2016

Unit: 5 Function

Paper : no 102 Programming in c language

Prepared by : Vinod.M.Makwana

- Modular Programming
- Function
- Elements of Function
- Types of Function
- Call by value and call by reference
- Recursion

Prepared by : Vinod.M.Makwana
Smt.K.B.Parekh College of Computer Sciecen,Mahuva
11/22/2016

1.What is modular programming?

Modular programming is the process of subdividing a computer program into separate sub-programs.

- Less code has to be written.
- A single procedure can be developed for reuse, eliminating the need to retype the code many times.
- Modular programming allows many programmers to collaborate on the same application.
- Code is short, simple and easy to understand.
- Errors can easily be identified, as they are localized to a subroutine or function.
- The same code can be used in many applications. The scoping of variables can easily be controlled.

2.Elements of Functions:

Function:

Function is s subprogram of main program

A function is a block of code that performs a specific task.

A function is a group of statements that together perform a task. Every C program has at least one function, which is **main ()**, and all the most trivial programs can define additional functions.

There are following elements of function.

1. Function declaration.
2. Function call.
3. Function definition.

[1].Function declaration:

1. Function prototype is known as function declaration.
2. Like a variable, before we use it we must define it, so before using function it must be declare.
3. It consist four parts:
   a. Function returns type.
   b. Function name
   c. Parameter list.
   d. Termination semicolon.

Assistant Professor : Vinod.M.Makwana

Syntax:

Function_type function_name(parameter list);

Example:

```
void sum();
void  sum(int);
void sum(int,int,float);
void sum(int,float,char);
void sum(int a,int b,float c);
int sum(int,float b,char c);
float sum(int a[],float b[],char *);
char str(char *);
int *sum();
int * sum(int *a,int *,float b);
```

Note:

1. The parameter list must be separated by comma.
2. The name of the parameter names do not need to be same in the prototype declaration and function definition.
3. The types must be same in function declaration and function definition, in number and order.
4. Use of the name of parameter in function declaration is optional.
5. Return type is optional.
6. The return type must be void if no value is return.
7. When the declaration type does not match with the types of function definition, compiler will produce an error.
8. Function declaration is optional if function definition is defined before the calling function.

[2].Function call

An instruction which executes a block of code is known as function.

A function can be called by simply using the name of function followed by list of actual parameter(or arguments), if any, enclosed in parenthesis.

Syntax:

```
Function_name(actual_parameter);
```

Assistant Professor : Vinod.M.Makwana

Example:

```
Sum()
Sum(5);
Sum(a);
Sum(a,b);
Sum(a+5,8,10);
Sum(5,9.8,’c’);
X=sum(7,a,9.0);
```

Note:

1. Using function call control is transferred to the function definition.
2. In function call actual arguments are data send to the function definition.
3. If the actual parameter is more than the formal parameters, the extra parameters will be discarded.
4. On the other hand, if the actual are less than the formal, the extra unmatched formal arguments will be initialized to some garbage value.
5. Any mismatch in data type may also result in some garbage values.

Assistant Professor : Vinod.M.Makwana

[3]. Function definition:

The function definition is an independent program module that is specially written to implement the requirement of the function.

In order to use this function we need to invoke it at required place in the program, this is called function call.

The function is calls the function is referred to as calling function.

The calling function should be declaring any function that is to be used later in the program, this is known as function declaration or function prototype.

Function definition is also known as function implementation shall include the following elements.

1. Function_type.
2. Function_name.
3. List of parameters.
4. Local variables declaration.
5. Function statement.
6. Return statement.

All the six elements are grouped into two parts, namely.

a. Function header (first three elements).
b. Function body (second three elements).

Syntax:

```
return_type function_name(parameter list)
{
        Local variables declaration;
        Executable statement1;
        Executable statement2;
....................
.....................

        return statements;
}
```

a. **Function header:**

The function header consists of three parts such as:

1. Function type
2. Function name
3. Parameter list

Function type also known as return type, it specified at the end of the function a function which values is return to the calling function. Return value may be int, float, char, double,long int etc. If function does not return any value then it can be specified using void keywork.Void is a fundamental data type of c programming.

Function name is any valid c identifier and therefore it must follow the rules which are following by to defining a variable. The name the variable should be appropriate to the task performed by the function.

The parameter list declares the variables that will receive the data sent by the calling function, they serve as input data to the function to perform a specified task. The parameter list contains declaration of variable separated by commas.

Function header does not end with semicolon.

Assistant Professor : Vinod.M.Makwana

Example:

```
void sum(int a)
{
}

int sum()
{
}

float sum(int x,int y)
{
}

char str(char p[])
{
}
```

b. **Function body :**
   The function body contains the local variable declaration and statements necessary for performing the required task. The function body enclosed the three parts such as:

   1. Local variables declaration.
   2. Function statements
   3. Return statement

The scope the local variable is limited from opening curly brackets to closing curly brackets.

Function statements referred as executable statements which are used to perform a specified task.

If function does not return any value, means a function is a type of void then return stamen is not necessary but, if function is return type then return statement must be a last statement of a particular function. Using return statement control is transfer to the calling function.

Example:

```
int sum(int a,int b)
{
        Int c;
        c=a+b;
        return c;
}
float avg(int a,float b,int c)
{
        Float x;
        x=(a+b+c);
        return (x/3);
}
```

Assistant Professor : Vinod.M.Makwana

```
Void display()
{
        Printf("hi.....");
}
```

3.Types of function

Q:what is user defined function? List out categories of function. Explain types of function with example.

A block of code which performs specified task.

A function which defines by the user as per requirement is known as user define function.

Categories of function

1. Inbuilt function.
2. User define function.

A function which is defined by the c liabrary,means there is no need to writes it code to perform a task is known as inbuilt function.

Ex:

a. Strcmp()
b. Strstr()
c. gets ()
d. puts()
e. printf()
f. scanf()

a function which is create by the user as per the user requirement is known as user defined function(UDF).

Exa:

a. main()
b. sum()
c. max()
d. min()
e. sort()

Assistant Professor : Vinod.M.Makwana

Types of function

1. Function with no arguments and no return values.
2. Function with arguments but no return values.
3. Function with no arguments but return single value.
4. Function with arguments and return single value.
5. Function that returns multiple values.

1. Function with no arguments and no return values.

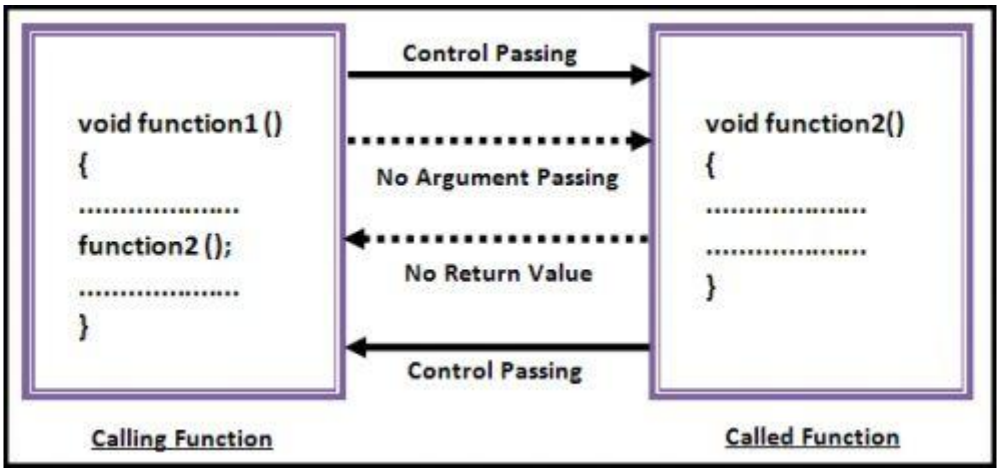
- When a function has no arguments, it does not receive any data from the calling function. Similarly, when it does not return a value, the calling function does not receive any data from the called function.
- In effect, there is no data transfer between the calling function and the called function.
- Following diagram shows that only control is transfer but data cannot be transfer.

Syntax:

```
Void sum();

Void mul();
```

Example:



2. Function with arguments and no return value.

- In this kind of function data is send from calling function to the called function.
- Calling function send data to the called function, called function receive data as input.
- It is also known as one way communication.
- Control is transfer, data is send but calling function does not receive any value from the called function.

Syntax:

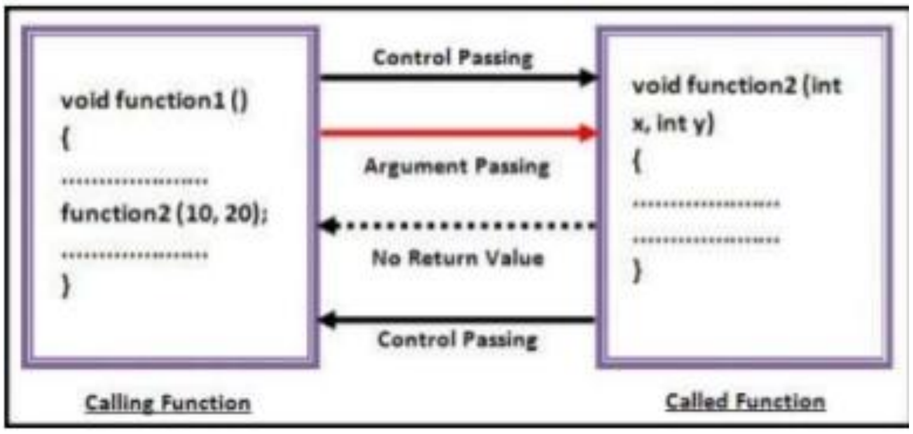
Assistant Professor : Vinod.M.Makwana

Void sum(int,int);

Void sum(int a);

Void sum(int a,float,char c);

Following diagram show the process of function.



- As per the above example function1() is a calling function where function2() as a called function.
- Function1() sends value such as 10,20 as inputs to the function2().
- Control is transfer from function1() to function2(),data are send but data cannot be returns from called function.

3. **<u>Function with no arguments but return single value.</u>**

- In this kind of function data cannot be send from calling function to called function but data may receive from called function to calling function.
- This function is also known as “one way communication” function.

Syntax:

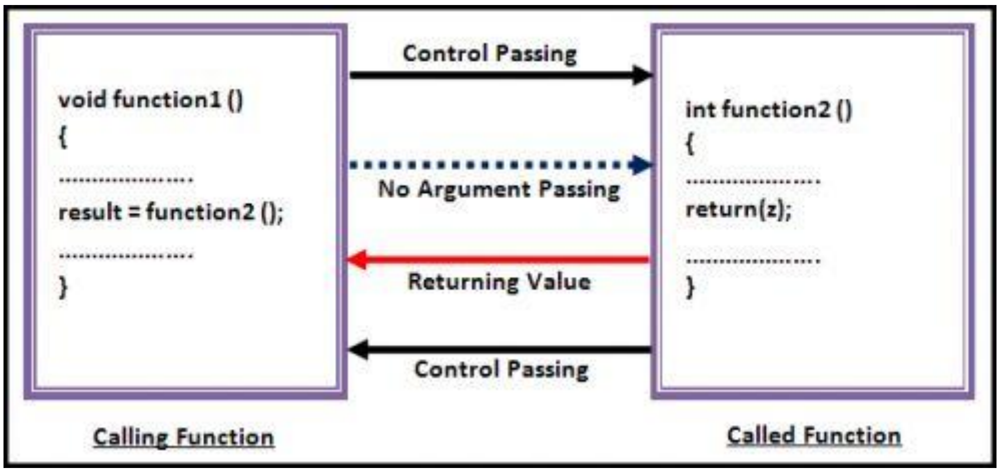
variable_name=function_name();

Example:

X=sum();

Assistant Professor : Vinod.M.Makwana

Following diagram shows process of function.



- As per the above function function1() is known as calling function and function2() is know as called function.
- From calling function control is transfer to the function call, but it cannot send data to the called function
- Called function must have a last statement as return statement , from called function control is transfer to the calling function and return some value, calling function must have a variable which hold the data which sending from the called function.
- As per above diagram z is return to the function1() and data stored into the result variable.

4. **Function with arguments and return single value.**

- This function sending data as well as receive data from calling function to called function respectively.
- This function is also known as "Two way communication" function.

Syntax:

Variable_name=function_name(parameters);

Example:

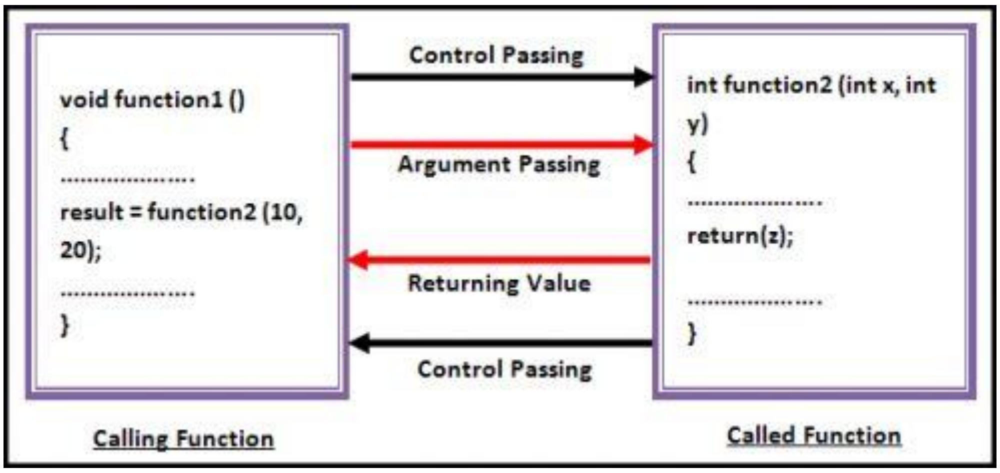
X=sum(f);

P=sum(x,m);

Z=sum(5,6,a);

Assistant Professor : Vinod.M.Makwana

Following diagram shows the process of function.



- As per above example function1() is a calling function while function2() is a called function.
- Control is transfer from function1() to function2(), function1() sends data to function2() via arguments, function2() takes as a inputs.
- Function2() again data is send to the function1() which receives as z by variable result.
- So in this kind of function data are send as well as receive.

5. **Function returns multiple values.**

- Generally function can return only single value at a time but using this function we can return multiple values at a time without return statement.
- In this function prototype must be pointer or an array which actual argument must takes as reference of the variable.

Example:

```
void data(int *a,int *b)

{
        (*a)++;
        (*b)++
}
```

Assistant Professor : Vinod.M.Makwana

```
Void main()
{
        Int a=10,b=20;
        Printf("Before calling function :a:%d\nb:%d",a,b);
        data(a,b);//call by value
        printf("\nAfter calling function :a:%d\nb:%d\n",a,b);
        data(&a,&b);//call by reference
        printf("after pass by reference :a:%d\nb:%d\n",a,b);
}
```

Example:2

```
Void change(int *a)
{
        Int i;
        For(i=0;i<5;i++)
                a[i]=a[i]+10;
}

Void main()
{
        int a[]={1,2,3,4,5},i;
        Clrscr();
        Change(a);//call by reference

        For(i=0;i<5;i++)
        Printf("\n%d",a[i]);
        getch();
}
```

As per above example if actual arguments are passing as reference(&) so, it affects to the calling function also while if arguments are passing as without reference then it can not affect to the calling function but affect to only called function.

Assistant Professor : Vinod.M.Makwana

Difference between call by value and call by reference

	Call by value	**Call by reference**
1	A copy of value is passed to the function	An address of value is passed to the function
2	Changes made inside the function is not reflected on other functions	Changes made inside the function is reflected outside the function also
3	Actual and formal arguments will be created in different memory location	Actual and formal arguments will be created in same memory location

4	Example
	
	Void sum(int p,int q)
	{
	    p++;
	    q++;
	}
	
	Void main()
	{
	    int a=10,b=20;
	    sum(a,b);
	    printf(“%d\t%d”,a,b);//a=10,b=20
	}
		Example
		
		Void Sum(int *p,int *q)
		{
		    (*p)++;
		    (*q)++
		}
		
		Void main()
		{
		    int a=10,b=20;
		    Sum(&a,&b);
		    Printf(“%d\t%d”,a,b);//a=11 b=21
		}

What is recursion? Explain with proper example.

- Function called itself is known as recursion.
- In programming language a function is called inside that function is called recursion.
- But while using recursion programmer must need to know exit point, so programmer must deep knowledge of the programming otherwise it goes into the infinite.
- Recursive functions are very useful to solve many mathematical problems, such as calculating the factorial of a number, generating Fibonacci series, etc.

Assistant Professor : Vinod.M.Makwana

```
//factorial

#include<stdio.h>
#include<conio.h>

long int fact(int x)
{
        static long int sum=1;

        if(x>=1)
        {
                sum=sum*x;
                x--;
                fact(x);
        }

        return sum;
}



void main()
{
        int n;
        long int p;
        clrscr();

        scanf("%d",&n);
        p=fact(n);

        printf("\n\nFactorial :%ld",p);

        getch();
}
```

Assistant Professor : Vinod.M.Makwana

Theory assignment

1. What is UDF? List out categories of function.
2. What is function? Explain elements of function.
3. Write a note on: passing arguments as array.
4. Write a note on: passing arguments as string.
5. Give difference between call by value and call by reference.
6. What is modular programming? Explain recursion in detail

Practical assignment: 6

Write a c program using all types of function

1. To perform addition, substraction, division, multiplication and remainder.
2. Factorial of given number.
3. Sum of digits.
4. Reverse number.

Write a c program using any type of function

1. Find binary of given decimal number.
2. Find decimal of given binary number.
3. Find octal from given decimal number.
4. Find decimal from given octal number.
5. Find hexadecimal from decimal.
6. Find decimal from hexadecimal number.
7. Find binary from given octal number.
8. Find octal from given binary number.
9. Find binary from given hexadecimal number.
10. Find hexadecimal from given binary number.
11. Find octal to hexadecimal.
12. Find hexadecimal to octal number.

Function and array

1. Find max from the given array.
2. Sort in ascending order.
3. Find out element is existing or not.
4. Enter 3X3 two matrix and perform matrix multiplication operation.
5. Check string is palindrome or not.
6. Copy one string to another (without using strcpy).
7. Connate one string to another.
8. Input paragraph and find number of words, lines, characters, alphabets, digits and special symbols.
9. Find vowels and consonants' from the given string.
10. Input character and to check whether it is digit, alphabets, space or special symbols.
11. Input any 10 names and sort in alphabetical order.

Assistant Professor : Vinod.M.Makwana

Assistant Professor : Vinod.M.Makwana