## DAY 8 :

## ASSIGNMENT 1:

## Task 1: Bit Manipulation Basics

Create a function that counts the number of set bits (1s) in the binary representation of an integer. Extend this to count the total number of set bits in all integers from 1 to n.

## ANSWER:

### Step 1: Find the position of the rightmost set bit in a single integer

The position of the rightmost set bit can be found using the expression num & -num. This expression isolates the rightmost set bit of num. Then, we can use the Integer.numberOfTrailingZeros method to find the position of that bit.

Here's the Java implementation:

```
public class BitManipulation {
    // Function to find the position of the rightmost set bit in a single integer
    public static int findRightmostSetBitPosition(int num) {
        if (num == 0) {
            return -1; // return -1 if there is no set bit
        }
        int position = Integer.numberOfTrailingZeros(num & -num) + 1;
        return position;
    }

    public static void main(String[] args) {
        int num = 18; // example number (binary: 10010)
        System.out.println("Position of the rightmost set bit in " + num + " is: " +
findRightmostSetBitPosition(num));
    }
}
```

## Explanation

### 1. findRightmostSetBitPosition Function:

- The expression num & -num isolates the rightmost set bit. For example, for num = 18 (binary 10010), num & -num results in 00010.

- Integer.numberOfTrailingZeros(num & -num) returns the number of trailing zeros in the binary representation of num & -num. This gives us the position of the rightmost set bit, minus one.

- Adding 1 to this result gives us the 1-based position of the rightmost set bit.

- If num is 0, we return -1 as there is no set bit.

### 2. main Method:

- The main method demonstrates the usage of the findRightmostSetBitPosition function by finding the position of the rightmost set bit in a given integer.

## Step 2: Find the positions of the rightmost set bit for all integers from 1 to n

We can extend this to iterate through all integers from 1 to \( n \) and print the position of the rightmost set bit for each integer.

Here's the Java implementation:

```
public class BitManipulation {
    // Function to find the position of the rightmost set bit in a single integer
    public static int findRightmostSetBitPosition(int num) {
        if (num == 0) {
            return -1; // return -1 if there is no set bit
        }
        int position = Integer.numberOfTrailingZeros(num & -num) + 1;
        return position;
    }

    // Function to find the positions of the rightmost set bit for all integers from 1 to n
```

```
    public static void findRightmostSetBitPositions(int n) {
        for (int i = 1; i <= n; i++) {
            System.out.println("Position of the rightmost set bit in " + i + " is: " +
findRightmostSetBitPosition(i));
        }
    }

    public static void main(String[] args) {
        int n = 10; // example number
        findRightmostSetBitPositions(n);
    }
}
```

## Explanation

### 1. findRightmostSetBitPositions Function:

- This function iterates through all numbers from 1 to \( n \).

- For each number, it calls findRightmostSetBitPosition and prints the result.

### 2. main Method:

- The main method demonstrates the usage of the findRightmostSetBitPositions function by printing the positions of the rightmost set bit for each integer from 1 to \( n \).

This code snippet provides a straightforward approach to finding and printing the positions of the rightmost set bit for a range of integers, using bit manipulation techniques in Java.