Day 2 : Special Logic Building Assignment: 10 special Recursion Programs

**1.Print 1 to n without using loops**

public class PrintNumbers {

public static void printNumbers(int n) {

if (n > 0) {

printNumbers(n - 1);

System.out.println(n);

}

}

public static void main(String[] args) {

int n = 10; // Change this to whatever value of n you want

System.out.println("Printing numbers from 1 to " + n + ":");

printNumbers(n);

}

}

Output:-

Printing numbers from 1 to 10:

1

2

3

4

5

6

7

8

9

10

**2. Sum of natural numbers using recursion**

public class SumOfNaturalNumbers {

public static int sum(int n) {

if (n == 0) {

return 0;

} else {

return n + sum(n - 1);

}

}

public static void main(String[] args) {

int n = 5; // Change this to whatever value of n you want

int result = sum(n);

System.out.println("Sum of natural numbers from 1 to " + n + " is: " + result);

}

}

Sum of natural numbers from 1 to 5 is: 15

**3. Mean of Array using Recursion**

public class MeanOfArray {

public static double mean(int[] arr, int index) {

if (index == arr.length - 1) {

return arr[index];

} else {

return arr[index] + mean(arr, index + 1);

}

}

public static void main(String[] args) {

int[] array = {5, 10, 15, 20, 25}; // Example array

double sum = mean(array, 0);

double mean = sum / array.length;

System.out.println("Mean of the array is: " + mean);

}

}

Output:- Mean of the array is: 15.0

**4. Sum of array elements using recursion**

public class SumOfArrayElements {

public static int sum(int[] arr, int index) {

if (index == arr.length) {

return 0;

} else {

return arr[index] + sum(arr, index + 1);

}

}

public static void main(String[] args) {

int[] array = {5, 10, 15, 20, 25}; // Example array

int result = sum(array, 0);

System.out.println("Sum of array elements is: " + result);

}

}

Output:- Sum of array elements is: 75

**5. Decimal to binary number using recursion**

public class DecimalToBinary {

public static String decimalToBinary(int n) {

if (n == 0) {

return "0";

} else if (n == 1) {

return "1";

} else {

return decimalToBinary(n / 2) + (n % 2);

}

}

public static void main(String[] args) {

int decimalNumber = 25; // Example decimal number

String binaryNumber = decimalToBinary(decimalNumber);

System.out.println("Binary representation of " + decimalNumber + " is: " + binaryNumber);

}

}

Output:-Binary representation of 25 is: 11001

**6. Sum of digit of a number using recursion**

public class SumOfDigits {

public static int sumOfDigits(int number) {

if (number < 10) {

return number;

} else {

return number % 10 + sumOfDigits(number / 10);

}

}

public static void main(String[] args) {

int num = 12345; // Example number

int sum = sumOfDigits(num);

System.out.println("Sum of digits of " + num + " is: " + sum);

}

}

Output:- Sum of digits of 12345 is: 15

**7. Print reverse of a string using recursion**

public class ReverseString {

public static void reverse(String str) {

if (str == null || str.length() == 0) {

return;

} else {

System.out.print(str.charAt(str.length() - 1)); // Print last character

reverse(str.substring(0, str.length() - 1)); // Recur with substring excluding last character

}

}

public static void main(String[] args) {

String input = "Hello, world!"; // Example string

System.out.print("Reverse of \"" + input + "\" is: ");

reverse(input);

}

}

Output:- Reverse of "Hello, world!" is: !dlrow ,olleH

**8. Program for length of a string using recursion**

public class StringLength {

public static int length(String str) {

if (str == null || str.isEmpty()) {

return 0;

} else {

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

}

}

public static void main(String[] args) {

String input = "Hello, world!"; // Example string

int length = length(input);

System.out.println("Length of \"" + input + "\" is: " + length);

}

}

Output:- Length of "Hello, world!" is: 13

**9. Tail recursion to calculate sum of array elements.**

public class SumOfArrayElements {

public static int sum(int[] arr) {

return sumHelper(arr, 0, 0);

}

private static int sumHelper(int[] arr, int index, int acc) {

if (index == arr.length) {

return acc;

} else {

return sumHelper(arr, index + 1, acc + arr[index]);

}

}

public static void main(String[] args) {

int[] array = {5, 10, 15, 20, 25}; // Example array

int result = sum(array);

System.out.println("Sum of array elements is: " + result);

}

}

Output:-Sum of array elements is: 75

**10. Recursive function to check if a string is palindrome**

public class PalindromeChecker {

public static boolean isPalindrome(String str) {

// Base cases

if (str == null || str.length() <= 1) {

return true;

} else {

// Check if first and last characters are equal

if (str.charAt(0) == str.charAt(str.length() - 1)) {

// Recur with substring excluding first and last characters

return isPalindrome(str.substring(1, str.length() - 1));

} else {

return false;

}

}

}

public static void main(String[] args) {

String palindrome = "radar"; // Example palindrome

String nonPalindrome = "hello"; // Example non-palindrome

System.out.println("\"" + palindrome + "\" is a palindrome: " + isPalindrome(palindrome));

System.out.println("\"" + nonPalindrome + "\" is a palindrome: " + isPalindrome(nonPalindrome));

}

}

Output:-

"radar" is a palindrome: true

"hello" is a palindrome: false

**11. Print Fibonacci Series in reverse order using Recursion**

public class ReverseFibonacci {

public static void reverseFibonacci(int n) {

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

System.out.print(fibonacci(i) + " ");

}

}

public static int fibonacci(int n) {

if (n <= 1) {

return n;

} else {

return fibonacci(n - 1) + fibonacci(n - 2);

}

}

public static void main(String[] args) {

int count = 10; // Number of Fibonacci numbers to print

System.out.println("Fibonacci series in reverse order:");

reverseFibonacci(count);

}

}

Output:- Fibonacci series in reverse order:

34 21 13 8 5 3 2 1 1 0