Local and Global Variable:

The variables may be classified as local or global variables.

Local Variable

The variables defined can be accessed only within the block in which they are declared. These variables are called "Local" variables

Example

```
funct (int ,int j)
{
        intk,m;
        _________;
        _________;
}
```

The integer variables **k** and **m** are defined within a function block of the "**funct()**". All the variables to be used within a function block must be either defined at the beginning of the block or before using in the statement. Local variables one referred only the particular part of a block of a function.

Global Variable

Global variables defined outside the main function block. Global variables are not contained to a single function. Global variables that are recognized in two or more functions. Their scope extends from the point of definition through the remainder of the program.

b. Calling functions by value or by reference

The arguments are sent to the functions and their values are copied in the corresponding function. This is a sort of information inter change between the calling function and called function. This is known as Parameter passing. It is a mechanism through which arguments are passed to the called function for the required processing. There are two methods of parameter passing.

1. Call by Value
2. Call by reference.

1. Call by value: When the values of arguments are passed from calling function to a called function, these values are copied in to the called function. If

any changes are made to these values in the called function, there are NOCHANGE the original values within the calling function.

Example

```
#include <stdio.h>
main();
{
        int n1,n2,x;
        int cal_by_val();
        N1 = 6;
        N2 = 9;
        printf( n1 = %d and  n2= %d\n", n1,n2);
        X = cal_by_Val(n1,n2);
        Printf( n1 = %d and  n2= %d\n", n1,n2);
        Printf(" x= %d\n", x);
                / * end of main*/
        /*function to illustrate call by value*/
        Cal_by_val(p1,p2)
        int p1,p2;
        {
                int  sum;
                Sum = (p1 + p2);
                P1 + = 2;
                P2* = p1;
                printf( p1 = %d and  p2= %d\n", p1,p2);
                return( sum);
        }
}
```

When the program is executed the output will be displayed

N1 = 6 and n2 = 9

P1 = 8 and p2 = 72

N1 = 6 and n2 = 9

X = 15

There is NO CHANGE in the values of n1 and n2 before and after the function is executed.

2. Cal by Reference: In this method, the actual values are not passed, instead their addresses are passed. There is no copying of values since their memory locations are referenced. If any modification is made to the values in the called function, then the original values get changed with in the calling function. Passing of addresses requires the knowledge of pointers.

Example

This program accepts a one-dimensional array of integers and sorts them in ascending order. [This program involves passing the array to the function].

```
# include <stdio.h>
main();
{
int num[20], I,max;
void sort_nums();
printf( " enter the size of the array"\n");
scanf("%d", &max);
for( i=0; i<max;I++)
sort_nums(num,max)        /* Function reference*/
printf("sorted numbers are as follows\n");
for( i=0; i<max;I++)
printf("%3d\n",num[i]);
/* end of the main*/
/* function to sort list of numbers*/
```

```
Void sort_nums(a,n)
Int a[],n;
{
        Int I,j,dummy;
        For(i=0;i<n;i++)
        {
                For(j=0; j<n; j++)
                {
                        If( a[i] >a[j])
                        {
                                Dummy = a[i];
                                a[i] = a[j];
                                a[j] = dummy;
                        }
                }
        }
}
```

3.5 Recursion

One of the special features of C language is its support to recursion. Very few computer languages will support this feature.

Recursion can be defines as the process of a function by which it can call itself. The function which calls itself again and again either directly or indirectly is known as recursive function.

The normal function is usually called by the main ( ) function, by mans of its name. But, the recursive function will be called by itself depending on the condition satisfaction.

For Example,

```
main ( )
{
```