# Lab # 03

# Recursion

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

## Lab Tasks:

**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.

**Source Code:**

```
import java.util.Scanner;
public class Lab3Task1 {
    static void Nnumbers(int k) {
        if (k > 0) {
            System.out.println(k);
            Nnumbers(k - 1);
        }
    }
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter a value of k: ");
        Nnumbers(input.nextInt());
        input.close();
    }
}
```

**Output:**

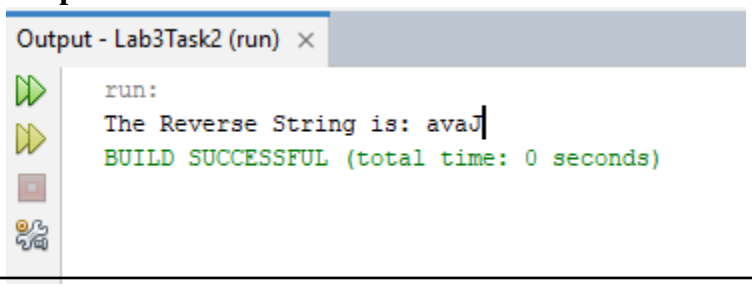
```
Output - Lab3Task1 (run)
run:
Enter a value of k: 10
10
9
8
7
6
5
4
3
2
1
BUILD SUCCESSFUL (total time: 2 seconds)
```

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

**Source Code:**

```
package lab3task2;
public class Lab3Task2 {
    public static void main(String[] args) {
        String str = "Java";
        String reversed = reverseString(str);
        System.out.println("The Reverse String is: " + reversed);
    }
    public static String reverseString(String str) {
        if (str.isEmpty()) {
            return str;
        }
        return reverseString(str.substring(1)) + str.charAt(0);
    }
}
```

**Output:**




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

**Source Code:**

```
package lab3task3;
import java.util.Scanner;
public class Lab3Task3 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a number: ");
        int n = sc.nextInt();
        System.out.println("Total is: " + recurSum(n));
    }
    public static int recurSum(int n) {
        if (n <= 1) {
            return n;
        } else {
            System.out.println(n);
        }
        return n + recurSum(n - 1);
    }
}
```

**Output:**

```
Output - Lab3Task3 (run)
run:
Enter a number: 10
10
9
8
7
6
5
4
3
2
Total is: 55
BUILD SUCCESSFUL (total time: 4 seconds)
```

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

**Source Code:**

```
package lab3task4;

public class Lab3Task4 {

    public static void main(String[] args) {

        int[] array = {1, 2, 3, 4, 5}; // Example array

        int sum = sumArray(array, array.length);

        System.out.println("Sum of elements in the array: " + sum);

    }

    public static int sumArray(int[] array, int n) {

        if (n <= 0) {

            return 0;

        }

        return array[n - 1] + sumArray(array, n - 1);

    }

}
```

**Output:**

```
Output - Lab3Task4 (run)
run:
Sum of elements in the array: 15
BUILD SUCCESSFUL (total time: 0 seconds)
```

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

**Source Code:**

```
package lab3task5;

public class Lab3Task5 {

    public static void main(String[] args) {

        int n = 5; // Example input

        int factorial = factorial(n);

        System.out.println("Factorial of " + n + " is: " + factorial);

    }

    public static int factorial(int n) {

        if (n <= 1) {

            return 1;

        }

        return n * factorial(n - 1);

    }

}
```

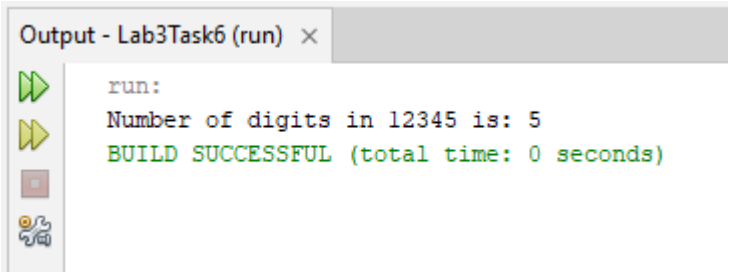
**Output:**

```
Output - Lab3Task5 (run)
run:
Factorial of 5 is: 120
BUILD SUCCESSFUL (total time: 0 seconds)
```

**6.** Write a program to count the digits of a given number using recursion.
**Source Code:**

```
package lab3task6;

public class Lab3Task6 {

    public static void main(String[] args) {

        int number = 12345; // Example input

        int count = countDigits(number);

        System.out.println("Number of digits in " + number + " is: " + count);

    }

    public static int countDigits(int n) {

        if (n == 0) {

            return 0;

        }

        return 1 + countDigits(n / 10);

    }

}
```

**Output:**



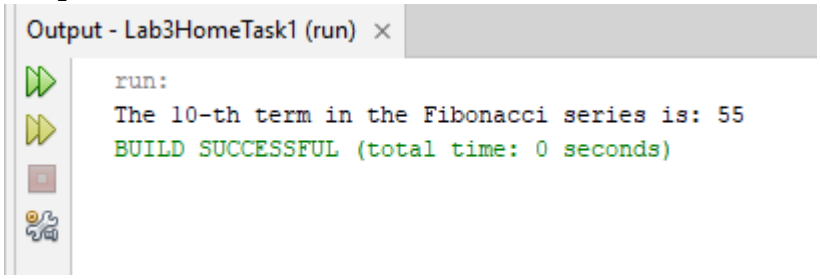
# Home Tasks:

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

**Source Code:**

```
package lab3hometask1;
import java.util.HashMap;
public class Lab3HomeTask1 {
    private static HashMap<Integer, Integer> memo = new HashMap<>();
    public static void main(String[] args) {
        int n = 10; // Example input
        int fibonacciTerm = fibonacci(n);
        System.out.println("The " + n + "-th term in the Fibonacci series is: " + fibonacciTerm);
    }
    public static int fibonacci(int n) {
        if (n <= 1) {
            return n;
        }
        if (memo.containsKey(n)) {
            return memo.get(n);
        }
        int result = fibonacci(n - 1) + fibonacci(n - 2);
        memo.put(n, result);
        return result;
    }
}
```

**Output:**




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

**Source Code:**

```
package lab3hometask2;
public class Lab3HomeTask2 {
    public static void main(String[] args) {
        int number = 12345; // Example input
        int count = countDigits(number);
        System.out.println("Number of digits in " + number + " is: " + count);
    }
    public static int countDigits(int n) {
        if (n == 0) {
            return 0;
        }
        return 1 + countDigits(n / 10);}}
```

**Output:**

```
Output - Lab3HomeTask2 (run)
run:
Number of digits in 12345 is: 5
BUILD SUCCESSFUL (total time: 0 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".

**Source Code:**

```
package lab3hometask3;

public class Lab3HomeTask3 {

    public static void main(String[] args) {

        String str = "madam"; // Example input

        System.out.println(isPalindrome(str) ? "YES" : "NO");

    }

    public static boolean isPalindrome(String str) {

        return isPalindromeHelper(str, 0);

    }

    private static boolean isPalindromeHelper(String str, int left) {

        int right = str.length() - 1 - left;

        return left >= right || (str.charAt(left) == str.charAt(right) && isPalindromeHelper(str, left + 1));

    }

}
```

**Output:**

```
Output - Lab3HomeTask3 (run)
run:
YES
BUILD SUCCESSFUL (total time: 0 seconds)
```

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

**Source Code:**

```
package lab3hometask4;
public class Lab3HomeTask4 {
    public static void main(String[] args) {
        int a = 48; // Example input
        int b = 18; // Example input
        int gcd = gcd(a, b);
        System.out.println("The GCD of " + a + " and " + b + " is: " + gcd);
    }
    public static int gcd(int a, int b) {
        if (b == 0) {
            return a;
        }
        return gcd(b, a % b);
    }
}
```

**Output:**