

Name: Swarnali Ghosh

Java Pre-Skilling Training Session

Assignment -5.5 (Task1 & Task2)

Module-5 (DAY 12)

Mail-id: [swarnalighosh666@gmail.com](mailto:swarnalighosh666@gmail.com)

### 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.

SOLUTION:

→ Here is the Java code implementation to perform the above function (counting the number of set bits (1s) in the binary representation of an integer as well as counting the total number of set bits in all integers from 1 to n)-

```
package com.wipro.swarnali;

public class SetBitsCounter {

    public static int countSetBits(int num) {
        int count = 0;
        while (num > 0) {
            count += num & 1;
        }
    }
}
```

```

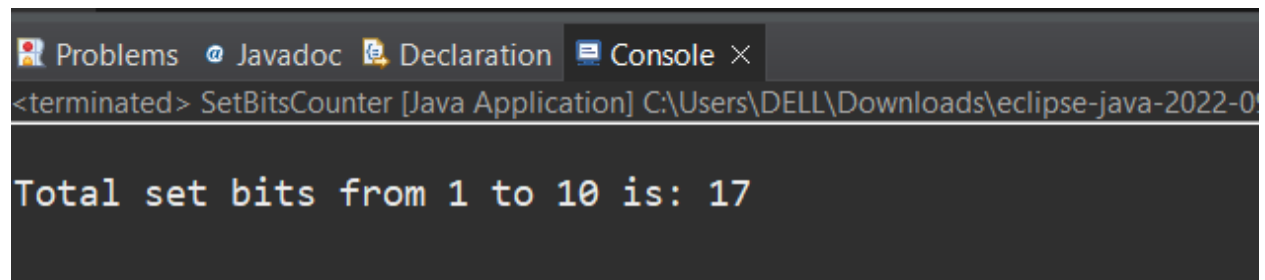
        num >>= 1;
    }
    return count;
}

public static int countTotalSetBits(int n) {
    int totalSetBits = 0;
    for (int i = 1; i <= n; i++) {
        totalSetBits += countSetBits(i);
    }
    return totalSetBits;
}

public static void main(String[] args) {
    int n = 10;
    System.out.println("\nTotal set bits from 1 to " + n + " is: "
+ countTotalSetBits(n));
}
}

```

Output: -



The screenshot shows the Eclipse IDE's console window. The title bar includes tabs for 'Problems', 'Javadoc', 'Declaration', and 'Console'. The console output shows the message: 'Total set bits from 1 to 10 is: 17'. The window title is '<terminated> SetBitsCounter [Java Application] C:\Users\DELL\Downloads\eclipse-java-2022-01'.

## Task 2: Unique Elements Identification

Given an array of integers where every element appears twice except for two, write a function that efficiently finds these two non-repeating elements using bitwise XOR operations.

## SOLUTION:

→ According to the question, here is an array of integers where every element appears twice except for two, and I have written a function that efficiently finds these two non-repeating elements using bitwise XOR operations.

Here is the Java code Implementation-

```
package com.wipro.swarnali;

public class UniqueElementsIdentification {

    public static void main(String[] args) {

        int[] num_s = {1, 2, 3, 2, 1, 4};

        int[] result = findUniqueElements(num_s);

        System.out.println("\nThe two non-repeating elements are: " +
result[0] + " and " + result[1]);
    }

    public static int[] findUniqueElements(int[] num_s) {

        // Step 1: XOR all the numbers to get the XOR of the two
unique numbers
        int xor = 0;
        for (int num : num_s) {
            xor ^= num;
        }

        // Step 2: Find a set bit in the result (rightmost set bit)
        int setBit = xor & -xor;

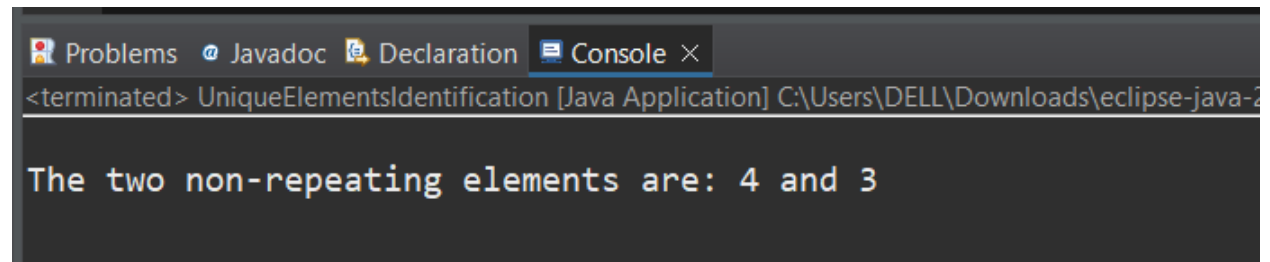
        // Step 3: Partition the array into two groups and XOR the
numbers in each group
        int unique1 = 0;
```

```
int unique2 = 0;

for (int num : num_s) {
    if ((num & setBit) == 0) {
        unique1 ^= num;
    } else {
        unique2 ^= num;
    }
}

return new int[]{unique1, unique2};
}
```

Output: -

The screenshot shows the Eclipse IDE's console window. The title bar at the top includes tabs for 'Problems', 'Javadoc', 'Declaration', and 'Console'. The 'Console' tab is active, displaying the output of the Java application. The output text reads: 'The two non-repeating elements are: 4 and 3'. Above the output, the console title is '<terminated> UniqueElementsIdentification [Java Application] C:\Users\DELL\Downloads\eclipse-java-2'.

```
<terminated> UniqueElementsIdentification [Java Application] C:\Users\DELL\Downloads\eclipse-java-2
The two non-repeating elements are: 4 and 3
```