Lab Assignment 9

Recursive Method

CSE110: Programming Language I

No of Tasks	Points to Score
10	**100**

Task01
[A,B,C should be written in a single java file]

A. Write a method called **oneToN** that prints 1 till N recursively.

Hint: N is a number taken as input from the user and you need to print the numbers starting from 1 to N recursively.

Sample Input	Sample Method Call	Output
N = 5	oneToN(1,N);	1 2 3 4 5
N = 11	oneToN(1,N);	1 2 3 4 5 6 7 8 9 10 11

B. Write a method **nToOne** that prints from N to 1 recursively.

Hint: N is a number taken as input from the user and you need to print the numbers starting from N to 1.

Sample Input	Sample Method Call	Output
N = 6	nToOne(1,N);	6 5 4 3 2 1
N = 3	nToOne(1,N);	3 2 1

C. Write a method called **recursiveSum** to sum till N recursively.

Hint: N is a number taken as input from the user and you need to add the numbers starting from 1 to N recursively and print the sum.

Sample Input	Sample Method Call	Output
N = 4	System.out.println(recursiveSum(1,N));	10
N = 12	System.out.println(recursiveSum(1,N));	78

Task02

Write a **recursive method** called **reverseDigits** that takes an integer n as an argument and prints the digits of n in reverse order.

Hint: Think about how you solved it using loop

Sample Input	Sample Method Call	Output
n = 12345	reverseDigits(n)	5 4 3 2 1
n = 649	reverseDigits(n)	9 4 6
n = 1000	reverseDigits(n)	0 0 0 1

Task03

Write a **recursive method** called **sumDigits** that takes an integer n as an argument and sums up the digits of n then **returns** the result.

Hint: Think about how you would solve it using loop

Sample Input	Sample Method Call	Output
n = 12345	int x = sumDigits(n); System.out.println(x);	15
n = 649	int x = sumDigits(n); System.out.println(x);	19

Task04

Write a **recursive method** called **reverse_string(s)** that returns the reverse of a given string **s**.

Sample Method Call	Output
System.out.println(reverse_string("Hello", 0))	olleH
System.out.println(reverse_string("swan", 0))	naws

Task05

Write a **recursive method** called **factorial(n)** that returns the factorial of a number **n**. Assume n >= 0.

Sample Input	Sample Method Call	Output
n = 5	int x = factorial(n) System.out.println(x)	120
n = 7	int x = factorial(n) System.out.println(x)	5040

Task06

Write a **recursive method** called **power(base, exponent)** that calculates base raised to the power of exponent (assume exponent is a non-negative integer).

Sample Method Call	Output
int x = power(5,3) System.out.println(x)	125
int x = power(8,4) System.out.println(x)	4096

Task07

Write a **recursive method** called **print_elements(arr, index)** that prints elements of an array starting from index to the end.

Given Array and Input	Sample Method Call	Output
int[] arr = {5,6,2,1,8,7}; int index = 2	print_element(arr,index)	2 1 8 7
int[] arr = {13,12,19,21,31,55}; int index = 0	print_element(arr,index)	13 12 19 21 31 55

Task08

1 The Fibonacci sequence is a series of numbers that starts with 0 & 1 and the rest of the numbers are generated by adding the immediate two numbers before it. It goes like this: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144 and so on.
In short, fibonacci(0) = 0, fibonacci(1) = 1 and fibonacci(n) = fibonacci(n-1) + fibonacci(n-2) .

Sample Method Call	Output
System.out.println(fibonacci(0))	0
System.out.println(fibonacci(1))	1
System.out.println(fibonacci(5))	5
System.out.println(fibonacci(9))	34

Task09

Trace the following code to generate the outputs. Show the necessary trace table.

1	`public class ClassWork1{`
2	`    public static int calculate(int n) {`
3	`        if (n <= 0){`
4	`            return 4;`
5	`        }`
6	`        else if (n % 2 != 0){`
7	`            return n + calculate(n - 1);`
8	`        }`
9	`        else{`
10	`            return n * calculate(n - 2);`
11	`        }`
12	`    }`
13	`    public static void main(String[] args) {`
14	`        int result = calculate(8);`
15	`        System.out.println(result);`
16	`        int result2 = calculate(5);`
17	`        System.out.println(result2);`
18	`    }`
19	`}`

Task10

Trace the following code to generate the outputs. Show the necessary trace table.

```
1  public class ClassWork2{
2    public static String fun(String s, int n){
3      if(s.length()==4){
4        return n+s+n;
5      } else if(n%2==0){
6        System.out.println(s+n+n+3);
7        return fun(s+n, n+3);
8      } else {
9        System.out.println(s+n+(n-1));
10       return fun(s+n, n-1);
11     }
12   }
13   public static void main(String[] args){
14     String s = fun("",1);
15     System.out.println(s);
16   }
17 }
```