N = 45 : 101101

	5	4	3	2	1	0
45 :	1	0	1	1	0	1
(1<<2) :	0	0	0	1	0	0
OR	1	0	1	1	0	1

→ 45 → N

	5	4	3	2	1	0	
	1	0	1	1	0	1	→ 45
OR	0	1	0	0	0	0	→ (1<<4)
	1	1	1	1	0	1	

→ $45 + 2^4$

$45 + (1<<4)$

Q → Given N, set it's ith bit !

$\boxed{N \mid (1<<i) \rightarrow \text{It's ith bit would be set (1)}}$

$N \mid (1<<i)$ → N < ith bit in N is ALREADY SET >

$N \mid (1<<i)$ → $N + (1<<i)$ < ith bit in N is NOT SET >

	5	4	3	2	1	0	
45 :	1	0	1	1	0	1	
1<<2 :	0	0	0	1	0	0	
XOR	1	0	1	0	0	1	→ N-(1<<2)

XOR

	5	4	3	2	1	0	
45 :	1	0	1	1	0	1	
(1<<4):	0	1	0	0	0	0	
	1	1	1	1	0	1	→ N+(1<<4)

$$\boxed{N \wedge (1<<i) \rightarrow \text{ith bit of N gets toggled} \quad 1 \rightarrow 0, \; 0 \rightarrow 1}$$

	5	4	3	2	1	0
45:	1	0	1	1	0	1
1<<2 :	0	0	0	1	0	0
AND	0	0	0	1	0	0

↓ (1<<2)

	5	4	3	2	1	0
45	1	0	1	1	0	1
1<<4:	0	1	0	0	0	0
	0	0	0	0	0	0

↓ 0

N & (1<<i) →
- (1<<i) : ith bit is SET
- 0 : ith bit is UNSET

Check if ith bit is SET → AND

Set the ith bit → OR

toggle the ith bit → XOR

Q Given N, UNSET the ith bit!

N = 25

4 3 2 1 0

1 1 0 0 1

i = 2

1 1 0 0 1

i = 3

4 3 2 1 0

1 1 0 0 1

1 0 0 0 1

④

```
if( (N & (1<<i)) > 0 ){   → ith bit is SET
    N = (N ^ (1<<i));     → Toggle
}
```

★

```
    1 0 1 1 0 1
OR  0 1 0 0 0 0
    -----------
    1 1 1 1 0 1   → SET the ith bit for sure
^   0 1 0 0 0 0
    -----------
    1 0 1 1 0 1   → UNSET it
```

$$\boxed{N = (N \mid (1<<i)) \wedge (1<<i)}$$

★

```
                i
N:        1 0 1 1 0 1
~(1<<i) & 1 1 0 1 1 1
          -----------
          1 0 0 1 0 1
          -----------
```

$1<<i$: 0 0 1 0 0 0

$\sim(1<<i)$: 1 1 0 1 1 1

$$\boxed{N = (N \,\&\, (\sim(1<<i)))}$$

Q Given N & i.
check if the ith bit is set or NOT!

1) if((N | (1<<i)) == N)
   ↳ ith bit SET

else ↳ ith bit UNSET

2) if((N ^ (1<<i)) == (N - (1<<i))){
   ↳ ith bit is SET

else ↳ ith bit is UNSET

3) if((N & (1<<i)) > 0)
   ↳ ith bit is SET

else
   → ith bit is UNSET.

$$\begin{array}{lcccccc} & 5 & 4 & 3 & 2 & 1 & 0 \\ N: & 1 & 0 & 1 & 1 & 0 & 1 \\ \& & 0 & 0 & 0 & 0 & 0 & 1 \end{array}$$

$(N >> 3)$: 000101

$\&$: 000001

$\boxed{000001}$

4) if(((N >> i) & 1) == 0)

→ ith bit is UNSET!

else

→ ith bit is SET!

Q Given a no. N. Count the total no. of set bits!

N = 6 → 110 → 2

N = 25 → 11001 → 3

N = 45 → 101101 → 4

N = 32 → 100000 → 1

N = 0 → 0 → 0

③ INT → 4 bytes → 32 bits

31 30 29 28 27 26 25 24 — 1 0

N =   0 0 1 0 1 0 0 0 — 0 — 1 0

```
cnt = 0;
f( i= 0; i < 32; i++) {
    if((N & (1<<i)) > 0) {
        cnt++;
    }
}
ret cnt;
```

32

$\boxed{TC = O(1)}$

N

10 : 0 0 0 0 1 0 1 0

N = (N >> 1)

5 : 0 0 0 0 0 1 0 1

N = (N >> 1)

2 : 0 0 0 0 0 0 1 0

N = (N >> 1)

1 : 0 0 0 0 0 0 0 1

STOP → 0 : 0 0 0 0 0 0 0 0

```
cnt = 0
while ( N > 0) {
    if ( (N & 1) > 0) {
        cnt++;
    }
    N = (N >> 1);  or  N = N/2;
}
ret cnt;
```

$\boxed{TC = O(\log_2 N)}$

N = 6

$$\begin{array}{r} 110 \\ \& \ 001 \\ \hline \boxed{000} \end{array}$$

N = N >> 1

N = 3

$$\begin{array}{r} 011 \\ \& \ 001 \\ \hline \boxed{001} \end{array}$$

cnt++ cnt = 1

N = N >> 1

N = 1

$$\begin{array}{r} 001 \\ \& \ 001 \\ \hline \boxed{001} \end{array}$$

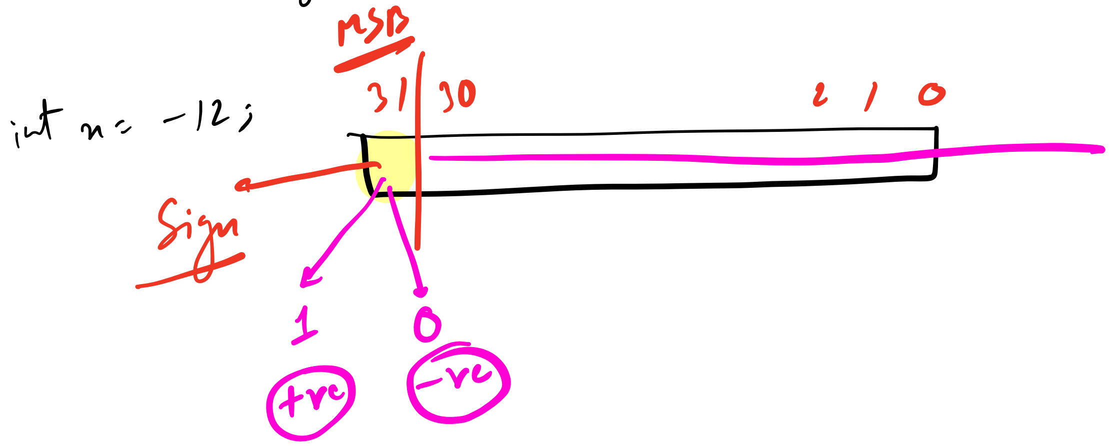
cnt++ cnt = 2

N = N >> 1

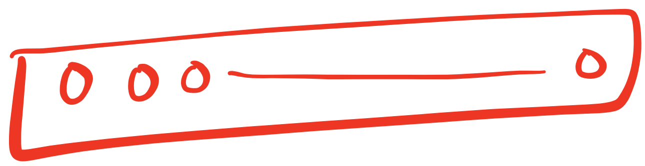
N = 0 000 STOP!

4 Negative No's

int n = -12;



1: +0



-0

2 diff rep for same no!

2. Additional circuit would be required for dealing with -ve no's.

How are -ve no's stored?

2's complement form

MSB bit : SIGN bit

0 : +ve no

1 : -ve no

8 bit No.

Byte x = -12

	7	6	5	4	3	2	1	0	
12 :	0	0	0	0	1	1	0	0	
Toggle all bits:	1	1	1	1	0	0	1	1	1's complement
+	0	0	0	0	0	0	0	1	
-12 :	1	1	1	1	0	1	0	0	2's complement

$-2^7 + 2^6 + 2^5 + 2^4 + 2^2$

$-128 + 64 + 32 + 16 + 4$

$-128 + 116$

$= -12$

8bit SIGNED

7	6	5	4	3	2	1	0
1	0	1	0	1	0	0	0

byte x = -88;

$-2^7 + 2^5 + 2^3$

$-128 + 32 + 8$

$-128 + 40 = -88$

7	6	5	4	3	2	1	0
0	1	0	1	0	0	0	0

$\underline{-2^7 \times 0} + 2^6 + 2^4 = 64 + 16$

$= 80$

N = -0

0 : 0000 0000

1's comp: 1111 1111

\+ 0000 0001

$\boxed{00000000}$

+0

8bit SIGNED

N = 45

7	6	5	4	3	2	1	0
0	0	1	0	1	1	0	1

1's C → 1 1 0 1 0 0 1 0

\+ 0 0 0 0 0 0 0 1

−45   1 1 0 1 0 0 1 1   2's C

45   + 0 0 1 0 1 1 0 1

0 ← 0 0 0 0 0 0 0 0

Subtraction of binary No's →

45 − 12 → 33

45 + (−12)

45   0 0 1 0 1 1 0 1

\+

−12   1 1 1 1 0 1 0 0

0 0 1 0 0 0 0 1

$2^5 + 2^0$

= 33

→ Subtraction does not need ANY ADDITIONAL circuit. it Uses ADDER CIRCUIT!

SIGNED	vs	UNSIGNED
both (+ve) & (-ve)		Only (+ve) no's.
int	→	uint

4bit

3	2	1	0
1	1	0	1

$$2^3 + 2^2 + 2^0 = 13$$

Q. What is the MAX no. an INT can store? SIGNED

31								—	.	1	0
0	1	1	1	1	1	1	-	———	1	1	1

$$2^{30} + 2^{29} + 2^{28} \text{ ——— } + 2^0$$

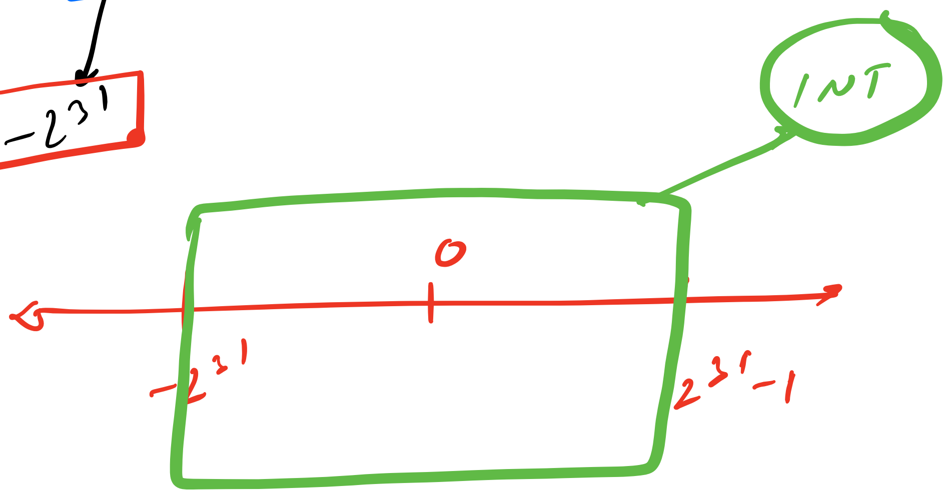
$$\boxed{2^{31}-1}$$

Q What is the SMALLEST NO — in INT?

31 —— 2 1 0

1 0 0 0 0 —— 0

$\boxed{-2^{31}}$



Q → find the range for UNSIGNED INT!

MAX

31 —— 2 1 0

1 1 1 1 1 1 1 1 1 1 1 1 1

$2^{31} + 2^{30} + 2^{29} + \text{——} + 2^{0}$

$\boxed{2^{32}-1}$

MIN

0 0 0 —— 0

$\boxed{0}$

SIGNED

byte	int	long
8 bit	32 bit	64 bit
2^7-1	$2^{31}-1$	$2^{63}-1$
		9×10^{18}
	$\sim -2 \times 10^9$ … 2×10^9	$\sim -9 \times 10^{18}$ … $+9 \times 10^{18}$

Q. Given an arry. find the sum of the Array!

```
~~int~~ long sum = 0;
for(i=0; i<N; i++){
    sum = sum + A[i];
}
ret sum;
```

✓

CONSTRAINTS

$1 <= N <= 10^5$

$1 <= A[i] <= 10^9$

sum $\rightarrow 10^{14}$

Q Given 2 ints a & b, ret a * b !

$\boxed{1 <= b, a <= 10^9}$

```
int ans = a × b;
ret ans;
```

✗ overflow

$10^9 \times 10^9 = 10^{18}$

int + int → int

⇓

```
long ans = a × b;
ret ans;
```

✗

⇓

```
long ans = long(a × b);
ret ans;
```

✗

⇓

```
long ans = long(a) × b;
ret ans;
```

✓

explicit type cast

l × i → l × l → l

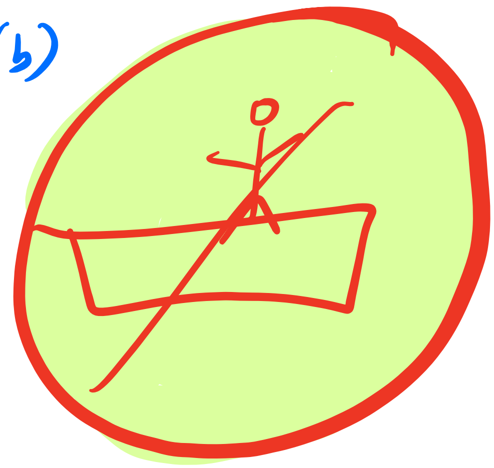
implicit type

type promotion

i + l → l + l → l

long ans = log(a) × log(b)

ret ans;



long ans = a × 1L × b

$i \times l \times i$

ret ans;