# Tutorial Sheet - Topic 07

## Programming Fundamentals 1

Methods, Parameters, Return Values, and Drivers

## Part A – Method Basics

### Q1. Terminology (recap)

Label each part of the method below:

```
public void printHello() {
    System.out.println("Hello");
}
```

(a) What is the **return type**?
(b) What is the **method name**?
(c) What are the **parameters**?
(d) What is inside the **method body**?

---

### Q2. Why use methods?

List **two reasons** why we use methods instead of repeating code.

---

### Q3. Predict the output

What is printed when this code runs?

```
public void greet() {
    System.out.println("Hi there!");
    System.out.println("Welcome to Java!");
}
```

If we call `greet();` twice, what happens?

---

# Part B – Writing and Calling Methods

## Q4. Writing your own method

Write a method called `sayName` that takes **no parameters** and prints your own name.

---

## Q5. Adding parameters

Write a method called `printSum` that takes **two integers** as parameters and prints their sum.
Example:
If called as `printSum(4, 6);`
→ Output should be: `The sum is 10`

---

## Q6. Return values

Write a method called `timesTwo` that takes one integer parameter and **returns** the number doubled.

Example:
If called as `timesTwo(8)` → the method should return `16`.

---

## Q7. Using a driver class

Explain briefly what the **Driver** class does in Java.
Why is the `main()` method usually placed there?

---

# Part C – Combining Methods and Drivers

## Q8. Read the example and answer:

```
public class Calculator {
    public int add(int x, int y) {
        return x + y;
    }
}
```

```
public class CalculatorDriver {
    public static void main(String[] args) {
        Calculator calc = new Calculator();
        int result = calc.add(5, 3);
        System.out.println("Sum = " + result);
    }
}
```

(a) What line **creates an object** of the Calculator class?
(b) What line **calls** the method?
(c) What is printed on screen?

---

## Q9. Extending the joke generator

Imagine a class `JokeGenerator` with three methods:

- `tellJoke()` – prints a joke (no input)
- `tellStory(String name)` – prints a story with a name
- `ageInTenYears(int age)` – returns a String message

Write a short pseudocode outline of a **Driver** class that:

1. Creates a `JokeGenerator` object
2. Calls each of the three methods in turn

---

## Q10. Writing other methods with parameters.

10.1 Write a method called `multiply` that takes **two integers** as parameters and **returns** their product.

10.2 Write a method that calls `multiply` and prints the result.

10.3 Where is the `multiply` method called?

## Q11 . Writing other methods with parameters.

11.1 Write a method called add3 that takes **three integers** as parameters and **returns** their sum.

11.2 Write a method that calls `add3` and prints the result.

11.3 Where is the `add3` method called?

# Part D – Beyond the Basics

## Q12. Boolean method

Write a method `isEven(int number)` that returns `true` if the number is even, otherwise `false`.
Then write one line of code that prints “Even!” if the method returns true.

---

## Q13. Overloading

The following two methods are in the same class:

```
void addNum(int x, int y) { ... }
void addNum(double x, double y) { ... }
```

(a) What is this feature called?
(b) What would happen if we called `addNum(3.5, 2.5);` ?

---

## Q14. Challenge – Fahrenheit to Celsius

Write a method that converts from Fahrenheit to Celsius and returns the Celsius value as a `double`.
Write the **method header** and one **example call** from `main()`.

Formula:
`C = (F - 32) × 5 / 9`

---

# ✳️ Optional Discussion Questions

- Why might Boolean methods make your program clearer?
- How do method parameters help avoid global variables?
- What makes overloaded methods useful in real programs?

---