Structured Programming
CSE 103

Professor Dr. Mohammad Abu Yousuf

RECURSION

- Recursion is a process by which a function calls itself repeatedly, until some specified condition has been satisfied.

Factorial of an integer

```
1.#include <stdio.h>
2.int fact (int);
3.int main()
4.{
5.    int n,f;
6.    printf("Enter the number whose factorial you want to calculate?");
7.    scanf("%d",&n);
8.    f = fact(n);
9.    printf("factorial = %d",f);
10.}
11.int fact(int n)
12.{
13.    if (n==0)
14.        return 0;
15.    else if ( n == 1)
16.        return 1;
17.    else
18.        return n*fact(n-1);
19.}
```

1

Output:
Enter the number whose factorial you want to calculate?5
factorial = 120

Factorial of an integer

```
return 5 * factorial(4) = 120
└── return 4 * factorial(3) = 24
    └── return 3 * factorial(2) = 6
        └── return 2 * factorial(1) = 2
            └── return 1 * factorial(0) = 1
```

1*2*3*4*5 = 120

```
1.#include<stdio.h>
2.int fibonacci(int);
3.void main ()
4.{
5.    int n,f;
6.    printf("Enter the value of n?");
7.    scanf("%d",&n);
8.    f = fibonacci(n);
9.    printf("Output is : %d",f);
10.}
11.int fibonacci (int n)
12.{
13.    if (n==0)
14.    {
15.        return 0;
16.    }
17.    else if (n == 1)
18.    {
19.        return 1;
20.    }
21.    else
22.    {
23.        return fibonacci(n-1)+fibonacci(n-2);
24.    }
25.}
```

Find the nth term of the Fibonacci series

```
Enter the value of n?
12
Output is: 144
```

```
/* read a line of text and write it out backwards, using recursion */

#include <stdio.h>

#define EOLN  '\n'

void reverse(void);          /* function prototype */

main()

{
    printf("Please enter a line of text below\n");
    reverse();
}

void reverse(void)

/* read a line of characters and write it out backwards */

{
    char c;

    if ((c = getchar()) != EOLN) reverse();
    putchar(c);
    return;
}
```

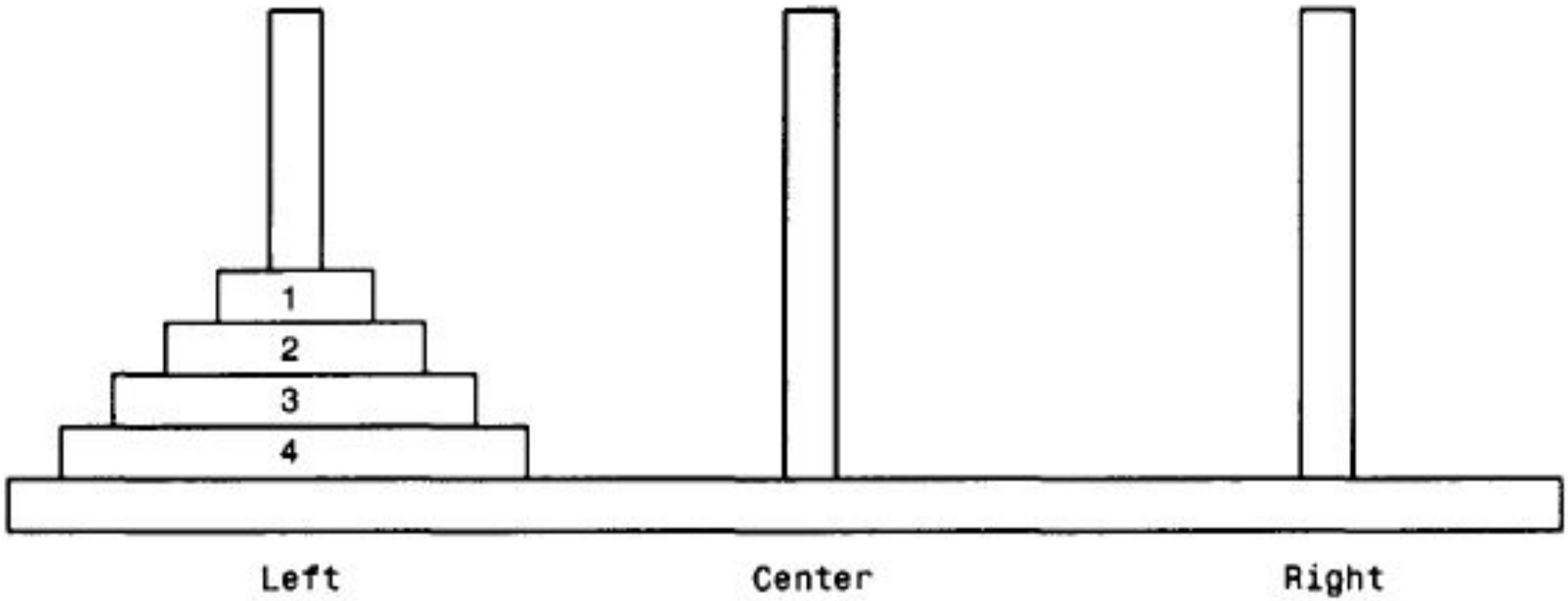
- Output:

  Now is the time for all good men to come to the aid of their country!

  !yrtnuoc r i e h t fo dia eht ot emoc ot nem doog lla rof emit eht si woN

- **The Towers of Hanoi :**
- The Towers of Hanoi is a well-known children's game, played with three poles and a number of different-sized disks. Each disk has a hole in the center, allowing it to be stacked around any of the poles. Initially, the disks are stacked on the leftmost pole in the order of decreasing size, i.e., the largest on the bottom and the smallest on the top.

- The object of the game is to transfer the disks from the leftmost pole to the rightmost pole, without ever placing a larger disk on top of a smaller disk. Only one disk may be moved at a time, and each disk must always be placed around one of the poles.


1
2
3
4
Left
Center
Right

- For the case where n =3, the following output is obtained

```
Welcome to the TOWERS OF HANOI

How many disks? 3

Move disk 1 from L to R
Move disk 2 from L to C
Move disk 1 from R to C
Move disk 3 from L to R
Move disk 1 from C to L
Move disk 2 from C to R
Move disk 1 from L to R
```

Find output

Describe the output generated by each of the following programs.

(*a*)

```
#include <stdio.h>

int funct(int count);

main()
{
   int a, count;

   for (count = 1; count <= 5; ++count)   {
       a = funct1(count);
       printf("%d  ", a);
   }
}

int funct1(int x)
{
   int y;

   y = x * x;
   return(y);
}
```

Find output

- Output of previous program: 1 4 9 16 25

Find output

```
#include <stdio.h>

int funct1(int n);

main()
{
   int n = 10;

   printf("%d", funct1(n));
}

int funct1(int n)
{
   if (n > 0) return(n + funct1(n - 1));
}
```

Find output

- Output of previous program : 55

Global and local variables

- A local variable is a variable that is declared inside a function. A global variable is a variable that is declared outside **all** functions.

- A local variable can only be used in the function where it is declared. A global variable can be used in all functions.

Global and local variables

```
1.void function1(){
2.int x=10;//local variable
3.}
```

```
1.int value=20;//global variable
2.void function1(){
3.int x=10;//local variable
4.}
```

```
1.#include<stdio.h>
2.   // Global variables
3.   int a;
4.   int b;
5.   int Add()
6.   {
7.      return a + b;
8.   }
9.   int Mul()
10.  {
11.  int c=10; //Local Variable
12.  int d=20;  ////Local Variable
13.  return c*d;
14.  }
15.  void main()
16.  {
17.     int Ans1, Ans2, c=30;// Local variable
18.     a = 50;
19.     b = 70;
20.     Ans1 = Add();
21.     Ans2= Mul();
22.     printf("The addition result is: %d\n",Ans1);
23.     printf("The Multiplication result is: %d\n",Ans2);
24.     printf("%d\n", c);
25.  }
```

Output:
The addition result is: 120
The Multiplication result is: 200
30

Thank You