Recursion Assignment

Q1 : Given an integer, find out the sum of its digits using recursion.
Solution:

```
import java.util.Scanner;

public class SumOfDigits {
    public static int Sum(int num){
        if (num==0)
            return num;
        else
            return (num%10 + Sum(num/10));
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter Number: ");
        int n=sc.nextInt();
        int result=Sum(n);
        System.out.println("Sum of digits of number is: "+result);
    }
}
```

Enter Number:
1234
Sum of digits of number is: 10

Q2: Given a number n. Find the sum of natural numbers till n but with alternate signs. That means if n = 5 then you have to return 1-2+3-4+5 = 3 as your answer.
Constraints : 0<=n<=1e6
Solution:

```
import java.util.Scanner;

public class AlternateSeries {
    public static int series(int n){
        if(n==0)
            return n;
        else if (n%2==0)
            return series(n-1)-n;
        else
            return series(n-1)+n;
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter number: ");
        int n=sc.nextInt();
        int result=series(n);
        System.out.println("The result of series is: "+result);
    }
}
```

Enter number:
5
The result of series is: 3

Q3: Print the max value of the array [13, 1, -3, 22, 5].
Solution:

```
public class MaxVal {
    public static void main(String[] args) {
        int[] arr = {13, 1, -3, 22, 5};
        int n = arr.length;
        int result = findMax(arr, n, 0);
        System.out.println("Max value of the array is: " + result);
    }

    public static int findMax(int[] arr, int len, int idx) {
        if (idx == len - 1)
            return arr[idx];
        else {
            int max = findMax(arr, len, idx + 1);
            return Math.max(arr[idx], max);
        }
    }
}
```

Max value of the array is: 22

Q4 : Find the sum of the values of the array [92, 23, 15, -20, 10].
Solution:

```
public class SumOfElements {
    public static void main(String[] args) {
        int[] arr={92,23,15,-20,10};
        int n= arr.length;
        int result=sumCalc(arr,n-1);
        System.out.println("Sum of Array Elements is: "+result);

    }

    public static int sumCalc(int[] arr, int idx) {
        if(idx==0)
            return arr[idx];
        else
            return arr[idx]+sumCalc(arr,idx-1);
    }
}
```

Sum of Array Elements is: 120

Q5. Given a number n. Print if it is an armstrong number or not. An armstrong number is a number if the sum of every digit in that number raised to the power of total digits in that number is equal to the number. Example : 153 = 1^3 + 5^3 + 3^3 = 1 + 125 + 27 = 153 hence 153 is an armstrong number. (Easy)
Solution:

```
import java.util.Scanner;

public class ArmstrongFind {
    public static void main(String[] args) {
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter number: ");
        int n=sc.nextInt();
        int temp=n;
        int digit=0;
        while(temp>0){
            if (temp%10!=0)
```

```
                digit++;
            temp/=10;
        }
        if (n==isArmstrong(n,digit))
            System.out.println("Yes");
       else
            System.out.println("No");
    }
    public static int isArmstrong(int num,int digit){
        if(num==0)
            return 0;
        else
            return power(num%10,digit)+isArmstrong(num/10,digit);
    }
    public static int power(int a, int b) {
        if(b==0)
            return 1;
        else if (a%2==0)
            return power(a,b/2)*power(a,b/2);
        else
            return a*power(a,b/2)*power(a,b/2);
    }
}
```

Enter number:
153
Yes