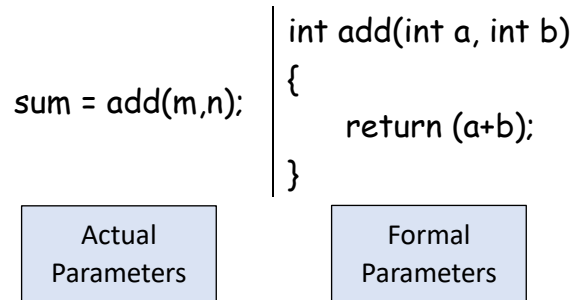


**Video No.:** 64

**Topic Name:** Call by Value & Call by Reference in C

**Actual Parameters:** The parameters passed to a function.

**Formal Parameters:** The parameters received by a function.



### Call By Value

Here the values of actual parameters will be copied to formal parameters and these two parameters store value in different location.

```
int main(){
    int x = 600, y = 500;
    fun(x, y);
    printf("The value of x in main is %d \n", x);
    printf("The value of y in main is %d\n", y);
}

int fun(int x, int y){
    X = 50;
    Y = 60;
    printf("x is %d \n", x);
    printf("y is %d\n", y);
}
```

>> Here in this line the function will jump to the fun() function

>> in fun() definition we declared the value of x and y again

>> but this value will only remain in the fun() function and won't change the original value of x and y


>> So, Basically, function can change the variable value but it will only remain in the local place

## Call By Reference

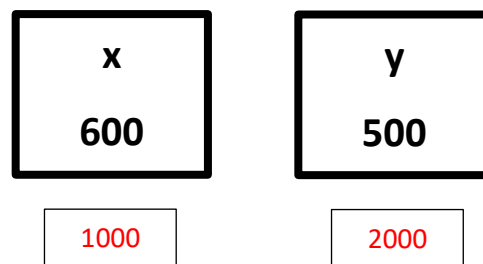
Here both actual and formal parameters refer to same memory location. Therefore, any changes made to the formal parameters will get reflected to actual parameters.

Here instead of passing values, we pass addresses

```
int main(){  
    int x = 600, y = 500;  
    fun(&x, &y);  
    printf("The value of x in main is %d \n", x);  
    printf("The value of y in main is %d\n", y);  
}  
  
int fun(int *prt1, int *prt2){  
    *prt1 = 50;  
    *prt2 = 60;  
}
```



>> Here in this line the function will jump to the fun() function but one thing is to notice that here we are passing the address of x and y through the function



>> Let the location of x is 1000 and y is 2000

>> To store the location in variable we need special type of variable called Pointer.

>> So, &x = 1000 and &y = 2000

>> \*prt1 gets 1000 and \*prt2 gets 2000

\*prt1 = 50; // this line says go to location 1000 and access its value change it 50

\*prt2 = 60; // this line says go to location 2000 and access its value change it 60

>> and so, the value of actual x, y is changed.

