Recursion in Java Assignment Questions

Assignment Questions:

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 → public class DigitSum {

public static int digitSum(int n) {

if (n < 10) {

return n;

} else {

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

}

}

public static void main(String[] args) {

int n = 1234;

int result = digitSum(n);

System.out.println(result);

}

}

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

Ans → public class AlternateSum {

public static int findAlternateSum(int n) {

int sum = 0;

for (int i = 1; i <= n; i++) {

if (i % 2 == 0) {

sum -= i;

} else {

sum += i;

}

}

return sum;

}

public static void main(String[] args) {

int n = 10;

int result = findAlternateSum(n);

System.out.println(result);

n = 5;

result = findAlternateSum(n);

System.out.println(result);

}

}

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

Ans → public class MaxValue {

public static int findMax(int[] arr, int start, int end) {

if (start > end) {

return Integer.MIN\_VALUE;

} else if (start == end) {

return arr[start];

} else {

int mid = (start + end) / 2;

int leftMax = findMax(arr, start, mid);

int rightMax = findMax(arr, mid + 1, end);

int max = Math.max(leftMax, rightMax);

return Math.max(max, arr[mid]);

}

}

public static void main(String[] args) {

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

int result = findMax(arr, 0, arr.length - 1);

System.out.println(result);

}

}

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

Ans → public class ArraySum {

public static int arraySum(int[] arr, int start, int end) {

if (start > end) {

return 0;

} else if (start == end) {

return arr[start];

} else {

int mid = (start + end) / 2;

int leftSum = arraySum(arr, start, mid);

int rightSum = arraySum(arr, mid + 1, end);

int sum = leftSum + rightSum;

return sum + arr[mid];

}

}

public static void main(String[] args) {

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

int result = arraySum(arr, 0, arr.length - 1);

System.out.println(result);

}

}

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 → import java.util.Scanner;

public class ArmstrongNumber {

public static boolean isArmstrong(int n) {

int originalNumber = n;

int numberOfDigits = String.valueOf(n).length();

int sum = 0;

while (n > 0) {

int digit = n % 10;

sum += Math.pow(digit, numberOfDigits);

n /= 10;

}

return originalNumber == sum;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter a number: ");

int n = scanner.nextInt();

if (isArmstrong(n)) {

System.out.println("Yes, it is an Armstrong number.");

} else {

System.out.println("No, it is not an Armstrong number.");

}

}

}