NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

INTRODUCTION TO COMPUTING (CS-101)

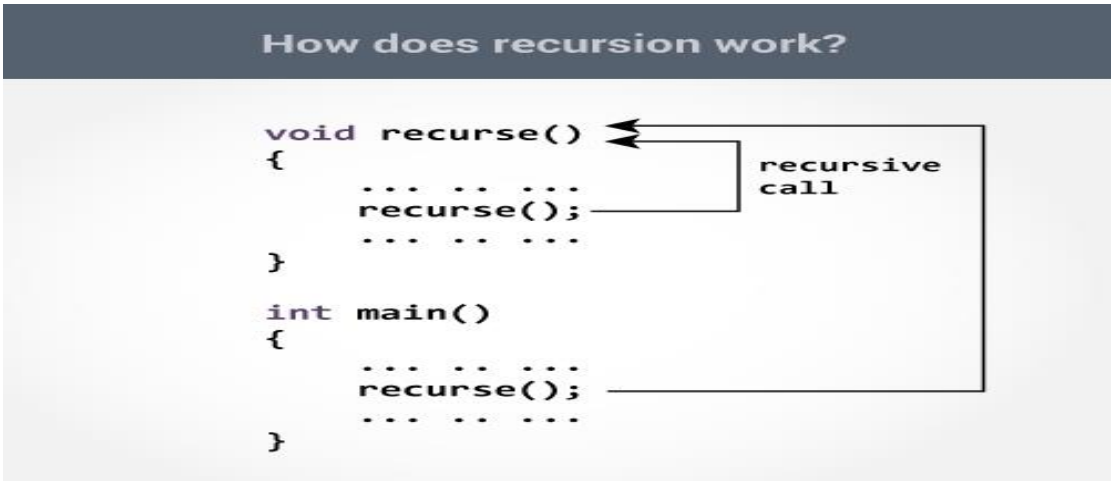
Rahemeen.khan || Majid Hussain
rahemeen.khan@nu.edu.pk || majid.hussain@nu.edu.pk

LAB # 04

Recursion Function

A function that calls itself is known as a recursive function. And, this technique is known as recursion.

How recursion works?



The recursion continues until some condition is met to prevent it.

To prevent infinite recursion, **if...else statement** (or similar approach) can be used where one branch makes the recursive call and other doesn't.

Factorial (5) = 5*factorial (4)
Factorial (4) = 4*factorial (3)
Factorial (3) = 3*factorial (2)

Factorial (2) = 2*factorial (1)
Factorial (1) = 1
n!=1 (for n=0)
n!=n*(n-1)! (For n>0)

Example: Sum of Natural Numbers Using Recursion:

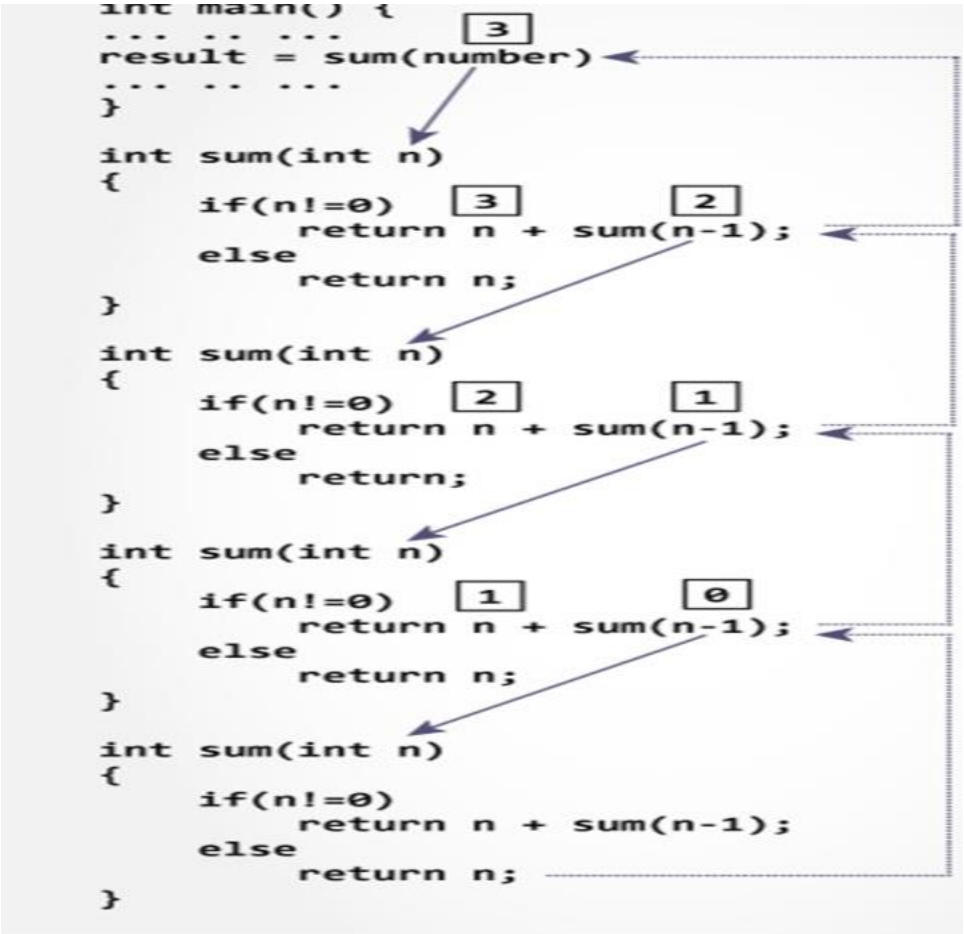
```c
#include <stdio.h>
int sum(int n);

int main()
{
    int number, result;

    printf("Enter a positive integer: ");
    scanf("%d", &number);

    result = sum(number);

    printf("sum=%d", result);
}

int sum(int num)
{
    if (num!=0)
        return num + sum(num-1); // sum() function calls itself
    else
        return num;
}
```

Output

Enter a positive integer:
3

Initially, the sum() is called from the main() function with number passed as an argument.Suppose, the value of num is 3 initially. During next function call, 2 is passed to the sum()function. This process continues until num is equal to 0.When num is equal to 0, the if condition fails and the else part is executed returning the sum of integers to the main() function.

How recursion works step by step:



Advantages and Disadvantages of Recursion

Recursion makes program elegant and cleaner. All algorithms can be defined recursively which makes it easier to visualize and prove.

If the speed of the program is vital then, you should avoid using recursion. Recursions use more memory and are generally slow. Instead, you can use loop.

TASK # 01

(a)Write a program to Print Fibonacci Series using recursion function.

(b)Write a program to count the digits of a given number using recursion function.

TASK # 02

(a)Write a program to GCD of two numbers using recursion function.

HINT: The GCD of 24 and 60 is **2** × **2** × **3** = **12**.

(b)Write a program to find Factorial of a Number Using Recursion.

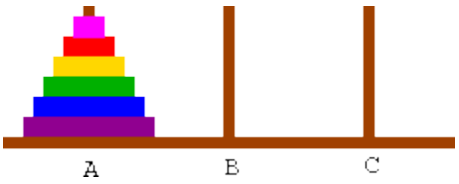
TASK # 03

Program to Solve Tower-of-Hanoi Problem using Recursion

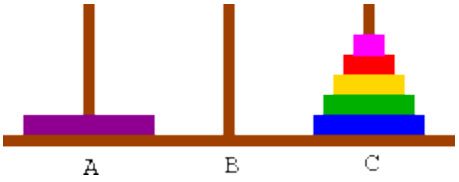
This C Program uses recursive function & solves the tower of hanoi. The tower of hanoi is a mathematical puzzle. It consists of threerods, and a number of disks of different sizes which can slideonto any rod. The puzzle starts with the disks in a neat stack in ascending order of size on one rod, the smallest at the top. We have to obtain the same stack on the third rod.

In our Towers of Hanoi solution, we recurse on the largest disk to be moved. That is, we will write a recursive function that takes as a parameter the disk that is the largest disk in the tower we want to move. Our function will also take three parameters indicating from which peg the tower should be moved (*source*), to which peg it should go (*destination*), and the other peg, which we can use temporarily to make this happen (*spare*).

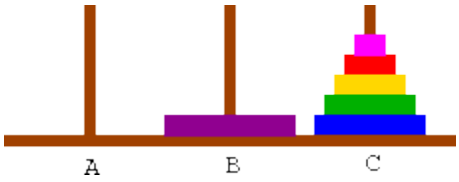
At the top level, we will want to move the entire tower, so we want to move disks 5 and smaller from peg A to peg B. We can break this into three basic steps.



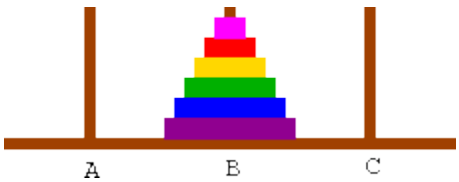
1. Move disks 4 and smaller from peg A (*source*) to peg C (*spare*), using peg B (*dest*) as a spare. How do we do this? By recursively using the same procedure. After finishing this, we'll have all the disks smaller than disk 4 on peg C.



2. Now, with all the smaller disks on the spare peg, we can move disk 5 from peg A (*source*) to peg B (*destination*).



3. Finally, we want disks 4 and smaller moved from peg C (*spare*) to peg B (*destination*). We do this recursively using the same procedure again. After we finish, we'll have disks 5 and smaller all on *destination*.



In pseudocode, this looks like the following. At the top level, we'll call MoveTower with *disk*=5, *source*=A, *dest*=B, and *spare*=C.

```
FUNCTION MoveTower(disk, source, dest, spare):
IF disk == 0, THEN:
move disk from source to dest
ELSE:
MoveTower(disk - 1, source, spare, dest)    // Step 1 above
```

```
move disk from source to dest              // Step 2 above
MoveTower(disk - 1, spare, dest, source)   // Step 3 above
END IF
```

TASK # 04

Using the recursive factorial function, write a program to calculate the number of possible ways of selecting ‘r’ items from a group of ‘n’ items given by the equation: And thus determine the number of possible ways of selecting 3 eggs from a pack of 12.

$$C(n,r)=n! /(r! (n-r)!)$$

TASK # 05

(a)Write a recursive function convert a decimal number into a binary number, printing the binary number in result without using an array.

(b)Write a recursive function that prints the numbers 1…n in descending order: