Bitwise Operator

All Bitwise Operator works on bits level

i) Bitwise AND (&)

5 & 6 ⇒

5 = 101

6 = 110

5 & 6 ⇒ 4

```
  101
& 110
-----
  100  ⇒ 4
```

ii) Bitwise OR (|)

5 | 6 ⇒ 7

5 ⇒ 101

6 ⇒ 110

---

111 ⇒ 7

iii) Bitwise XOR (^)

$5 \wedge 6 \Rightarrow 3$

$5 \Rightarrow 101$

$6 \Rightarrow 110$

$011 \Rightarrow 3$

iv) Bitwise NOT (~)

unary operator

$\sim 5 = \sim(00000000 \ldots 00000101)$

$= 11111111 \ldots 11111010$

$\Rightarrow -6$

$$\sim x = -(x+1)$$

It flips the bits

v) Left Shift operator (<<)

5 = 00000000 ---- 00000101

5 << 1 = 00000000 ---- 00001010

= 10

$$a << n = a \times 2^n$$

No. of times left shift operation to be Performed.

- This operator shift all bits left side & add 0 to LSB & discard the MSB.

- Left shifting a negative no. is undefined.

vi) Right shift operator (>>)

10 = 00000000 00001010

10 >> 1 = 00000 00000101

= 5

- $a >> n = a / 2^n$

- signed integer handles -ve no with right shift. but in unsigned integer -ve no. will give a very large +ve no.

-10 → 

-10 >>1 → 

-10 >>2 → 

-10 >>3 → 

-10 >>4 → 

Max we can get -1 while doing right shift on a -ve number.

- We cannot shift by a -ve no. as garbage value is throwns.

Eg $(10 << -1)$ ⟶ Some garbage value.