Static Variables

As we know that the scope in any programming, whether it is C or Python, is a region of the program where a defined variable can exist and beyond which it cannot be accessed. For a quick recap, let us revise the concept of a local and global variable.

Local Variables

Local variables are declared inside a function or a block of code; they cannot be accessed outside the function. The local variables can be used only by statements that are inside that function or block of code. The system does not initialize local variables, we must initialize it ourself. The scope of these variables will be within the function only.

Global Variables

Global variables are defined outside a function, usually in the main. Global variables hold their values, and we can access them inside any of the functions defined for the program. Global variables are initialized by the system automatically when we define them.

If the local and global variables have the same name, the local variable will take preference.

Formal Parameters

Formal parameters have precedence over global variables, and they are treated as local variables within a function. The term **formal parameters are used** to refer to the **parameters** in the definition of the method. In contrast, actual **parameters** are the variable or expression that appears in the function or method call in the calling environment.

Now let us move to our main topic of this tutorial, i.e., static variables. Static is a keyword in the C language. We can use it with variables and functions.

What is a static variable?

A static variable is known to retain the value even after they exit the scope. Static variables retain their value and are not initialized again in the new scope. The static variable until the end of the program is kept in the memory, whereas a normal variable is destroyed when a function is over. They can be defined inside or outside the function. Static variables are local to the block. The default value of static variables is zero. The keyword **static** is used to declare a static variable.

Syntax:

```
   static Datatype Variable_name = Variable_value;
```

datatype − The data type of variable like char, int, float, etc.

variable_name − This is the name of the variable.

Variable_value − Value to initialize the variable. By default, it is zero.

Example:

```
#include <stdio.h>

int myfunc()
{
 static int i=0;
  i++;
  return i;
}

int main()
{
printf("Value:%d",myfunc());
printf("\nValue:%d",myfunc());

    return 0;
}
```

Output:-

```
Value:1
Value:2
```

Differences between static local and the static global variable

Static global variable

If we declare the variable with a static keyword outside the function, then it is known as a static global variable. This variable will be accessible throughout the program.

Static local variable

When a variable with a static keyword is declared inside a function is known as a static local variable. The scope of the static local variable will be the same as the local variables, but its memory will be available throughout the execution of the program.

Properties of a static variable

The following are some properties of a static variable:

- Static variable memory is allocated within a static variable.
- A static variable will retain the value even after they exit the scope.
- Static variable memory is available throughout the program.
- If we do not assign any value to the static variable, then the default value will be 0.