Function in C

```
#include <stdio.h>
   void callme()
{
printf("You call me form Main \n");
   }

int main()
   {
   callme();
   return 0;
   }
```

```
#include <stdio.h>
void max(int a, int b){
    int maximum = (a>=b) ? a : b ;
    printf("max is: %d", maximum);
    }
int main()
    {
    int x, y;
    scanf("%d", &x);
    scanf("%d", &y);
    max(x,y);
    return 0;
    }
```

How to call C functions in a program?

There are two ways that a C function can be called from a program. They are,

- Call by value
- Call by reference

1. Call by value:

- In call by value method, the value of the variable is passed to the function as parameter.
- The value of the actual parameter can not be modified by formal parameter.
- Different Memory is allocated for both actual and formal parameters. Because, value of actual parameter is copied to formal parameter.

Note:

- Actual parameter – This is the argument which is used in function call.
- Formal parameter – This is the argument which is used in function definition

```
#include<stdio.h>
// function prototype, also called function declaration
void swap(int a, int b);

int main()
{
   int m = 22, n = 44;
   // calling swap function by value
   printf(" values before swap  m = %d \nand n = %d", m, n);
   swap(m, n);
}

void swap(int a, int b)
{
   int tmp;
   tmp = a;
   a = b;
   b = tmp;
   printf(" \nvalues after swap m = %d\n and n = %d", a, b);
}
```

2. Call by reference:

- In call by reference method, the address of the variable is passed to the function as parameter.
- The value of the actual parameter can be modified by formal parameter.
- Same memory is used for both actual and formal parameters since only address is used by both parameters.

Example program for C function (using call by reference):

- In this program, the address of the variables “m” and “n” are passed to the function “swap”.
- These values are not copied to formal parameters “a” and “b” in swap function.
- Because, they are just holding the address of those variables.
- This address is used to access and change the values of the variables.

```
#include<stdio.h>
// function prototype, also called function declaration
void swap(int *a, int *b);

int main()
{
   int m = 22, n = 44;
   //  calling swap function by reference
   printf("values before swap m = %d \n and n = %d",m,n);
   swap(&m, &n);
 }

void swap(int *a, int *b)
{
   int tmp;
   tmp = *a;
   *a = *b;
   *b = tmp;
   printf("\n values after swap a = %d \nand b = %d", *a, *b);
}
```