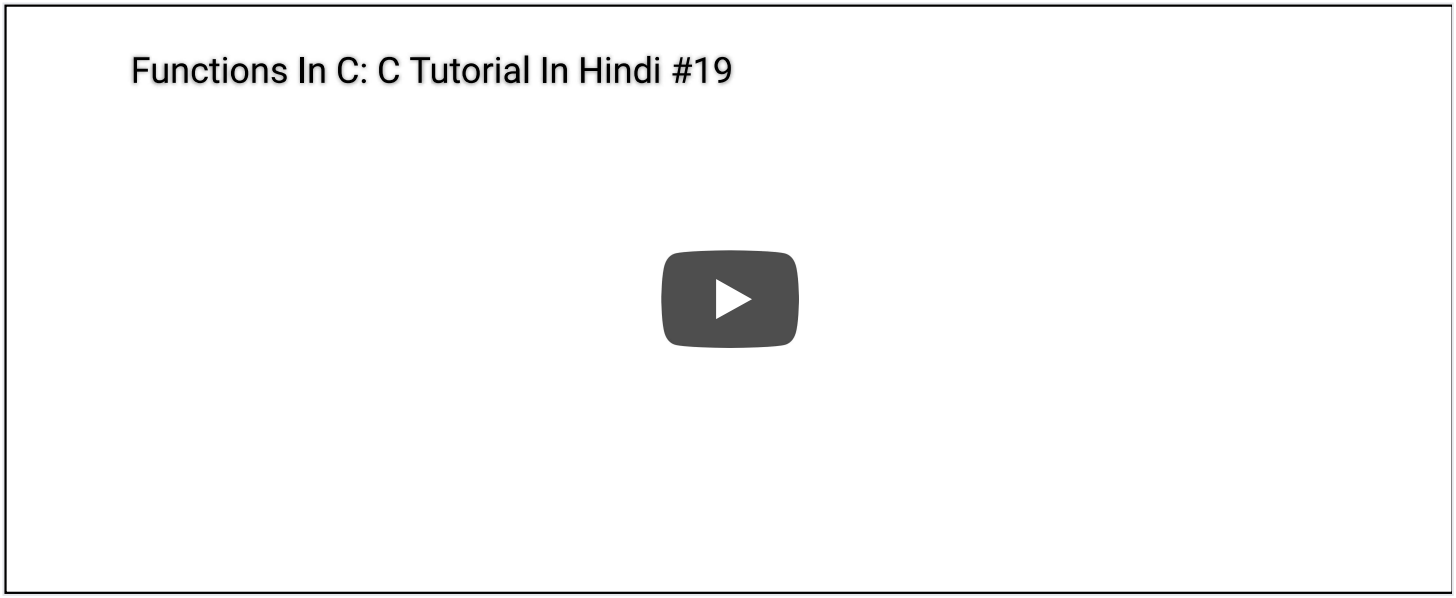
CodeWithHarry



Course: C Language Tutorials For Beginners

Overview Q&A Downloads Announcements

Functions In C: C Tutorial In Hindi #19

Functions :

- Functions are used to divide a large C program into smaller pieces.
- Function can be called multiple or several times to provide reusability and modularity to the C program.
- Functions are also called as procedure or subroutines.
- It is a piece of code to accomplish certain operation.

Now let's understand in simple terms what function is?
Function is nothing but a group of code put together and given a name and it can be called anytime without writing the whole code again and again in a program.
I know its syntax is bit difficult to understand but don't worry after reading this whole information about Functions you will know each and every term or thing related to

Functions.

Advantages of Functions :

- We can avoid rewriting same logic or code through functions.
- We can divide the work among programmers using functions.
- We can easily debug or can find bugs in any program using functions.

Function Aspects :

There are 3 aspects of function :-

1. Declaration
2. Definition
3. Call

- A function is declared to tell the compiler about its existence.
- A function is defined to get some task done. (It means when we define function we write whole code of that function. In this actual implementation of function is done.)
- A function is called in order to be used.

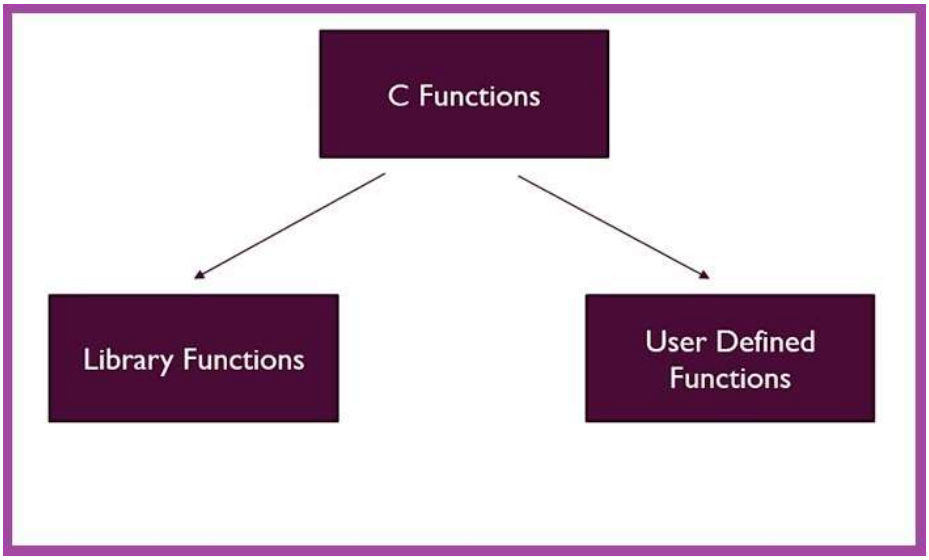
Types of Functions :

- Library Functions – These are pre-defined functions in C Language. These are the functions which are included in C header files.

E.g. printf(), scanf() etc.

- User defined Functions – Functions created by programmer to reduce complexity of a program i.e. these are the functions which are created by user or programmer.

E.g. Any function created by programmer.



NOTE -: Every C Program must contain one function i.e. main() function as execution of every C program starts from main() function.

Ways to define a Function :

There are 4 ways in which we can define any C Function and these are :

- Without arguments and without return value.
- Without arguments and with return value.
- With arguments and without return value.
- With arguments and with return value.

1.Without arguments and without return value : In this function we don't have to give any value to the function (argument/parameters) and even don't have to take any value from it in return.

```
//No Argument and No Return Value
void Star_pattern()
{
    int a;
```

```
    printf("Enter how many stars(*) you want : \n"); /* In this both declaratic
    scanf("%d", &a );
    for (int i = 0; i<a; i++)
    {
        printf("*");
    }
}
```

As you can see in above example the function with name 'Star_pattern' is not taking any argument (Since it's parentheses are empty) and even it is not returning any value as we haven't even used return keyword for returning something and in return_type we have written 'void' which means nothing.
So, with this example you can understand about functions which do not take any argument or value and even don't return anything.

2.Without arguments and with return value : In these type of functions we don't have to give any value or argument to function but in return we get something from it i.e. some value.

```
// Without arguments and with return value :
int Sum(); /* Declaration of Function */

 /*Other Code*/
int Sum()
{

    int x,y,z;
    printf("Enter no. 1 : \t");
    scanf("%d",&x);
    printf("\nEnter no. 2 : \t");
    scanf("%d",&y );
    z=x+y;
    return z;
}
```

In above example you can see that our function with name 'Sum' is not taking any value or argument (Since parentheses are empty) but it is returning an integer value. So, that's how we can use functions which don't take any argument but returns something.

3.With arguments and without return value : In these type of function we give arguments or parameters to function but we don't get any value from it in return.

```
// With arguments and without return value :
void Product(int x ,int y ); /* Declaration of Function */

            // Other Code
Product(5,4) /* Calling Product Function in main() */


void Product(int a,int b)
{
    int Multiplication;
    Multiplication = a*b ; /* Definition of Function */
    printf("The Product is %d\n\n",Multiplication);
}
```

In this example as you can see we have declared function globally i.e. this function can be used anywhere in our program not just only in main() function. After that we have defined it to tell what this function will do and in this function as it will take some value or argument, it will process that data and will give some output but it will not return anything.
When we pass arguments in function at the time of calling it at that time only the value of variables get copied to variable of that function means only values are passed. It means any modification which is done on passed values do not affect the actual values of the variables in main() function. And this call is known as **Call by Value**.

4.With arguments and with return value : In these functions we give arguments to function and in return we also get some value from it.

```
// With arguments and with return value :
float Percentage(int x,int y); // Declaration of Function
/*Code*/ int x;
x = Percentage(80,95); // Calling Function
 /*Code*/
float Percentage(int x,int y)
{
    float perct;
    perct = ((x+y)/200.0)*100.0; // Definition of Function
    return perct;
}
```

So, this is the widely used method to define function in this way we give arguments to function and also get some value in return from it.
So, that's all about Functions in C Language.

Code as described/written in the video

```
#include <stdio.h>
int sum(int a, int b);
void printstar(int n)
{
    for(int i = 0; i < n; i++)
    {
        printf("%c", '*');
    }
}


int takenumber()
{
    int i;
    printf("Enter a number");
```

```
    scanf("%d", &i);
    return i;
}
int main()
{
    int a, b, c;
    a = 9;
    b = 87;
    // c = sum(a, b);
    // printstar(7);
    c = takenumber();
    // printf("The sum is %d \n", c);
    printf("The number entered is %d \n", c);
    return 0;
}


int sum(int a, int b)
{
    return a + b;
}
```

Previous Next

CodeWithHarry