Difference between
Call By Value & Call By Reference:

Functions can be invoked in two ways: Call By Value and Call By Reference.

The parameters or arguments which are passed by the calling function are called ACTUAL PARAMETERS and the parameters
which are used in the function definition statement which contain data type(s) on its time of declaration are known as
FORMAL PARAMETERS.

CALL BY VALUE:

In this parameter passing method, the values of actual parameters are copied to the function's formal parameters and the
two types of parameters are stored at the different memory locations. So any changes made inside the function are not
reflected in actual parameters of the Caller.

CALL BY REFERENCE:

In this parameter passing method, location of the actual parameters are passed into the function's formal parameters.
Both the actual and formal parameters refer to the same memory locations, so any changes made inside the function are
actually reflected in actual parameters of the Caller.

CALL BY VALUE	CALL BY REFERENCE
While calling a function, we pass values of variables to it. Such functions are known as “Call By Values”.	While calling a function, instead of passing the values of variables, we pass address of variables(location of variables) to the function known as “Call By References.

In this method, the value of each variable in calling function is copied into corresponding dummy variables of the called function.	In this method, the address of actual variables in the calling function are copied into the dummy variables of the called function.
With this method, the changes made to the dummy variables in the called function have no effect on the values of actual variables in the calling function.	With this method, using addresses we would have an access to the actual variables and hence we would be able to manipulate them.

```c
// C program to illustrate
// call by value

#include <stdio.h>
#include <stdlib.h>

// Function Prototype
void swapx(int x, int y);

// Main function
int main(){
    int a = 10, b = 20;

    // Pass by Values
    swapx(a, b);
    printf("a=%d b=%d\n", a, b);
    return 0;
}

// Swap function that swaps
// two values
void swapx(int x, int y){
    int t;
    t = x;
    x = y;
    y = t;
    printf("x=%d y=%d\n", x, y);
}
```

```c
// C program to illustrate
// Call by Reference

#include <stdio.h>
#include <stdlib.h>

// Function Prototype
void swap(int*, int*);

// Main function
int main(){
    int a = 10, b = 20;

    // Pass reference
    swap(&a, &b);
    printf("a=%d b=%d\n", a, b);
    return 0;
}

// Function to swap two variables
// by references
void swap(int* x, int* y){
    int t;
    t = *x;
    *x = *y;
    *y = t;
    printf("x=%d y=%d\n", *x, *y);
}
```

By- Nishant Jaiswal(21165MCA004)
Class: MCA Ist Year