DAY 8

1. How to Print Values in the Console

In Java, we use the System.out.println() statement to display information. This command is part of the standard Java library and is your primary tool for showing text, variable values, or the results of calculations.

Code Example:

Java

```
public class PrintExample {
  public static void main(String[] args) {
    // This prints a simple string to the console
    System.out.println("Hello, World!");
  }
}
```

Real-time use case:

- **Debugging:** As a developer, you'll use print statements constantly to see the value of a variable at a specific point in your code. This is an essential technique for finding and fixing bugs.
- **User Feedback:** When a user interacts with a command-line program, print statements are used to give them instructions or show them the results of their actions.

---

2. Different Statements for Printing Values

Java provides a few different statements for printing, each with a slightly different behavior. The most common ones are println(), print(), and printf().

- **System.out.println()**: This is the most frequently used statement. It prints the specified value to the console and then moves the cursor to the **next line**. ln stands for "line."
- **System.out.print()**: This statement prints the value to the console but **does not** move the cursor to the next line. Any subsequent output will appear on the same line.
- **System.out.printf()**: This is a powerful, formatted print statement. It allows you to print formatted output using a format string and a list of arguments. f stands for "formatted."

K VENKATA NAGARJUNA REDDY

DAY 8

Code Example:

Java

```
public class PrintStatements {
    public static void main(String[] args) {
        System.out.print("This is on the first line. ");
        System.out.print("This is also on the first line.\n");
        // Output: This is on the first line. This is also on the first line.

        int age = 20;
        double price = 45.75;

        System.out.println("Your age is: " + age + " years old.");
        // Output: Your age is: 20 years old.

        System.out.printf("The price is $%.2f and your age is %d.", price, age);
        // Output: The price is $45.75 and your age is 20.
    }
}
```

---

3. Understanding the Difference Between Print Statements

The key difference lies in how they handle the **newline character**.

- println() automatically adds a newline (\n) after printing.
- print() does not, keeping all output on the same line.
- printf() gives you the most control. You explicitly use format specifiers like %d (for integers) and %.2f (for a float/double with two decimal places), and can add a newline (\n) manually within the format string.

Visualizing the difference:

K VENKATA NAGARJUNA REDDY

DAY 8

4. Interview Questions on Print Statements

1. **What is the difference between System.out.print() and System.out.println()?**
   - **Expected Answer:** println() prints the output and then moves to a new line, while print() prints the output and keeps the cursor on the same line.
2. **When would you use System.out.printf() over println()?**
   - **Expected Answer:** When you need to display formatted output with specific alignment, padding, or number of decimal places. It's more efficient for complex output formatting.
3. **Explain the purpose of the \n and \t escape sequences.**
   - **Expected Answer:** \n represents a **newline** character, and \t represents a **tab** character. They are used to control the layout of the text.
4. **How would you print the value of a variable named score using println()?**
   - **Expected Answer:** System.out.println("The score is: " + score);
   - This shows an understanding of string concatenation using the + operator.
5. **What is a format specifier in printf()? Give an example.**
   - **Expected Answer:** A format specifier is a placeholder that indicates where a variable's value should be inserted. Examples include %s for a string, %d for an integer, and %f for a floating-point number.

K VENKATA NAGARJUNA REDDY