MA 511: Computer Programming

Lecture 8

http://www.iitg.ernet.in/psm/indexing_ma511/y08/index.html

Partha Sarathi Mandal

psm@iitg.ernet.ac.in

Dept. of Mathematics, IIT Guwahati

Semester 1, 2008-09

Mon 10:00-10:55 Tue 11:00-11:55 Fri 9:00-9:55 Class: 1G2

MA512 Lab : Wed 14:00-16:55

Function (**int**)

Example: Multiplication of 2 numbers

```
#include<stdio.h>

int multip(int a, int b);
main(){
        int x, y, k;
        printf("input X and Y: ");
        scanf("%d %d", &x,&y);
        k = multip(x,y);
        printf("multiplication = %d", k);
}
int multip(int m, int n){
        int p, q, r;
        p = m;
        q = n;
        r  =  p*q;
     return(r);
}
```

Alternative:

```
int multip(int m, int n){
                int r;
                r  =  m*n;
return(r);
}
```

Example: Multiplication of 2 numbers

```
#include<stdio.h>

int multip(int m, int n){
        int p, q, r;
        p = m;
        q = n;
        r  =  p*q;
     return(r);
}

main(){
        int x, y, k;
        printf("input X and Y: ");
        scanf("%d %d", &x,&y);
        k = multip(x,y);
        printf("multiplication = %d", k);
}
```

Function (**float**)

Example: Division of 2 numbers

```
#include<stdio.h>
float division(int a, int b);
main(){
        int x, y;
        float k;
        printf("input X and Y: ");
        scanf("%d %d", &x,&y);
        k = division(x,y);
        printf("division = %f", k);
}
float division(int m, int n){
        int p, q
        float r;
        p = m;
        q = n;
        r  =  (float)p/(float)q;
    return(r);
}
```

Example: Division of 2 numbers

```
#include<stdio.h>
float division(int m, int n){
        int p, q;
        float r;
        p = m;
        q = n;
        r  =  (float)p/(float)q;
    return(r);
}

main(){
        int x, y;
        float k;
        printf("input X and Y: ");
        scanf("%d %d", &x,&y);
        k = division(x,y);
        printf("division = %f", k);
}
```

Function (**void**)

Example: Multiplication of 2 numbers

```
#include<stdio.h>

void multip(int a, int b);

main(){
        int x, y, k;
        printf("input X and Y: ");
        scanf("%d %d", &x,&y);
        multip(x,y);
}
void multip(int m, int n){
        int p, q, r;
        p = m;
        q = n;
        r  =  p*q;
        printf("multiplication = %d", r);
}
```

Example: Multiplication of 2 numbers

```
#include<stdio.h>

void multip(int m, int n){
        int p, q, r;
        p = m;
        q = n;
        r  =  p*q;
      printf("multiplication = %d", r);
}

main(){
        int x, y, k;
        printf("input X and Y: ");
        scanf("%d %d", &x,&y);
        multip(x,y);
}
```

Definition of **Function**

- Its a self-contained *program segment* that carries out some specific, well define task.
- Every C program consists at least one function called **main**.
- Execution of a C program always begin by carrying out the instructions in **main**. Other functions are the subordinate to **main**.
- A function processes information that is passed (*arguments* , also called *parameters*) to it from the *calling portion* of the program, and *return* a single value.
- Some functions accept arguments but do not return anything called *void.*
- Some functions may return *multiple values*, with the help of pointer and arguments.

Function Prototypes

date-type **name**(*type 1 arg 1, type 2 arg 2, … type n arg n*);

Example:

```
int multip(int a, int b);
void multion(int a, int b);
float division(int a, int b);

#include<stdio.h>

date-type name(type 1 arg 1, type 2 arg 2, … type n arg n);

main(){

        name(arg 1, arg 2, … arg n);
}

date-type  name(type 1 arg 1, type 2 arg 2, … type n arg n){
}
```

Function Prototypes

```
#include<stdio.h>

date-type  name(type 1 arg 1, type 2 arg 2, … type n arg n){
      .
      .
}


main(){
                  .
      name(arg 1, arg 2, … arg n);
}
```

Code for n!

```
#include<stdio.h>
long int factorial(int n){
    int i;
    long int product = 1;
    if (n>1){
        for(i=2; i<=n; i=i+1)
            product *=i;
    }
    return(product);
}
main(){
    int n;
    print(“Type n = ”);
    scanf(“%d”, &n);
    printf(“n! = %ld ”, factorial(n));
}
```

Recursion

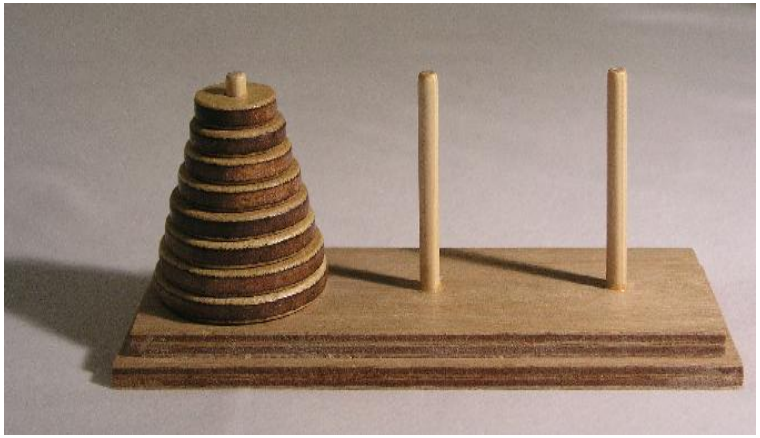
- Is a process by which a function calls itself repeatedly until some specific condition has been satisfied.

Example:

```
long int factorial(int n){
    If(n<=1)
        return(1);
    else
        return(n*factorial(n-1));
}
main(){
    int n;
    print(“Type n = ”);
    scanf(“%d”, &n);
    printf(“n! = %ld ”, factorial(n));
}
```

The Towers of Hanoi

- It consists of three rods, and a number of disks of different sizes which can slide onto any rod.
- The puzzle starts with the disks neatly stacked in order of size on one rod, the smallest at the top, thus making a conical shape.
- The objective of the puzzle is to move the entire stack to another rod, obeying the following rules:
  - Only one disk may be moved at a time.
  - Each move consists of taking the upper disk from one of the pegs and sliding it onto another rod, on top of the other disks that may already be present on that rod.
  - No disk may be placed on top of a smaller disk.



The Towers of Hanoi

- Can you write a c program for the **Towers of Hanoi** using *recursion* where number of disks is input ?