PROGRAMMING

FOR PROBLEM SOLVING

{ C LANGUAGE }

Module :- 6

start

Topic :-

- Introduction and writing functions
- Scope of variables function (including using built in libraries)
- Parameter passing in functions
- Call by value
- Passing arrays to functions
- Idea of call by reference

Function

- It is a collection of statements that performs an specific task.
- It executes when it is called by its name.
- A large program is divided into a number of small building block for simplicity and this building block is called function.
- We can call a function again and again.
- The most impotent features of function is code reusability.
- The C library provides many pre-defined functions.

Syntax of function

Function declaration

```
Return_type  function_name ( parameter type )
```

Function definition

```
Return_type  function_name ( parameter list )
                    {

                    }
```

Function calling

```
Function_name ( passing_parameter )
```

Description

return type	1. It is a keyword which indicates that which type of value is being returning by the function. 2. If we do not to return any value then we use void keyword in place of return_type.
function name	1. It is the actual name of the function. 2. It is also used at the time of calling the function.
parameter list	1. It is the place where we can pass a number of parameter/variable. 2. These variables may be used in the program. 3. The value of parameter can be initialized or we can pass it from the calling of function. 4. It is optional part.
body	1. It is the place where the actual code is written to perform the specific task

Key point about the function

Function Declaration: At this stage the function is declared

For example :
void add();

This is a function declaration in which void(no return type) indicates there is no value returning by this function and add is the name of function.
Note-It is optional.

Function Definition: This is the place where actual code is written to perform the task.

For example :

```
void add()
{
      int x,y=20,z=30;
      x=y+z;
      printf("Add=%d",x);
}
```

This is a function definition and here we can see that code is written to perform the addition task.

Function Calling: At this stage the fucntion is called .

For example :
add();

To call a function just write function name and put semi-colon(;) after it.

Example

```c
#include<stdio.h>

void add();//function declaration

void add()//function definition
{
    int x,y=20,z=30;
    x=y+z;
    printf("Add=%d",x);
}

int main()
{
    add();//function calling
}

/*
###Output###
Add=50
*/
```

Type of function

Predefined Function

The function which is predefined in the library is called predefined function.

for example:-
printf() , scanf() , clrscr() , getch() etc.

Userdefined Function

The function which is made by theuser is called userdefined function.

for example:-
add(),sub(),multi(),div()[Note: These are userdefined name.it may different.] etc.

Category of User defined Function

There are four category of user defined function

1. Function with no return type and no parameter.
2. Function with no return type and with parameter.
3. Function with return type and no parameter.
4. Function with return type and with parameter.

Function with no return type and no parameter

The function in which there is no parameter and there is no value returning by that function is called Function with no return type and no parameter

```c
//Function with no return type and no parameter
#include<stdio.h>
void add();//function declaration
void add()//function definition
{
    int x,y=20,z=30;
    x=y+z;
    printf("Add=%d",x);
}
int main()
{
    add();//function calling
}
/*
###Output###
Add=50
*/
```

Function with no return type and with parameter

The function in which there is some parameter and there is no value returning by that function is called Function with no return type and with parameter.

```c
//Function with no return type and with parameter
#include<stdio.h>
void add(int y,int z);//function declaration
void add(int y,int z)//function definition
{
    int x;
    x=y+z;
    printf("Add=%d",x);
}
int main()
{
    //here 20 will assign to y and 30 to z
    add(20,30);//function calling
}
/*
###Output###
Add=50
*/
```

Function with return type and no parameter

The function in which there is no parameter and there is some value returning by that function is called Function with return type and no parameter.

```c
//Function with return type and no parameter
#include<stdio.h>
int add();//function declaration
int add()//function definition
{
    int x,y=20,z=30;
    x=y+z;
    //return value of x
    return x;
}
int main()
{
    //assigning return value to variable rs
    int rs=add();//function calling
    printf("Add=%d",rs);
}
/*
###Output###
Add=50
*/
```

Function with return type and with parameter

The function in which there is some parameter and there is some value returning by that function is called Function with return type and with parameter.

```c
//Function with return type and with parameter
#include<stdio.h>
int add(int y,int z);//function declaration
int add(int y,int z)//function definition
{
    int x;
    x=y+z;
    //return value of x
    return x;
}
int main()
{
    //assigning return value to variable rs
    int rs=add(20,30);//function calling
    printf("Add=%d",rs);
}
/*
###Output###
Add=50
*/
```

Calling of Function

There are two way of calling a function

1. Call By Value.
2. Call By Reference

Call by Value

In this type of calling a function direct value is passed at the time of calling.
In call by value the changes made in formal parameters don't reflect in actual parameters.

```c
// //call by value

#include<stdio.h>
 void add(int m)//function definition
{
    //adding 10 to m
    m=m+10;
}
int main()
{
    int x=10;
    printf("Before Calling x=%d\n",x);
    add(x);
    printf("After Calling x=%d",x);
}
/*
###Output###
Before Calling x=10
After Calling x=10
*/
```

```c
//call by value
#include<stdio.h>
void swap(int , int);//function declaration
int main()
{
    int a,b;
    printf("enter a : ");
    scanf("%d",&a);
    printf("enter b : ");
    scanf("%d",&b);
    printf("\nin main before swap call a = %d , b = %d",a,b);

    swap(a,b);//function call , call by value

    printf("\nin main after swap call a = %d , b = %d",a,b);

    return 0;
}
void swap(int x, int y)//function definition
{
    int temp;
    temp=x;
    x=y;
    y=temp;
    printf("\nswap fun a = %d , b = %d",x,y);
}
/*
enter a : 5
enter b : 7

in main before swap call a = 5 , b = 7
swap fun a = 7 , b = 5
in main after swap call a = 5 , b = 7
*/
```

Call by Reference

1.In this type of calling a function ,the reference of the value is passed at the time of calling.

2.In call by value the changes made in formal parameters reflect in actual parameters.

3.Reference is also called address.

4.When the address of data is passed at the time of calling so it is neccessary to use pointer in the place of parameter.

```c
// //call by reference

#include<stdio.h>
 void add(int *m)//function definition
{
    //adding 10 to m
    *m=*m+10;
}
int main()
{
    int x=10;
    printf("Before Calling x=%d\n",x);
    add(&x);
    printf("After Calling x=%d",x);
}
/*
###Output###
Before Calling x=10
After Calling x=20
*/
```

KKS Computer Science

```c
//call by reference
#include<stdio.h>
void swap(int* , int*); //function declaration
int main()
{
    int a,b;
    printf("enter a : ");
    scanf("%d",&a);
    printf("enter b : ");
    scanf("%d",&b);
    printf("\nin main before swap call a = %d , b = %d",a,b);

    swap(&a,&b);//function call , call by reference

    printf("\nin main after swap call a = %d , b = %d",a,b);

    return 0;
}
void swap(int *ptr1, int *ptr2)//function definition
{
    int temp;
    temp=*ptr1;
    *ptr1=*ptr2;
    *ptr2=temp;
    printf("\nswap fun a = %d , b = %d",*ptr1,*ptr2);
}
/*enter a : 5
enter b : 7

in main before swap call a = 5 , b = 7
swap fun a = 7 , b = 5
in main after swap call a = 7 , b = 5
*/
```

Function with default value

In this type of function ,the function contains a number of parameter with some initial value

[for example : void sum(int x=10,int y=20)].

At the time of calling if there is no value is passed [for example: sum();] then the default value will be x=10 and y=20

but if value is passed [for example: sum(5,6);] then the value will be x=5 and y=6.

For better understanding see the example below-

Example

1. Addition program using function
2. Subtraction program using function
3. Multiplication program using function
4. Division program using function
5. Rectangle area using function
6. Circle area using function

Passing Array to Function

In this type of function ,there is an array in the place of parameter
[for example:void sum(int ar[5])] and its value is passed at the time of calling.

```c
//passing array in function
#include<stdio.h>
void sum(int ar[5])//function definition
{
    int i, s=0;
    for( i=0;i<5;i++)
    //finding the sum
    s=s+ar[i];
    printf("Total sum of element=%d",s);
}
int main()
{
    int x[5]={10,20,50,40,60};
    sum(x);//function calling with array
}
/*
### Output ###
Total sum of element=180
*/
```

In the above example we can see that there is an array ar[5] in place of parameter and there is an other array x[5]={10,20,50,40,60} and it is passed at the time of calling therefor the value of array x will be copied into array ar.

Thank You !!

Dhanybad !!

Shukriya !!