**STORAGE CLASSES**

|  |  |  |  |
| --- | --- | --- | --- |
| **VARIABLE TYPE** | **STORAGE** | **LIFE** | **SCOPE** |
| local variable | Inside frame of the function in Stack Area | Life of the frame. Till end of the function or Block | Inside the block or function in which it is declared |
| function Param | Inside frame of the function in Stack Area | Life of the frame. Till end of the function | Inside the function |
| static local variable | G.D.S. | From the function call till end of the program | Inside the function or the block |
| static variable | G.D.S. | Life of the program | File scope |
| Global variable | G.D.S. | Life of the program | Program or Project scope |

A variable declared inside a block of { } is called as local variable and its scope (visibility/accessibility) is only inside that block.

The best example for a block is function { }

Like wise we have, if {} else{} loop{} swtitch{} etc.

The variable declared inside a block is visible/accessible only inside that block.

Further, its life is also inside the block. Thus, when the control comes outseide the block, the variable declared inside the block will deallocated (deleted from memory).

The local variables will have either default value or junk value(compiler dependent). In some compilers like Visual Studio, the variables will be empty util they are initialzed, i.e. the 1st assignment to the variable is considered to be the initialization. This is decided by the compiler. Thus local variable values will be known at compile time.

Now, the variables which are Function arguments(parameters) too are similar to local variables. However the only difference is that, function parameters will get their value only during runtime. And thus their value is not assertained in compile time.

STATIC LOCAL VARIABLE:

It is also a local variable. Thus it is accessible only inside the block. However, it's life is retained until end of the program.

Note that, it gets life only once.

Thus, when the control goes to the declarative statement of a static local variable for the 2nd time onwards, there wont be any memory allocation, because the variable already exists in memory.

GLOBAL VARIABLE:

A global is one which is declared outside the function (in fact all functions).

Note that, all functions are global in nature, meaning no function can be defined inside another.

Thus, a global variable is accessible or visible to all the functions inside the file.

However, if the functions in other FILES, if they wish to access global variable defined/declared in other file, then we have to do the below 2 things:

1. include the file in which the global variable is defined

2. say to the compiler that you wish to use an EXTERNAL variable (variable declared in other file) using the keyword extern

A global variable has project scope (in all files and all functions can access it) and its life is till end of the program.

STATIC GLOBAL VARIABLE:

If we wish to have a global variable in a file, but we dont want any other file to access it (functions from other files) then make it static.

Thus, a static global variable has file scope. And its life is life of the program.