Q1 : Given an integer, find out the sum of its digits using recursion.
Input: n= 1234
Output: 10
Explanation: 1+2+3+4=10

```
public class Main {
    public static int sumOfDigits(int n) {
        // Base case:
        if (n == 0) {
            return 0;
        }

        // Recursive call
        return n % 10 + sumOfDigits(n / 10);
    }

    public static void main(String[] args) {
        int n = 1234;
        System.out.println(sumOfDigits(n));
    }
}
```

Output :
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
Input1 : n = 10
Output 1 : -5
Explanation : 1-2+3-4+5-6+7-8+9-10 = -5
Input 2 : n = 5
Output 2 : 3

```
class Main {
    public static void main(String[] args) {
        int n = 10;

        int result = alternateSum(n, 1, 1);
        System.out.println("result : " + result);
    }

    public static int alternateSum(int n, int currNum, int sign) {
        if(currNum > n)
            return 0;

        return currNum*sign + alternateSum(n, currNum+1, -1*sign);
    }
}
```

Output :
result : 5

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

```
class Main {
   public static void main(String[] args) {
      int[] arr = new int[] { 13, 1, -3, 22, 5};

      int max = getMax(arr, 0);
      System.out.println("Maximum num is : " + max);
   }

   public static int getMax(int[] arr, int ind) {
      int n = arr.length;

      if(ind == n)
         return Integer.MIN_VALUE;

      return Math.max(arr[ind], getMax(arr, ind+1));
   }
}
```

Output :
Maximum num is : 22

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

```
class Main {
    public static void main(String[] args) {
        int[] arr = new int[] { 92, 23, 15, -20, 10};

        int sum = getSum(arr, 0);
        System.out.println("sum is : " + sum);
    }

    public static int getSum(int[] arr, int ind) {
        int n = arr.length;

        if(ind == n)
            return 0;

        return arr[ind] + getSum(arr, ind+1);
    }
}
```

Output :

```
sum 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)
Input1 : 153
Output1 : Yes
Input 2 : 134
Output2 : No

```
class Main {
    public static void main(String[] args) {
        int n = 153;

        int sum = getCubicSum(n);

        if(sum == n)
            System.out.println("Armstrong Number");
        else
            System.out.println("Not a Armstrong Number");
    }

    public static int getCubicSum(int n) {
        if(n==0)
            return 0;

        return (int) Math.pow(n%10, 3) + getCubicSum(n/10);
    }
}


// output :
N = 153, armstrong number
N = 100, not a armstrong number
N = 134, not a  armstrong number
```