Call by reference

Before we discuss function call by reference, lets understand the terminologies that we will use while explaining this:
Actual parameters: The parameters that appear in function calls.
Formal parameters: The parameters that appear in function declarations.
For example: We have a function declaration like this:

```
int sum(int a, int b);
```

The a and b parameters are formal parameters.

We are calling the function like this:

```
int s = sum(10, 20); //Here 10 and 20 are actual parameters
or int s = sum(n1, n2); //Here n1 and n2 are actual parameters
```

When we call a function by passing the addresses of actual parameters then this way of calling the function is known as call by reference. In call by reference, the operation performed on formal parameters, affects the value of actual parameters because all the operations performed on the value stored in the address of actual parameters.

```
#include <stdio.h>
void increment(int    *var){
      /* Although we are performing the increment on variable
        * var, however the var is a pointer that holds the address
        * of variable num, which means the increment is actually done
        * on the address where value of num is stored.
        */
*var = *var+1;
}
int main(){
        int num=20;
        /* This way of calling the function is known as call by
          * reference. Instead of passing the variable num, we are
          * passing the address of variable num
          */
        increment(&num);
        printf("Value of num is: %d", num);
    return 0;}
```

Output:
Value of num is: 21

Example 2: Swapping numbers using Call by Reference

```
#include <>
void swapnum ( int *var1, int *var2 ){
    int tempnum ;
    tempnum = *var1 ;
    *var1 = *var2 ;
```

```
    *var2 = tempnum ;
}

int main( ){
    int num1 = 35, num2 = 45 ;
    printf("Before swapping:");
    printf("\nnum1 value is %d", num1); // 35
    printf("\nnum2 value is %d", num2);// 45

    /*calling swap function*/
    swapnum( &num1, &num2 );

    printf("\nAfter swapping:");
    printf("\nnum1 value is %d", num1); // 45
    printf("\nnum2 value is %d", num2);// 35
    return 0;}
```

Output:

```
Before swapping:
num1 value is 35
num2 value is 45After swapping:
num1 value is 45
num2 value is 35
```