Storage Classes

"A storage class defines scope, default initial value, and a lifetime of a variable."

Here, scope refers to the variable's availability at places. The initial default value refers to the value present in the variables as default before being initialized, and the lifetime refers to the variable's duration of life.

Now, as we are familiar with the basic concept of storage classes, let us move to its types. There are four types of storage classes depending upon the type of variables they store. The following are the type of storage classes. You can guess their stored variables by their names:

- Automatic Variables
- External Variables
- Static Variables
- Register Variables

Note: Each storage class differs, with respect to scope, default initial value, and a lifetime of a variable.

Auto Storage Class:

Variables being formed in a function and whose storage class has not been defined initially fall in this category automatically. Its scope is minimum as it can only be accessed inside the function it is initialized in. No other function can access it. Until the variable has been assigned some value, it stores garbage value as default. Their lifetime depends upon the function block's length as the lifetime is until the function block's end.

```
int a;
auto int a;
//Both are the same.
```

External Storage Class:

These sorts of variables are defined outside the function, hence can be used inside any function, meaning that they can be used globally. Their initial value is set to 0. As they can be used throughout the program, so their lifetime equals the lifetime of the program. Too many global variables in a program can cause security issues and also are not usually recommended.

Extern Keyword:

Using the extern keyword, we inform our compiler that the variable is already declared at some other place. By doing so, we can use the same variable with the same space, without allocating its new memory and accessing the same variable in some other file. Its syntax is simple as we have to use the extern keyword, and it will automatically access it from the other file.

```
extern int a;
```

Static Storage Class:

Static variables are a little bit technical as their lifetime is throughout the program, but their scope is limited to the function they are initialized in. It comes in handy when we are changing their value in the program as the program will store the new value, overwriting the previous one. Their initial default value is 0, and their syntax is very easy as we just have to use the Static keyword during initialization.

```
static int a;
```

Register Storage Class:

It is very similar to the Auto storage class as its scope is limited to the function it is defined in, the initial default value is 0, and lifetime is till the end of the function block. Now the major difference between it and the others is that it requests the CPU's register memory instead of the local memory for fast access. It is usually used for the programs that need to be accessed faster than the other or used frequently.

```
register int a;
```