**RECURSION IN JAVA**

**ASSIGNMENT**

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

Input: n= 1234

Output: 10

Explanation: 1+2+3+4=10

Ans:

**Program:**

import java.io.\*;

class SumOfDigit

{

static int SumOfDigit(int n)

{

if (n == 0)

return 0;

return (n % 10 + SumOfDigit(n / 10));

}

public static void main(String args[])

{

int num = 1234;

int result = SumOfDigit(num);

System.out.println("Sum of digits in " +num + " is " + result);

}

}

//Output:

// Sum of digits in 1234 is 10

Time complexity : O(log n)

Space complexity : O(log n)

If(n==0)

return 0

1%10=1

+

12%10=2

+

123%10=3

+

1234%10=4

+

return 10

Q2: Given a number n. Find the sum of natural numbers till n but with alternate sign

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

Ans:

import java.util.\*;

class AlternateNatural

{

static int AlterSign(int N)

{

int alternateSum = 0;

for(int i = 1; i <= N; i++)

{

if (i % 2 == 0)

{

alternateSum += -i;

}

else

{

alternateSum += i;

}

}

return alternateSum;

}

public static void main(String[] args)

{

int N = 10;

System.out.print(AlterSign(N));

}

}

//Output: -5

Time Complexity: O(N)   
 Space Complexity: O(1)

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

Ans:

Method 1 (Linear Search): We can traverse the array and keep track of maximum and element. And finally return the maximum element.

class maximumValue

{

static int findMaximum(int arr[], int low, int high)

{

int max = arr[low];

int i;

for (i = low; i <= high; i++)

{

if (arr[i] > max)

max = arr[i];

}

return max;

}

public static void main (String[] args)

{

int arr[] = {13, 1, -3, 22, 5};

int n = arr.length;

System.out.println("The maximum element is "+findMaximum(arr, 0, n-1));

}

}

//Output:The maximum element is 22

Time Complexity : O(n)

Space Complexity: O(1)

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

Ans:

class SumArray

{

static int arr[] = {92, 23, 15, -20, 10};

static int sum()

{

int sum = 0;

int i;

for (i = 0; i < arr.length; i++)

sum += arr[i];

return sum;

}

public static void main(String[] args)

{

System.out.println("Sum of given array is "+ sum());

}

}

//Output:Sum of given array is 120

Time Complexity: O(n)  
Space Complexity: O(1)

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

Ans: Time Complexity: O(n).  
Space Complexity: O(1).

public class ArmNum

{

public void isArmstrong(String n)

{

char[] s = n.toCharArray();

int size = s.length;

int sum = 0;

for (char num : s)

{

int temp = 1;

int i= Integer.parseInt(Character.toString(num));

for (int j = 0; j <= size - 1; j++)

{

temp \*= i;

}

sum += temp;

}

if (sum == Integer.parseInt(n))

{

System.out.println("Yes");

}

else

{

System.out.println("No");

}

}

public static void main(String[] args)

{

ArmNum am = new ArmNum();

am.isArmstrong("153");

am.isArmstrong("134");

}

}

//Output:

// Yes

// No