CALL BY VALUE & CALL BY REFRENCE

Subject:- PPS

Semester:- 2nd

Branch:- MECH+ CIVIL

We can call the functions by two methods:-

1. **Call by Value.**
2. **Call by Reference.**

- Call by Value means that we deals with the direct values. And Call by Reference means that we deal with addresses where the values are stored.
- For dealing with addresses we must have the clear concept of pointers. Pointers are variables that can hold the address of any variable.
- So functions are very important part of C. Even in the practical field we can’t make any project without using function as it would become more complex and difficult without them.
- But the most important thing which is also an issue if common sense that the name of the function must be relevant to the working of the function so that just by seeing the name only we can understand what implementation that function is doing.

1. Call by value

Call by Value : If we call a function in C by passing values of variables as the parameters/arguments to the function then such type of function call is known as Call by value.

Call by Reference :

- We know that whatever variable we are using in our program, ultimately it is getting stored somewhere in memory. So instead of passing values of variables as parameters to the function, if we pass address of those variables and somehow able to access data contained inside those addresses then we will be able to call functions. Interestingly, concepts of Pointers provide us the advantage of manipulating addresses. So calling a function by passing addresses of variables as the parameters to the function is known as Call by Reference.

Concept of Actual and Formal arguments

- Arguments passed to the function during function call are known as actual arguments and the arguments we use during a function definition are known as formal arguments. For a function call to be valid the type, order and number of actual and formal arguments must always be same.
- During call by value method the ‘value’ of each of the actual arguments in the calling function is copied into corresponding formal arguments of the called function. In this method, the changes made to the formal arguments in the called function have no effect on the values of actual arguments in the calling function.

Program of call by value method

```
1. #include<stdio.h>
2. int sum(int x, int y);   /*function declaration*/
3. int main()
4.   {
5. int num1, num2, add;
6. printf(" Enter any two numbers \n");
7. scanf("%d, %d", &num1, &num2);
8. add = sum(num1, num2);  /*Calling the add function */
9. return 0;
10. }
11. int add( int x,  int y)    /* Function Definitions */
12. {
13. int sum;
14. sum=x+y;
15. return sum;
16. }
```

Program of call by reference method

```
1. #include<stdio.h>
2. void swap(int *, int *);  /*function declarations */
3. int main();
4. {
5. int a=5, b=6;
6. swap(&a, &b);          /*Calling functions by    reference
//passing the address */
7. printf("a=%d b=%d \n",a,b);
8. return 0;
9. }
10. void swap ( int *x, int *y)     /*Function Definitions */
11. {
12. int temp;
13. temp= *x;
14. *x=*y;
15. *y=temp;
16. }
```

THANK YOU