

String and Method: Following code shows how to pass a string as an argument and manipulate that string in Java.

`Method03.java`

```java
public class Method03 {

    public static void main(String[] args) {
        String userOne = userInfo("Nazmul Alam Diptu",
"nazmul.diptu@northsouth.edu", "Bashundhara R/A, Dhaka");
        System.out.println(userOne);
    }

    public static String userInfo(String name, String email, String add) {
        String thanksMsg = "Hello, "
                + name + ". Thanks for trying.\n\n"
                + "Your information given below:\n";
        String userInformation = "Name: " + name
                + "\nemail: " + email
                + "\nAddress: " + add;
        return thanksMsg.concat(userInformation);
    }
}
```

Array and Method: In Java, array is pass as a reference. Following code shows how to pass an array as an argument and reverse the elements of that array using methods.

`Method04.java`

```java
public class Method04 {
    public static void main(String[] args) {
        int[] numberList = { 1, 2, 3, 4, 5, 6 };
        reverseArr(numberList);
        for (int i = 0; i < numberList.length; i++)
            System.out.print(numberList[i] + ", ");
    }

    public static void reverseArr(int[] arr) {
        int arrLen = arr.length;
        for (int i = 0; i < arrLen / 2; i++) {
            int temp = arr[i];
            arr[i] = arr[arrLen - 1 - i];
            arr[arrLen - 1 - i] = temp;
        }
    }
}
```

Faculty : Mr. Tarek Mizan
Lab Instructor: Nazmul Alam Diptu
Email : nazmul.diptu@northsouth.edu
Class Timing: ST 1:00 PM – 2:30 PM (LIB-611)
Topic: Loops, Jump

Tasks:

1. Write a method countVowels(String sentence) that takes a String as a parameter and returns the number of vowels.

```
Sample String                               Sample Output
"Hello, Can you count NO Of vowels"         Vowels: 11
```

2. Write a method isPalindrome(String input) that determines if a String is a palindrome or not and in the main class takes a string as user input. A palindrome is when a String remains the same after reversing. The method should return the Boolean type.

```
Sample Input:                               Sample Output
Enter a string or number: 12321             12321 is a palindrome!
Enter a string or number: AOBBOA            AOBBOA is a palindrome!
Enter a string or number: JHON              JHON s not a palindrome!
```

3. Write a method sumDigit() that takes an integer and returns the sum of digits.

```
Sample Input :            Sample Output :
num = 1234                Sum of digits: 10
```