LAB # 03

RECURSION

OBJECTIVE: To understand the complexities of the recursive functions and a way to reduce these complexities.

LAB TASK

1. Write a program which takes an integer value (k) as input and prints the sequence of numbers from k to 0 in descending order.

```
package labs3;
import java.util.Scanner;

public class LABS3 {
    public static void printSequence(int k) {

        if (k < 0) {
            return;
        }
        System.out.print(k + " ");
        printSequence(k - 1);
    }

        public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter an integer value (k): ");
        int k = scanner.nextInt();
        printSequence(k);

    }
```

```
run:
Enter an integer value (k): 5
5 4 3 2 1 0 BUILD SUCCESSFUL (total time: 16 seconds)
```

2. Write a program to reverse your full name using Recursion.

```java
package labs3;
import java.util.Scanner;

public class LABS3 {
    public static String reverse(String name) {
        if (name.isEmpty()) {
            return name;
        }
        return reverse(name.substring(1)) + name.charAt(0);
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter your full name: ");
        String fullName = scanner.nextLine();
        System.out.println("Reversed name: " + reverse(fullName));

    }
```

```
run:
Enter your full name: sara abbasi
Reversed name: isabba aras
BUILD SUCCESSFUL (total time: 4 seconds)
```

3. Write a program to calculate the sum of numbers from 1 to N using recursion. N should be user input.

```java
package labs3;
import java.util.Scanner;

public class LABS3 {

    public static int sum(int n) {
        if (n <= 1) {
            return n;
        }
        return n + sum(n - 1);
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a value for N: ");
        int n = scanner.nextInt();
        System.out.println("Sum from 1 to " + n + " is: " + sum(n));

    }
```

```
run:
Enter a value for N: 5
Sum from 1 to 5 is: 15
BUILD SUCCESSFUL (total time: 3 seconds)
```

4. Write a recursive program to calculate the sum of elements in an array.

```java
package labtask3;
import java.util.Scanner;


public class Labtask3 {
    public static int sumArray(int[] arr, int index) {
        if (index < 0) {
            return 0;
        }
        return arr[index] + sumArray(arr, index - 1);
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of elements in the array: ");
        int size = scanner.nextInt();
        int[] arr = new int[size];
        System.out.println("Enter the elements of the array:");
        for (int i = 0; i < size; i++) {
            arr[i] = scanner.nextInt();
        }
        System.out.println("Sum of array elements: " + sumArray(arr, size - 1));

    }
```

```
run:
Enter the number of elements in the array: 4
Enter the elements of the array:
1
2
3
4
Sum of array elements: 10
BUILD SUCCESSFUL (total time: 7 seconds)
```

5. Write a recursive program to calculate the factorial of a given integer n

```java
package labtask3;
import java.util.Scanner;


public class Labtask3 {
    //q4
    public static int factorial(int n) {
        if (n <= 1) {
            return 1;
        }
        return n * factorial(n - 1);
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number to calculate factorial: ");
        int n = scanner.nextInt();
        System.out.println("Factorial of " + n + " is: " + factorial(n));

    }
}
```

```
run:
Enter a number to calculate factorial: 5
Factorial of 5 is: 120
BUILD SUCCESSFUL (total time: 1 second)
```

6. Write a program to count the digits of a given number using recursion.

```java
package labtask3;
import java.util.Scanner;


public class Labtask3 {
    public static int countDigits(int n) {
        if (n == 0) {
            return 0;
        }
        return 1 + countDigits(n / 10);
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number to count its digits: ");
        int n = scanner.nextInt();
        System.out.println("Number of digits in " + n + " is: " + countDigits(Math.abs(n)));
    }
```

```
run:
Enter a number to count its digits: 12345
Number of digits in 12345 is: 5
BUILD SUCCESSFUL (total time: 19 seconds)
```

HOME TASK

1. Write a java program to find the N-th term in the Fibonacci series using Memoization.

```java
package labtask3;
import java.util.Scanner;
import java.util.HashMap;

public class Labtask3 {
    private static HashMap<Integer, Long> memo = new HashMap<>();

    public static long fibonacci(int n) {
        if (n <= 1) {
            return n;
        }
        if (memo.containsKey(n)) {
            return memo.get(n);
        }
        long result = fibonacci(n - 1) + fibonacci(n - 2);
        memo.put(n, result);
        return result;
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the position (N) of Fibonacci series: ");
        int n = scanner.nextInt();
        System.out.println("The " + n + "-th term in the Fibonacci series is: " + fibonacci(n));
        scanner.close();
    }
```

```
run:
Enter the position (N) of Fibonacci series: 10
The 10-th term in the Fibonacci series is: 55
BUILD SUCCESSFUL (total time: 4 seconds)
```

2. Write a program to count the digits of a given number using recursion.

```java
package labtask3;
import java.util.Scanner;


public class Labtask3 {
    public static int countDigits(int n) {
        if (n == 0) {
            return 0;
        }
        return 1 + countDigits(n / 10);
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a number to count its digits: ");
        int n = scanner.nextInt();
        System.out.println("Number of digits in " + n + " is: " + countDigits(Math.abs(n)));

    }
```

```
run:
Enter a number to count its digits: 12345
Number of digits in 12345 is: 5
BUILD SUCCESSFUL (total time: 3 seconds)
```

3. Write a java program to check whether a given string is a palindrome or not. A palindrome is a string that reads the same forwards and backwards.Print "YES" if the string is a palindrome, otherwise print "NO".

```
package labtask3;
import java.util.Scanner;


public class Labtask3 {
    public static boolean isPalindrome(String str, int start, int end) {
        if (start >= end) {
            return true;
        }
        if (str.charAt(start) != str.charAt(end)) {
            return false;
        }
        return isPalindrome(str, start + 1, end - 1);
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter a string to check if it's a palindrome: ");
        String input = scanner.nextLine();
        if (isPalindrome(input, 0, input.length() - 1)) {
            System.out.println("YES");
        } else {
            System.out.println("NO");
        }

    }
```

```
run:
Enter a string to check if it's a palindrome: madam
YES
BUILD SUCCESSFUL (total time: 6 seconds)
```

4. Write a recursive program to find the greatest common divisor (GCD) of two numbers using Euclid's algorithm.

```java
package labtask3;
import java.util.Scanner;


public class Labtask3 {
    public static int gcd(int a, int b) {
        if (b == 0) {
            return a;
        }
        return gcd(b, a % b);
    }

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the first number: ");
        int a = scanner.nextInt();
        System.out.print("Enter the second number: ");
        int b = scanner.nextInt();
        System.out.println("The GCD of " + a + " and " + b + " is: " + gcd(a, b));
        scanner.close();
    }
}
```

```
run:
Enter the first number: 48
Enter the second number: 18
The GCD of 48 and 18 is: 6
BUILD SUCCESSFUL (total time: 8 seconds)
```