

	$+10$	$\cancel{+10}$	$-1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 0 \times 2^0$
(4)	0000	0000	-8
	0001	1111 (-1)	
	0010	1110 (-2)	
	0011	1101 (-3)	
	0100	1100 (-4)	
	0101	1011 (-5)	$\rightarrow 10+7$
	0110	1010 (-6)	$\boxed{-8 \rightarrow +7}$
	0111	1001 (-7)	$\boxed{-8 \rightarrow +7}$

(10101)

$$\begin{array}{r} +8 \\ + -7 \\ \hline 1 \end{array} \quad \begin{array}{r} +8 \\ - +7 \\ \hline 1 \end{array}$$

2^b Complement is a weight system —
But sign bit always treated as (Eve) Number

$$\begin{array}{r} 0111 \\ -1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 \\ \hline 7 \end{array} \quad \begin{array}{r} 100 \\ -1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 \\ -8 + 0 + 0 + 1 \Rightarrow -7 \end{array}$$

$$\begin{array}{r} \cancel{+10} \rightarrow \text{Difference} \\ +8 \\ \hline 00000010 \\ \hline 00000001 \end{array}$$

$$\begin{array}{r} 1000 \\ -1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 0 \times 2^0 \\ \hline -8 \end{array}$$

$$\begin{array}{r} 20 \\ 10 \\ 15 \\ 2 \\ 1 \\ 0 \end{array}$$

int $x = 20$;

int $c = x \& 9$;

$$\begin{array}{r} A \ B \mid Y \\ 0 \ 0 \quad 0 \\ 0 \ 1 \quad 0 \\ 1 \ 0 \quad 0 \\ 1 \ 1 \quad 1 \end{array}$$

$$\begin{array}{r} +20 \\ 00010100 \\ -11 \\ 11110101 \\ \hline 00010100 \\ \downarrow \\ C \leftarrow 20 \end{array}$$

\rightarrow

Truth Table

A	B	Y
0	0	0
0	1	1
1	0	1

Truth Statement:
If Input are same
output is zero
otherwise one

i i | 0

int z = $\text{~}x \ll 1$

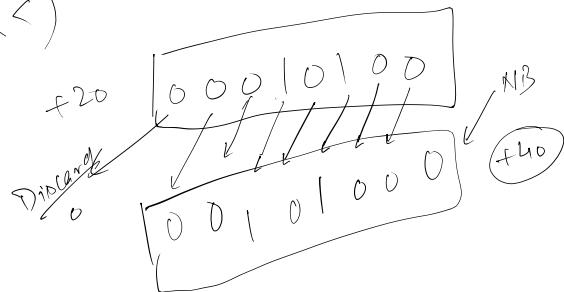
$$\begin{array}{r} 00010100 \\ 11110101 \\ \hline 1110001 \end{array}$$

$$\begin{aligned} & -1 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^0 \\ & -128 + 64 + 32 + 1 \\ & -128 + 97 \Rightarrow -31 \end{aligned}$$

int z = $\text{~}x \ll 1$
↓

Left shift :- (\ll)

$z = \text{~}x \ll 1$



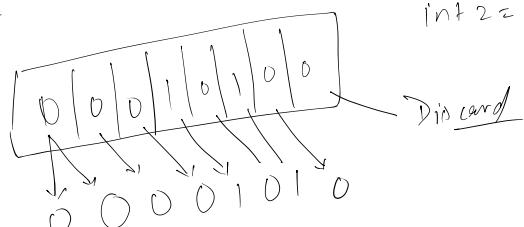
if we leftshift
a number
it will be multiplied
by $\times 2$

99
990

If left shift
Any Number
then it will be
multiplied by
its Base x.

Right shift :-

z
→ sign maintained
by right shift.



int z = $\text{~}x \gg 1$;

In C programming, storage classes define the scope, visibility, lifetime, and default initial value of variables and functions. They determine where a variable or function can be accessed from and how long it exists in memory.

There are four storage classes in C:

1. auto

- Default storage class for local variables.
- Automatically assigned to variables declared inside a function if no other storage class is specified.
- Scope: Local to the block in which it's defined.
- Lifetime: Exists during the execution of the block.
- Default initial value: Garbage value (uninitialized).

void main() {

 auto int x = 5; // 'auto' is optional here

 printf("%d", x);

2. register

- Suggests that the variable be stored in a CPU register instead of RAM for faster access.
- Scope: Local to the block.
- Lifetime: Duration of the block.
- Default initial value: Garbage value.
- You cannot get the address of a register variable using the & operator.

void main() {

 register int speed = 20;

 printf("%d", speed);



3. Static

- Keeps the value of a variable between function calls.
- Can be used with local variables and global variables/functions.
- For local variables:
 - Scope: Local to the function/block.
 - Lifetime: Entire program execution.
 - Default Value: Zero.
- For global variables/functions:
 - Visibility: Limited to the file (file scope).

```
void example() {
```

```
    static int count = 0;
```

```
    count++;
```

```
    printf("%d", count);
```

```
}
```

```
// Output across multiple calls: 1 2 3 ...
```

```
4. extern
```

- Used to declare a global variable or function that is defined in another file or later in the code.

```
• Scope: Global.
```

```
• Lifetime: Entire program.
```

```
• It does not allocate memory; just declares the variable.
```

```
// file1.c
```

```
int count = 10;
```

```
// file2.c
```

```
extern int count;
```

```
void printCount() {
```

```
    printf("%d", count);
```

```
}
```

Summary Table

Storage Class	Scope	Lifetime	Default Value	Usage
auto	Local	Block	Garbage	Default for local variables
register	Local	Block	Garbage	Fast-access (no address)
static	Local/File	Entire program	Zero	Persistent local vars
extern	Global	Entire program	Zero (if defined)	Access global vars/functions

If you want a deep dive into any one of them or example use cases, let me know!

