1. Print 1 to n without using loops

import java.io.\*;

import java.util.\*;

public class a

{

static void printNos(int n)

{

if (n > 0) {

printNos(n - 1);

System.out.print(n + " ");

}

return;

}

public static void main(String[] args)

{

int n = 10;

printNos(n);

}

}

C:\Users\mujah\Downloads\Algorithm and Data Structure>java a

1 2 3 4 5 6 7 8 9 10

2. Sum of natural numbers using recursion

class SumofNatural{

static int sum(int n){

if (n >= 10){ //1< 0--f

return n;

}

else{

return n+sum(n+1); //1+sum(1+1)---1+1---2 go to int n= 2

}

}

public static void main (String arg[]){

int result = sum (1);

System.out.println(result);

}

}

C:\Users\mujah\Downloads>java SumofNatural

55

3. Sum of array elements using recursion

public class SumofArray{

static int displaySum(int arr[], int n){

if (n <= 0)

return n;

else

return displaySum(arr, n-1)+ arr[n-1]; }

public static void main(String arg[]){

int arr []= {1, 2, 3, 4};

int n = arr.length;

System.out.println(displaySum(arr, n));

}

}

C:\Users\mujah\Downloads>java SumofArray

10

4. Decimal to binary number using recursion

import java.io.\*;

import java.util.\*;

public class a

{

static int find(int decimal\_number)

{

if (decimal\_number == 0)

return 0;

else

return (decimal\_number % 2 + 10 \*

find(decimal\_number / 2));

}

public static void main(String args[])

{

int decimal\_number = 10;

System.out.println(find(decimal\_number));

}

}

C:\Users\mujah\Downloads\Algorithm and Data Structure>java a

1010

5. Sum of digit of a number using recursion

import java.io.\*;

import java.util.\*;

public class a

{

static int sum\_of\_digit(int n)

{

if (n == 0)

return 0;

return (n % 10 + sum\_of\_digit(n / 10));

}

public static void main(String args[])

{

int num = 12345;

int result = sum\_of\_digit(num);

System.out.println("Sum of digits in " +

num + " is " + result);

}

}

C:\Users\mujah\Downloads\Algorithm and Data Structure>java a

Sum of digits in 12345 is 15

6. Print reverse of a string using recursion

import java.io.\*;

import java.util.\*;

public class a

{

public static String reverse(String str, int len) {

if (len < 1) {

return "";

}

if (len == 1) {

return String.valueOf(str.charAt(0));

}

return str.charAt(len - 1) + reverse(str, len - 1);

}

public static void main(String[] args) {

String str = "I am Ayesha from Mumbai";

System.out.println(reverse(str, str.length()));

}

}

C:\Users\mujah\Downloads\Algorithm and Data Structure>java a

iabmuM morf ahseyA ma I

7. Program for length of a string using recursion

import java.io.\*;

import java.util.\*;

public class a

{

private static int recLen(String str)

{

if (str.equals(""))

return 0;

else

return recLen(str.substring(1)) + 1;

}

public static void main(String[] args)

{

String str ="IAMAYESHA";

System.out.println(recLen(str));

}

}

C:\Users\mujah\Downloads\Algorithm and Data Structure>java a

9

8. Tail recursion to calculate sum of array elements.

class a

{

static int arrSum(int []array, int size, int sum)

{

if (size == 0)

return sum;

return arrSum(array, size - 1, sum + array[size - 1]);

}

public static void main(String[] args)

{

int array[] = { 2, 9, 1, 7 };

int size = array.length;

System.out.print(arrSum(array, size, 0));

}

}

C:\Users\mujah\Downloads\Algorithm and Data Structure>java a

19

9. Recursive function to check if a string is palindrome

class a

{

static boolean isPalRec(String str, int s, int e)

{

if (s == e)

return true;

if ((str.charAt(s)) != (str.charAt(e)))

return false;

if (s < e + 1)

return isPalRec(str, s + 1, e - 1);

return true;

}

static boolean isPalindrome(String str)

{

int n = str.length();

if (n == 0)

return true;

return isPalRec(str, 0, n - 1);

}

public static void main(String args[])

{

String str = "madam";

if (isPalindrome(str))

System.out.println("Yes");

else

System.out.println("No");

}

}

C:\Users\mujah\Downloads\Algorithm and Data Structure>java a

Yes

10. Print Fibonacci Series in reverse order using Recursion

class a

{

static void reverseFibonacci(int n)

{

int a[] = new int[n];

a[0] = 0;

a[1] = 1;

for (int i = 2; i < n; i++)

{

a[i] = a[i - 2] + a[i - 1];

}

for (int i = n - 1; i >= 0; i--)

{

System.out.print(a[i] +" ");

}

}

public static void main(String[] args)

{

int n = 10;

reverseFibonacci(n);

}

}

C:\Users\mujah\Downloads\Algorithm and Data Structure>java a

34 21 13 8 5 3 2 1 1 0