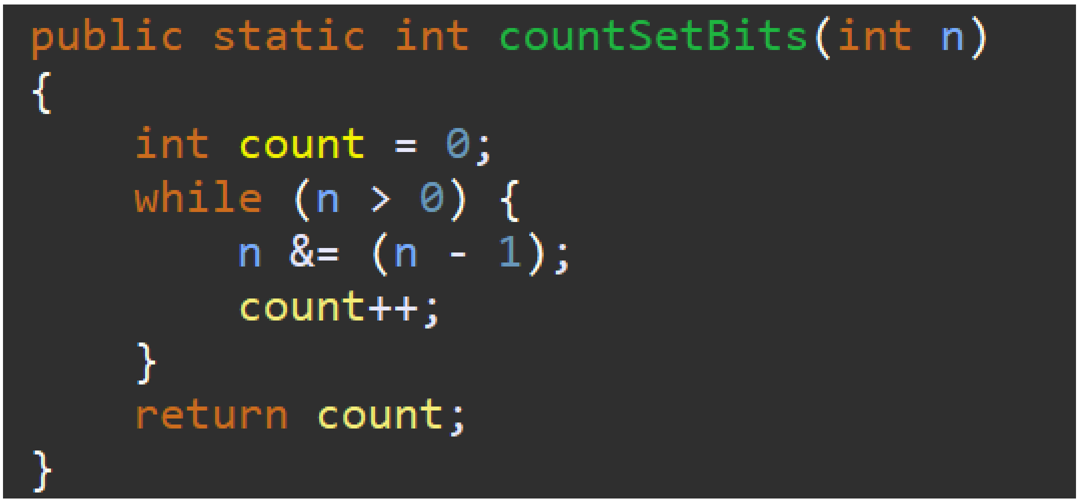
Task 1: Bit Manipulation Basics

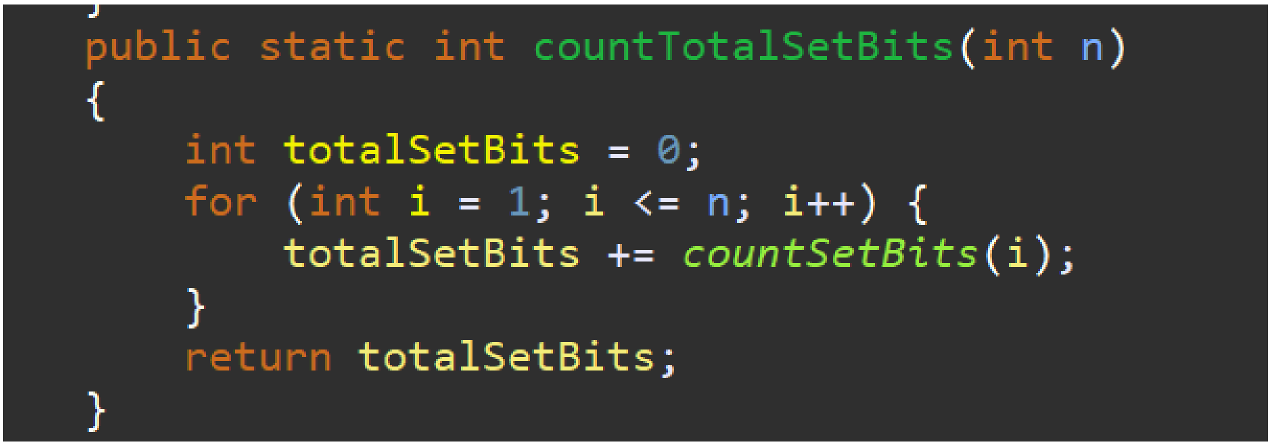
Create a function that counts the number of set bits (1s) in the binary representation of an integer. Extend this to count the total number of set bits in all integers from 1 to n.

Function for counting number of set bits: -



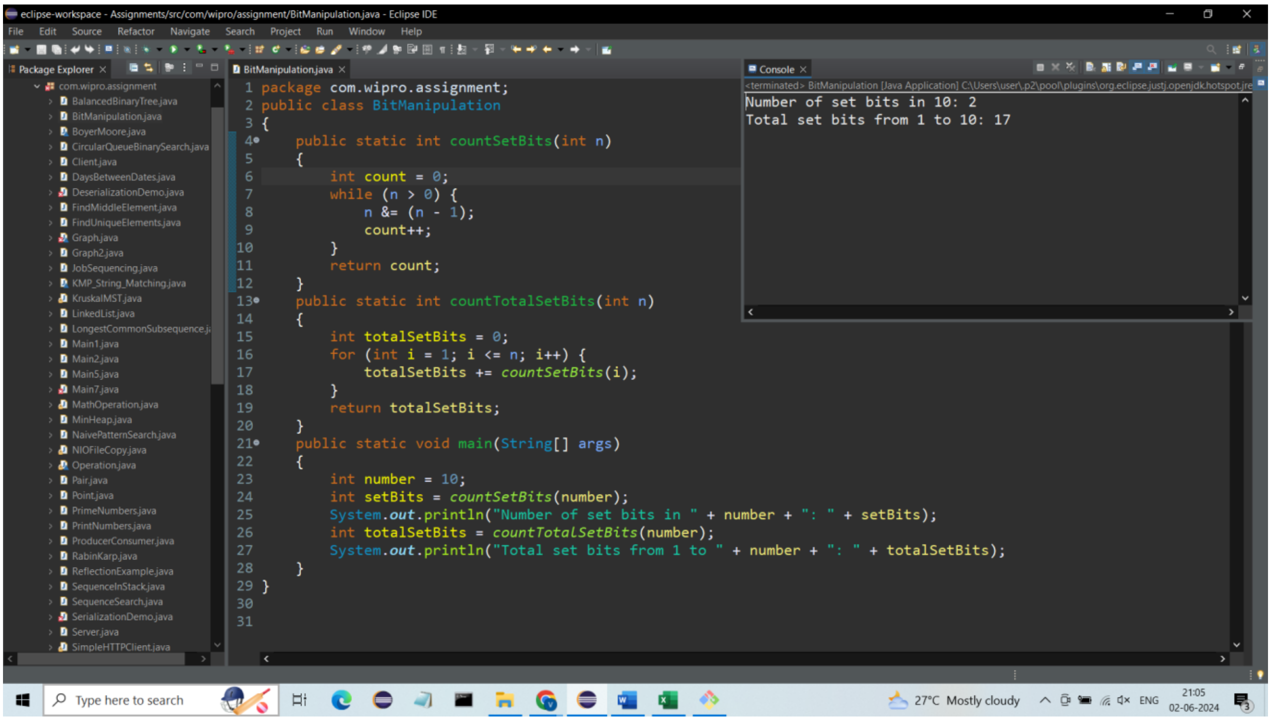
```java
public static int countSetBits(int n)
{
    int count = 0;
    while (n > 0) {
        n &= (n - 1);
        count++;
    }
    return count;
}
```

Function for counting total number of set bits in all integers from 1 to n: -



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

Output: -



```java
package com.wipro.assignment;
public class BitManipulation
{
    public static int countSetBits(int n)
    {
        int count = 0;
        while (n > 0) {
            n &= (n - 1);
            count++;
        }
        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 number = 10;
        int setBits = countSetBits(number);
        System.out.println("Number of set bits in " + number + ": " + setBits);
        int totalSetBits = countTotalSetBits(number);
        System.out.println("Total set bits from 1 to " + number + ": " + totalSetBits);
    }
}
```

```
Number of set bits in 10: 2
Total set bits from 1 to 10: 17
```

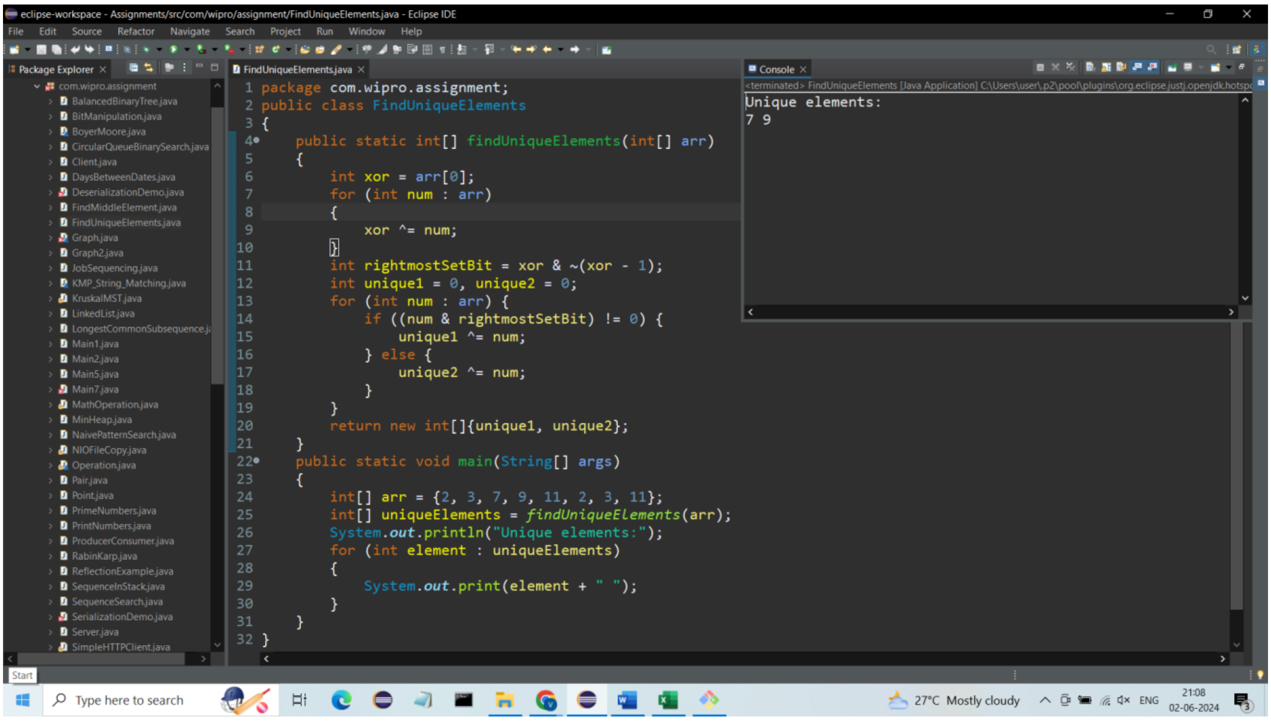
Task 2: Unique Elements Identification

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.

Function for utilizing XOR operations to find the two unique elements in an array where all other elements appear twice.

```java
public static int[] findUniqueElements(int[] arr)
{
	int xor = arr[0];
	for (int num : arr)
	{
		xor ^= num;
	}
	int rightmostSetBit = xor & ~(xor - 1);
	int unique1 = 0, unique2 = 0;
	for (int num : arr) {
		if ((num & rightmostSetBit) != 0) {
			unique1 ^= num;
		} else {
			unique2 ^= num;
		}
	}
	return new int[]{unique1, unique2};
}
```

Output: -



```java
package com.wipro.assignment;
public class FindUniqueElements
{
	public static int[] findUniqueElements(int[] arr)
	{
		int xor = arr[0];
		for (int num : arr)
		{
			xor ^= num;
		}
		int rightmostSetBit = xor & ~(xor - 1);
		int unique1 = 0, unique2 = 0;
		for (int num : arr) {
			if ((num & rightmostSetBit) != 0) {
				unique1 ^= num;
			} else {
				unique2 ^= num;
			}
		}
		return new int[]{unique1, unique2};
	}
	public static void main(String[] args)
	{
		int[] arr = {2, 3, 7, 9, 11, 2, 3, 11};
		int[] uniqueElements = findUniqueElements(arr);
		System.out.println("Unique elements:");
		for (int element : uniqueElements)
		{
			System.out.print(element + " ");
		}
	}
}
```

```
Unique elements:
7 9
```