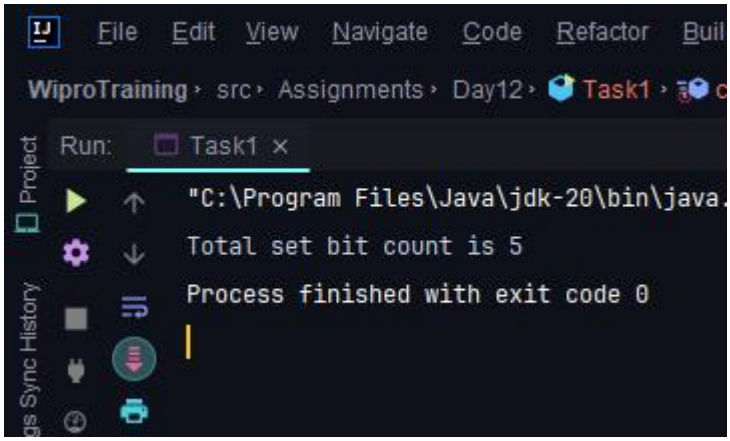
Day 12

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.

Program:

```
package Assignments.Day12;

public class Task1 {

	static int countSetBits(int n)
	{
	   int bitCount = 0;
	   for (int i = 1; i <= n; i++)
	      bitCount += countSetBitsUtil(i);

	   return bitCount;
	}

	static int countSetBitsUtil(int x)
	{
	   if (x <= 0)
	      return 0;
	   return (x % 2 == 0 ? 0 : 1)
	         + countSetBitsUtil(x / 2);
	}
	public static void main(String[] args)
	{
	   int n = 4;
	   System.out.print("Total set bit count is ");
	   System.out.print(countSetBits(n));
	}
}
```

Output:



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.

Program:

```
package Assignments.Day12;

import java.util.*;

public class Task2 {

 public static int[] get2NonRepeatingNos(int[] nums) {

        int diff = 0;
        for (int num : nums) {
            diff ^= num;
        }

        diff &= -diff;
        int[] rets = new int[2];
        Arrays.fill(rets, 0);
        for (int num : nums) {
            if ((num & diff) == 0) {
                rets[0] ^= num;
            } else {
                rets[1] ^= num;
            }
```

```
        }

        if (rets[0] > rets[1]) {
            int temp = rets[0];
            rets[0] = rets[1];
            rets[1] = temp;
        }

        return rets;
    }

    public static void main(String[] args) {
        int[] arr = {2, 3, 7, 9, 11, 2, 3, 11};
        int[] result = get2NonRepeatingNos(arr);
        System.out.println("The non-repeating elements are " + result[0] + " and "
+ result[1]);
    }
}
```

Output: