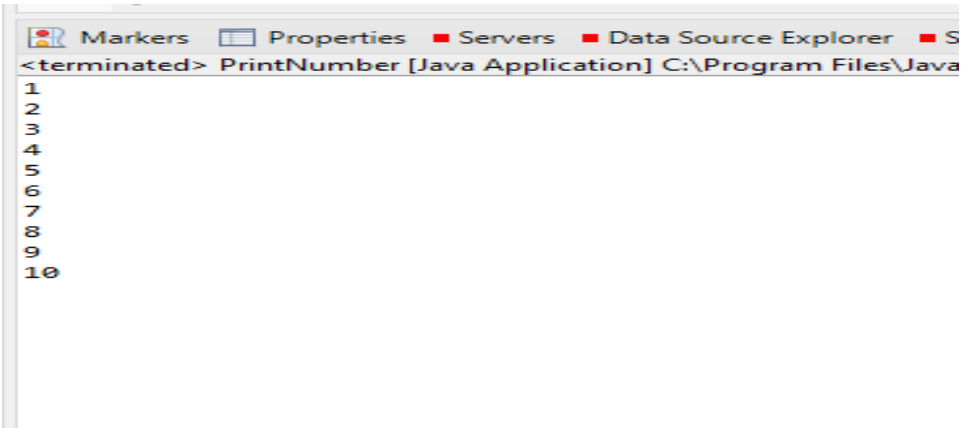
Day 2 : Special Logic Building

1. Print 1 to n without using loops

```
package Assignment2;

public class PrintNumber {

        public static void printNumbers(int n) {
            if (n > 0) {
                printNumbers(n - 1);
                System.out.println(n);
            }

        }


        public static void main(String[] args) {
            int n = 10; // Change n to whatever value you want
            printNumbers(n);
        }

    }
```

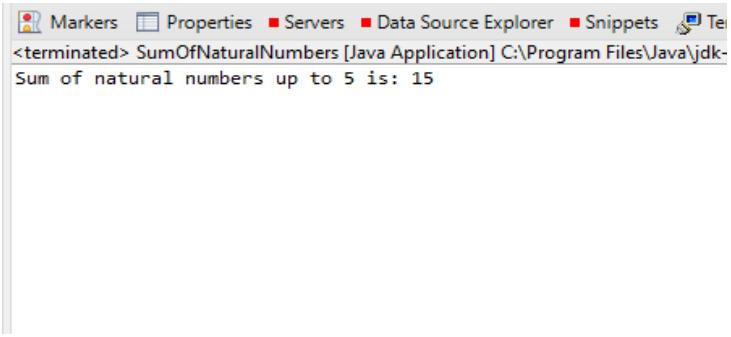


2)

1. Sum of natural numbers using recursion

```
package Assignment2;

public class SumOfNaturalNumbers {

        public static int sumOfNumbers(int n) {
            if (n == 0) {
```

```
            return 0;
        } else {
            return n + sumOfNumbers(n - 1);
        }

    }


    public static void main(String[] args) {
        int n = 5; // Change n to whatever value you want
        int sum = sumOfNumbers(n);
        System.out.println("Sum of natural numbers up to " + n + " is:
" + sum);
    }

}
```



3)

1. Mean of Array using

   Recursion class

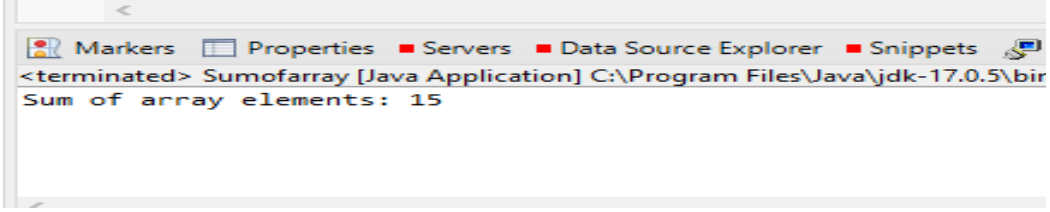
   MeanArray{

```
public static double Mean(int []arr,int n){
if(n==0)return 0.0;
return (Mean(arr,n-1)*(n-1)+arr[n-1])/n;
}
public static void main(String[]args){
int arr[]={1,2,3,4,5};
System.out.println("the mean of the array is "+Mean(arr,arr.length));
}}
```

```
E:\edac_dsa_assignments\Assignment2>java MeanArray
the mean of the array is 3.0
```

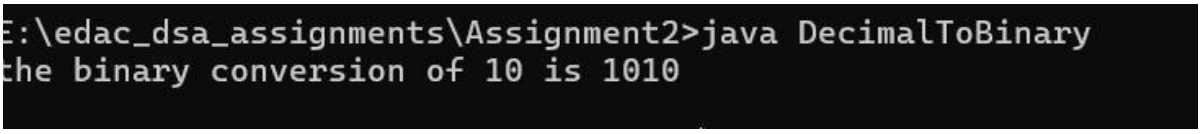
1.1 Sum of array elements using recursion

```
package Assignment2;

public class Sumofarray {

	public static int arraySum(int[] arr, int n) {
	    // Base case: when n is 0, return 0
	    if (n <= 0) {
	        return 0;
	    } else {
	        // Recursive step: add current element and sum of remaining elements
	        return arr[n - 1] + arraySum(arr, n - 1);
	    }
	}

	public static void main(String[] args) {
	    int[] arr = {1, 2, 3, 4, 5}; // Example array
	    int sum = arraySum(arr, arr.length);
	    System.out.println("Sum of array elements: " + sum);
	}
}
```



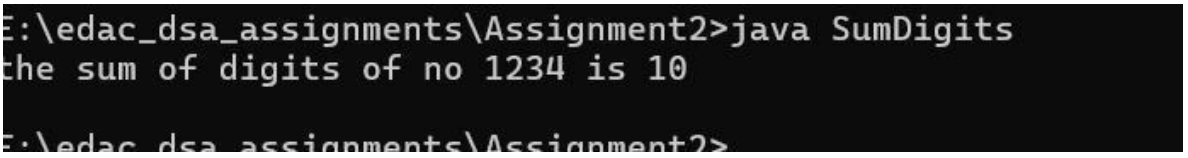
2.Decimal to binary number using recursion

```
class DecimalToBinary{
public static int Binary(int n){
if(n==0)return 0;
return Binary(n/2)*10+n%2;
}
public static void main(String[]args){
int n=10;
System.out.println("the binary conversion of "+n+ " is "+Binary(n));
}}
```



3. Sum of digit of a number using recursion

```
class SumDigits{
public static int Sum(int n){
if(n==0)return 0;
return Sum(n/10)+n%10;
}
public static void main(String[]args){
int n=1234;
System.out.println("the sum of digits of no "+n+" is "+Sum(n));
}}
```



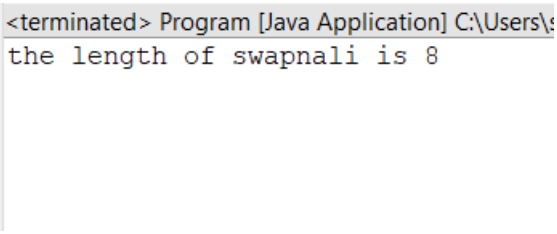
4. Print reverse of a string using recursion

```
class StringReverse{
public static String Reverse(String input){
if( input.length()==0 || input.length()==1){return input;}
return Reverse(input.substring(1))+input.charAt(0);
}
public static void main(String[]args){
String str="swapnali";
System.out.println("the rev of "+str+"is "+Reverse(str));
}}
```

```
<terminated> Program [Java Application] C:\Users\
the rev of swapnali is ilanpaws
```

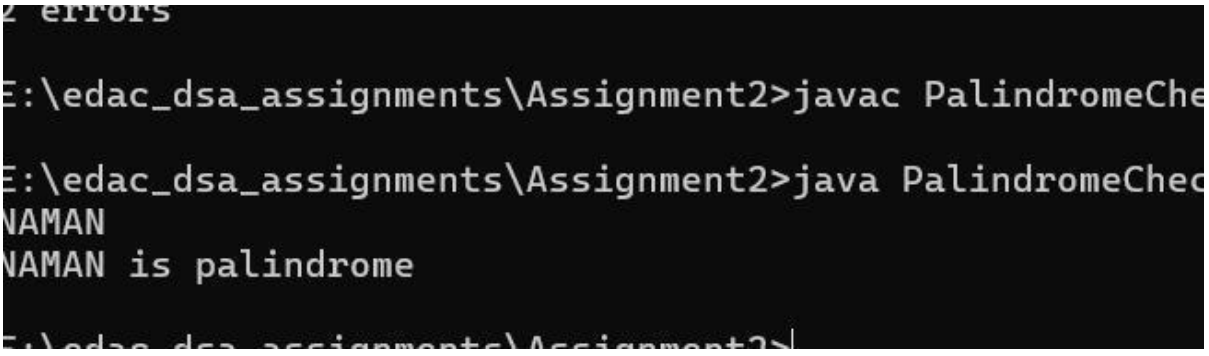
5. Program for length of a string using recursion

```
class StringLength{
public static int Length(String str){
if(str.length()==0){return 0;}
return Length(str.substring(1))+1;
}
public static void main(String[]args){

String str="swapnali";
System.out.println("the length of "+ str +" is "+Length(str));
}}
```



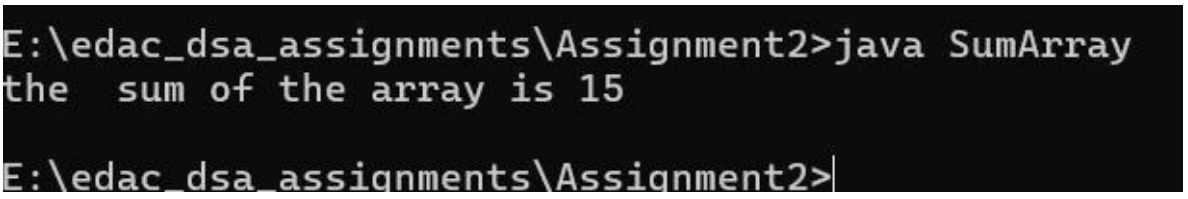
6. Recursive function to check if a string is palindrome

```
import java.util.Scanner;
class PalindromeChecker{
public static boolean isPalindrome(String input){
if(input.length()==0 || input.length()==1){return true;}
if(input.charAt(0)!=input.charAt(input.length()-1)){
return false;}
return isPalindrome(input.substring(1,input.length()-1));
}
public static void main(String[]args){
Scanner sc=new Scanner(System.in);
String str=sc.next();
if(isPalindrome(str)){
System.out.println(str + " is palindrome " );}
else{
System.out.println(str + " is not palindrome " );
}
}}
```



7. Tail recursion to calculate sum of array elements.

```
class SumArray{
public static int Sum(int []arr,int n){
if(n==0){return 0;}
int sSum=Sum(arr,n-1);
return sSum+arr[n-1];
}
public static void main(String[]args){
int arr[]={1,2,3,4,5};
System.out.println("the  sum of the array is "+Sum(arr,arr.length));


}}
```



8. Print Fibonacci Series in reverse order using Recursion

```
import java.util.*;
class PrintFibonacciReverse{
public static List<Integer>list=new ArrayList<>();
public static List<Integer> printFib(int n){
if(n==1){list.add(0);return list;}
if(n==2){list.add(0);list.add(1);return list;}
List<Integer>result=printFib(n-1);
int fLast=result.get(result.size()-1);
int sLast=result.get(result.size()-2);
int last=fLast+sLast;
if(last<n){

result.add(last);
}
return result;

}
public static void printReverse(List<Integer>list){
if(list.size()==0)return;
Integer val=list.get(0);
list.remove(val);
printReverse(list);
System.out.print(val+" ");

}
```

```
public static void main(String[]args){
Scanner sc=new Scanner(System.in);
int n=sc.nextInt();
List<Integer>res=printFib(n);
printReverse(res);
}}
```