

PRACTICAL -1

Introduction to pointers. Call by Value and Call by reference.

- **CALL BY VALUE:** In call by value method, the value of the actual parameters is copied into the formal parameters.
- In call by value method, we cannot modify the value of the actual parameter by the formal parameter.
- In call by value, different memory is allocated for actual and formal parameters since the value of the actual parameter is copied into the formal parameter.
- The actual parameter is the argument which is used in the function call whereas formal parameter is the argument which is used in the function definition.

CODE:

```
#include<stdio.h>
void swap (int, int);

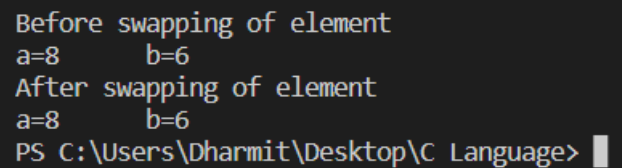
void main ()
{
    int a = 8, b = 6;
    printf("\n Before swapping of element");
    printf("\n a=%d \t b= %d ",a , b);

    swap (a, b);
    // a and b are the actual parameters in this call.
    printf ("\n After swapping of element");
    printf ("\n a=%d \t b= %d", a, b);
}
void swap (int x, int y)

// x and y are formal parameters.

{
    int t=x;
    x = y;
    y = t;
}
```

OUTPUT:



```
Before swapping of element
a=8      b=6
After swapping of element
a=6      b=8
PS C:\Users\Dharmit\Desktop\C Language>
```

CALL BY REFERENCE: In call by reference, the address of the variable is passed into the function call as the actual parameter.

- The value of the actual parameters can be modified by changing the formal parameters since the address of the actual parameters is passed.
- In call by reference, the memory allocation is similar for both formal parameters and actual parameters.
- All the operations in the function are performed on the value stored at the address of the actual parameters, and the modified value gets stored at the same address.

CODE:

```
#include<stdio.h>
void swap (int*, int*);
void main ()
{
    int a = 8, b = 6;
    printf("Before swapping of element");
    printf("\n a=%d \t b=%d", a, b);

    swap (&a, &b);
    // a and b are the actual parameters in this call.
    printf("\n After swapping of element");
    printf("\n a=%d \t b=%d", a ,b);
}

void swap (int*x, int*y)
// x and y are formal parameters.
{
    int t=*x;
    *x=*y;
    *y=t;
}
```

OUTPUT:

```
Before swapping of element
a=8    b=6
After swapping of element
a=6    b=8
PS C:\Users\Dharmit\Desktop\C Language>
```