Storage classes are used to describe the features of a variable. These features basically include the scope, visibility and life-time which help us to trace the existence of a particular variable during the runtime of a program.

In C language, each variable has a storage class which decides the following
things:

- scope i.e where the value of the variable would be available inside a program.
- default initial value i.e if we do not explicitly initialize that variable, what will be its default initial value.
- lifetime of that variable i.e for how long will that variable exist.

FEATURE	STORAGE CLASS			
	Auto	Extern	Register	Static
Existence	Exists when the function or block in which it is declared is entered. Ceases to exist when the control returns from the function or the block in which it was declared	Exists throughout the execution of the program	Exists when the function or block in which it is declared is entered. Ceases to exist when the control returns from the function or the block in which it was declared	***Local:*** Retains value between function calls or block entries ***Global:*** Preserves value in program files
Default value	Garbage	Zero	Garbage	Zero

FEATURE	STORAGE CLASS			
	Auto	Extern	Register	Static
Accessibility	Accessible within the function or block in which it is declared	Accessible within all program files that are a part of the program	Accessible within the function or block in which it is declared	***Local:*** Accessible within the function or block in which it is declared ***Global:*** Accessible within the program in which it is declared
Storage	Main Memory	Main Memory	CPU Register	Main Memory

The following storage classes are most often used in C programming,

1. Automatic variables
2. External variables
3. Static variables
4. Register variables

Automatic variables: auto

Program:

```
#include <stdio.h>
int main( )
{
        auto int j = 1;
        {
                auto int j= 2;
                {
                        auto int j = 3;
                        printf ( "%d  ", j);
                }
                printf ( "%d  ",j);
                }
        printf( "%d", j);
}
```

Output :

3 2 1

Static variables: static

```
#include <stdio.h>
void display();
int main()
```

```
{
        display();
        display();
}
void display()
{
        static int c = 1;
        c += 5;
        printf("%d ",c);
}
```

Output:

```
6  11
```

Register variables : register

```
#include <stdio.h>
int main( )
{
        register int a=25;
        printf("%d",a);
        return 0;
}
```

Output:

```
25
```

External Variables : extern

File1.c

```
#include<stdio.h>
#include "file2.c"
extern int a;
int main()
{
        printf("%d",a);
        return 0;
}
```

File2.c

```
int a =50;
```

Output:

```
50
```