DAY 7

1. What is a Method?

A **method** (also known as a function) is a block of code that performs a specific task. Think of it as a small, self-contained machine. You give it some input, it does its job, and it might give you back some output. Using methods is the best way to follow the **DRY (Don't Repeat Yourself)** principle, because it lets you reuse code without rewriting it.

Real-time use case: In an e-commerce application, you'd have a method called calculateTotalPrice(). You give it a list of items and their prices, and it performs the calculation, returning the final total. This method can be called from many different parts of the application without having to repeat the calculation logic.

---

2. How to Create a Method

Creating a method involves defining its signature and its body. A typical method signature looks like this:

access_modifier static return_type methodName(parameters) { // Method body }

- **access_modifier**: Controls who can use the method (e.g., public, private). We'll dive deeper into this in our OOP lessons.
- **static**: (Optional) Means the method belongs to the class itself, not an object. We'll use this a lot for now because our main method is static.
- **return_type**: The type of value the method gives back.
- **methodName**: A descriptive name for the method.
- **parameters**: (Optional) The list of inputs the method accepts.

Code Example:

Java

```
public class Calculator {
   // This method takes two integer parameters and returns an integer
   public static int add(int num1, int num2) {
      int sum = num1 + num2;
      return sum; // This is the return statement
   }
}
```

K VENKATA NAGARJUNA REDDY

3. What are Parameters?

Parameters are variables that act as placeholders for the values a method needs to receive as input. They are defined in the method's signature. When you call a method, you provide **arguments**, which are the actual values passed to the parameters.

Code Example:

In the add method above, num1 and num2 are the parameters. When you call the method, you pass arguments to them.

Java

```
// Calling the 'add' method with arguments 10 and 20

int result = Calculator.add(10, 20);

System.out.println(result); // Output: 30
```

---

4. What is the Return Type?

The **return type** specifies the data type of the value that a method sends back after it finishes its job. If a method does not return a value, its return type is void. A method with a return type other than void must include a return statement that sends back a value of the declared type.

Code Example:

Java

```
public class User {
   // This method has a return type of String
   public static String getUserStatus(boolean isLoggedIn) {
      if (isLoggedIn) {
         return "Active";
      } else {
         return "Offline";
      }
   }


   // This method has a return type of void because it doesn't return anything
   public static void printMessage(String message) {
      System.out.println(message);
   }
}
```

K VENKATA NAGARJUNA REDDY

DAY 7

5. Interview Questions on Methods

1. **What is the primary purpose of a method in Java?**
    - **Expected Answer:** To perform a specific task, organize code, and promote code reuse (DRY principle).
2. **Explain the difference between a method's parameters and its arguments.**
    - **Expected Answer:** Parameters are the variables in the method definition, while arguments are the actual values passed during the method call.
3. **What is a void method?**
    - **Expected Answer:** A method that does not return any value. It's used when a method's purpose is to perform an action rather than to produce an output.
4. **Can you have a method without a return type?**
    - **Expected Answer:** No, every method must have a return type. If it doesn't return a value, its return type must be void.
5. **What happens if you have a method with a return type of int but you don't include a return statement?**
    - **Expected Answer:** The Java compiler will produce an error, as it requires all code paths in a non-void method to return a value.

K VENKATA NAGARJUNA REDDY