

Statement Purpose:

Purpose of this Lab is to familiarize the students with the use and power of recursion in Java. They will learn what types of problems can be solved using recursion. They will also be taught what are the merits and demerits of using recursion. The students will learn how to solve the problems like finding factorial of any number, binary search, and merge sort using recursion.

Activity Outcomes:

After completing the activities given at the end of this lab material, the students will learn how to write simple Java programs using recursion. They will understand how to use recursion for solving various problems like

- Finding sum of all even integers from 1 to any given number
- Finding and returning the difference of all integers from 1 to any given number.
- Reversing the elements of an array without using any extra array
- Displaying an integer number in reverse

Theory Review (10 Minutes):

Recursion

The idea of calling one function from another immediately suggests the possibility of a function calling **itself**. The function-call mechanism in Java supports this possibility, which is known as **recursion**. Recursion is a powerful general-purpose programming technique, and is the key to numerous critically important computational applications, ranging from combinatorial search and sorting methods that provide basic support for information processing to the Fast Fourier Transformation for signal processing.

Types of Problems that can be solved using Recursion

Recursion solves large problems by reducing them to smaller problems of the same form. For example, finding factorial of any number, generating Fibonacci sequence, any number raised to the power of any number, binary search, merge sort, Tower of Hanoi etc.



Advantages and disadvantages of using Recursion

- Recursion reduces the size of code for solving a specific problem as compared to its iterative counterpart.
- Recursion consumes more space in memory than iterative code because of recursive calls which use stack.
- Recursive code is simpler to write but difficult to understand.

Characteristics of a Recursive Method

1. A simple base case which is used to terminate the recursive call.
2. A way of getting our larger problem closer to the base case i.e. a way to chop out part of the larger problem to get a somewhat simpler problem.
3. A recursive call which passes the simpler problem back into the method.

Practice Activity with Lab Instructor: (10 Minutes)

Make a new project in NetBeans called Recursion1Project, and then create a package with the name "recursion" inside it. Then create a main class "SumNums" inside this package.

Your job is to write a recursive code in this class to find the sum of all integers from first to second integer. For example, if first integer is 5 and second integer is 10, your program should find the sum $5+6+7+8+9+10=45$ using recursion.

Solution:

1. Create a project with the name SumNumbersRecursion.
2. Create a package with the name "Recursion" within this project.
3. Create a main class in this package with the name "SumNums".

Before writing the code for this method, let us try to understand this problem.

In order to find the sum of integers from first integer 'a' to second integer 'b' using recursion, we use the following procedure:

Our recursive method should find the sum from 'a' to 'b'.

Suppose first integer 'a' is 5 and second integer 'b' is 10. Then we can divide this larger problem into smaller problem at each step until we reach some base case as follows:



$$\text{SumRec}(5, 10) = 10 + 9 + 8 + 7 + 6 + 5$$

$$= 10 + \text{SumRec}(5, 9)$$

$$= 10 + 9 + \text{SumRec}(5, 8)$$

$$= 10 + 9 + 8 + \text{SumRec}(5, 7)$$

$$= 10 + 9 + 8 + 7 + \text{SumRec}(5, 6)$$

$$= 10 + 9 + 8 + 7 + 6 + \text{SumRec}(5, 5)$$

$$= 10 + 9 + 8 + 7 + 6 + 5$$

$$= 45$$

Moving one step closer towards base case

Moving one step more towards base case

Moving one step more towards base case

Moving one step more towards base case

This is Base Case. When second integer i.e. 'b' becomes equal to first integer i.e. 'a', the method should return first integer i.e. 'a'

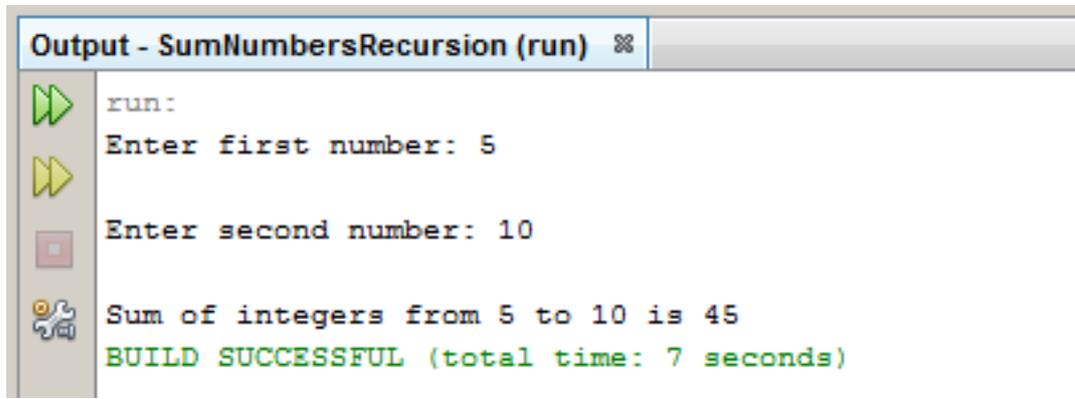
4. Let us write the code for this method. Add the following lines of code within SumNums main class.



```
Start Page  SumNums.java
Source History
1
2 package Recursion;
3 import java.util.Scanner;
4 public class SumNums {
5
6     public static void main(String[] args) {
7         Scanner sc = new Scanner(System.in);
8         System.out.print("Enter first number: ");
9         int x = sc.nextInt();
10        System.out.println();
11        System.out.print("Enter second number: ");
12        int y = sc.nextInt();
13        System.out.println();
14        if (x > y) {
15            int temp = x;
16            x = y;
17            y = temp;
18        }
19
20        System.out.println("Sum of integers from "+x+" to "+y+" is "+SumNums(x,y));
21    }
22
23    public static int SumNums(int a, int b){
24        if (b == a){
25            return a;
26        }
27        else {
28            return b+SumNums(a, b-1);
29        }
30    }
31 }
```

5. Save and Run this main class.
6. When you enter first integer as 5 and second integer as 10, you will get the following output.





```
Output - SumNumbersRecursion (run) ⌵
run:
Enter first number: 5
Enter second number: 10
Sum of integers from 5 to 10 is 45
BUILD SUCCESSFUL (total time: 7 seconds)
```

LAB EXERCISES: (60 Minutes)

1. Add a method named `sumEvenNums()` in the above program which finds and returns the sum of all even integers from 1 to any given number.
2. Add a method named `subNums()` in the above program which finds and returns the difference of all integers from 1 to any given number. For example, if number = 5, the result should be $(1 - 2 - 3 - 4 - 5)$ i.e. -13
3. Add a method named `ArrayReverse()` in the above program which reverses the elements of an array without using any extra array.
4. Add a method named `RevIntRecursive()` in the above program which displays an integer number in reverse.

