BITWISE OPERATORS

① Bitwise AND : (&)

x	y	AND
0	0	0
0	1	0
1	0	0
1	1	1

Eg: 5 & 7 : 101 & 111 = 101 = (5)

② Bitwise OR : (|)

x	y	OR
0	0	0
0	1	1
1	0	1
1	1	1

Eg 2 | 5 : 010 | 101 = 111 : 7

③ Bitwise NOT : (~)

x	NOT
0	1
1	0

Eg: (Consider 4-byte representation)

a = 2 = 000...0010 (30 bits)

~a = ~2 = 111...1101

Signed Bit

Two's complement = 000...0010 + 1 = 000...0011 = (-2)

④ Bitwise XOR : (^)

x	y	XOR
0	0	0
0	1	1
1	0	1
1	1	0

Eg. 5 ^ 7 = 101 ^ 111 = 010 = 2

⑤ Left Shift : (<<)

Eg: $5 << 1$ → Left shift 5, 1 time

$101 << 1 \rightarrow 00\cdots 0101 << 1$

$= 00\cdots 1010 = 10 = 5 \times 2^1$

Eg $3 << 2 \rightarrow 00\cdots 011 << 2$

$00\cdots 1100 = 12 = 3 \times 2^2$

* In most cases we multiply with power of 2 But in some case, this isn't true.

Eg: $0100\cdots 00 << 1 = 1000\cdots 00$

Positive ⟶ Negative ??

So << is Ok for smaller numbers

⑥ Right Shift : (>>)

Eg $15 >> 1 = 000\cdots 1111 >> 1$

$= 000\cdots 0001 = 1$

Pending in << and >> for positive numbers

for negative number, padding is Compiler dependent.

⑦ Increment / Decrement

→ We can write $i = i+1$ as (i++) or (++i) → Pre-Increment

→ Post-Increment

→ We can write $i = i-1$ as (i--) or (--i) → Pre-decrement

→ Post decrement

Post increment : The value gets used first and then increment

Eg. int i=3, a=2;
int sum = a + (i++);
sum = 2 + 3;
sum = 5;
Now i is 4

Pr increment : The value get incremented first and then gets used.

Eg. int i=3; a=2;
int sum = a + (++i) ; i has become 4 first
sum = 2+4;
sum = 6;

Post-Decrement : The value gets used first and then decrements

Eg : int i=3, a=2;
int sum = a + (i--);
sum = 2+3;
sum = 5;
New i=2

Pr-Decrement : The value gets decrement first and then gets used.

Eg: int i=3, a=2
int sum = a + (--i); i has become 2 first
sum = 2+2
sum = 4

Eg
```
int i=7;
cout <<(++i)<<endl;
// 8
cout <<(i++)<<endl;
// 8 , i=9
cout <<(i--)<<endl;
// 9 , i=8
cout <<(--i)<<endl;
// 7 , i=7
```

output
8
8
9
7

Eg
```
int main()
{
    int a,b=1;
    a=10
    if(++a)
        cout<<b;
    else
        cout<<++b;
}
```

output1
1

```
int main()
{
    int a=1
    int b=2;
    if(a-->0 && ++b>2)
    {
        cout<<"Stage1";
    }
    else
    {
        cout<<"stage2";
    }
    cout<<a<<" "<<b<<endl;
}
```

output
Stage1 0 3

```
int main()
{
  int a=1
  int b=2;
  if(a-->0 || ++b>2)
  {
    cout<<"Stage1";
  } else
  {
    cout<<"Stage2";
  }
  cout<<a<<" "<<b<<endl;
}
```

Output

Stage 1 0 2

Hint ⇒ Only one of the conditions must be true for || so it won't check ++b>2.

```
int main()
{
    int num=3;
    cout<<(25*(++num));
}
```

100

```
int main()
{
    int a=1
    int b = a++;
    int c = ++a;
    cout<<b;
    cout<<c;
}
```

Output

1 3