**CALL BY VALUE & CALL BY REFERENCE**

The arguments passed to function can be of two types namely

1.Values passed

2. Address passed

The first type refers to call by value and the second type refers to call by reference.

Call by reference

* Passes the address of argument
* Constants cannot be passed to the function with exception of strings.
* Changes in function effect original argument.
* Only used with trusted functions
* Using pointers, a function can return more than one value indirectly.

Call by value

* Copy of argument is passed to function
* Constants can be passed to the function
* Changes to the copy in function do not effect original variable’s value.
* Use when function does not need to modify argument
* Function can return only one value
* Avoids accidental changes
* All calls in C are call-by-value.

For instance consider **program1**

void interchange(int x1,int y1)   
{   
 int z1;   
 z1=x1;   
 x1=y1;   
 y1=z1;   
 printf(“x1=%d y1=%d”,x1,y1);   
}

main()   
{   
 int x=50, y=70;   
 interchange(x,y);   
 printf(“x=%d y=%d”,x,y);   
}

Here the value to function interchange is passed by value.

Consider **program2**

void interchange(int \*x1, int \*y1)   
{   
 int z1;   
 z1=\*x1;   
 \*x1=\*y1;   
 \*y1=z1;   
 printf(“\*x=%d \*y=%d”,x1,y1);   
}

main()   
{   
 int x=50, y=70;   
 interchange(&x,&y);   
 printf(“x=%d y=%d”,x,y);   
}

Here the function is called by reference. In other words address is passed by using symbol & and the value is accessed by using symbol \*.

The main difference between them can be seen by analyzing the output of program1 and program2.

The output of program1 that is call by value is: But the output of program2 that is call by reference is:

x1=70 y1=50   
x=50 y=70

\*x=70 \*y=50   
x=70 y=50

This is because in case of call by value the value is passed to function named as interchange and there the value got interchanged and got printed as x1=70 y1=50

and again since no values are returned back and therefore original values of x and y as in main function namely x=50 y=70 got printed.