# Class Notes 10

- Python
- JavaScript
- JAVA

**Question 1**

Given a number, we need to find sum of its digits using recursion.

Examples:

```
Input : 12345
Output : 15

Input : 45632
Output :20
```

**Explanation :**

The step-by-step process for a better understanding of how the algorithm works.

Let the number be 12345.

Step 1-> 12345 % 10 which is equal-too 5 + ( send 12345/10 to next step )

Step 2-> 1234 % 10 which is equal-too 4 + ( send 1234/10 to next step )

Step 3-> 123 % 10 which is equal-too 3 + ( send 123/10 to next step )

Step 4-> 12 % 10 which is equal-too 2 + ( send 12/10 to next step )

Step 5-> 1 % 10 which is equal-too 1 + ( send 1/10 to next step )

Step 6-> 0 algorithm stops

**Time complexity :** O(logn) where n is the given number.

**Auxiliary space :** O(logn) due to recursive stack space.

**Solution :**

```java
import java.io.*;

class sum_of_digits
{
  static int sum_of_digit(int n)
  {
    if (n == 0)
      return 0;
    return (n % 10 + sum_of_digit(n / 10));
  }

  // Driven Program to check above
  public static void main(String args[])
  {
    int num = 12345;
    int result = sum_of_digit(num);
    System.out.println("Sum of digits in " +
            num + " is " + result);
  }
}
```

**Question 2**

Given two numbers **a** and **b**, the task is to find the GCD of the two numbers.

**Note:** GCD (Greatest Common Divisor) or HCF (Highest Common Factor) of two numbers is the largest number that divides both of them.

***Input:*** *a = 20, b = 28*

**Output:** 4

**Explanation:** The factors of 20 are 1, 2, 4, 5, 10 and 20. The factors of 28 are 1, 2, 4, 7, 14 and 28. Among these factors, 1, 2 and 4 are the common factors of both 20 and 28. The greatest among the common factors is 4.

***Input:*** *a = 60, b = 36*

**Output:** 12

**Time Complexity:** O(log(min(a,b))

**Auxiliary Space:** O(log(min(a,b))

**Solution :**

```java
import java.io.*;
```

```
class Test
{
  // Recursive function to return gcd of a and b
  static int gcd(int a, int b)
  {
  if (b == 0)
    return a;
  return gcd(b, a % b);
  }

  // Driver method
  public static void main(String[] args)
  {
    int a = 98, b = 56;
    System.out.println("GCD of " + a +" and " + b + " is " + gcd(a, b));
  }
}
```

### Question 3

Given an array of integers, print sums of all subsets in it. Output sums can be printed in any order.

**Examples :**

```
Input : arr[] = {2, 3}
Output: 0 2 3 5

Input : arr[] = {2, 4, 5}
Output : 0 2 4 5 6 7 9 11
```

**Explanation :**

We can recursively solve this problem. There are total 2n subsets. For every element, we consider two choices, we include it in a subset and we don't include it in a subset.

**Time complexity :** O(2^n)

**Auxiliary Space :** O(n)

**Solution :**

```
import java.io.*;

class GFG {

  static void subsetSums(int[] arr, int l, int r, int sum)
  {

    // Print current subset
    if (l > r) {
      System.out.print(sum + " ");
      return;
    }

    // Subset including arr[l]
    subsetSums(arr, l + 1, r, sum + arr[l]);

    // Subset excluding arr[l]
    subsetSums(arr, l + 1, r, sum);
  }

  // Driver code
  public static void main(String[] args)
  {
    int[] arr = { 5, 4, 3 };
    int n = arr.length;

    subsetSums(arr, 0, n - 1, 0);
  }
}
```