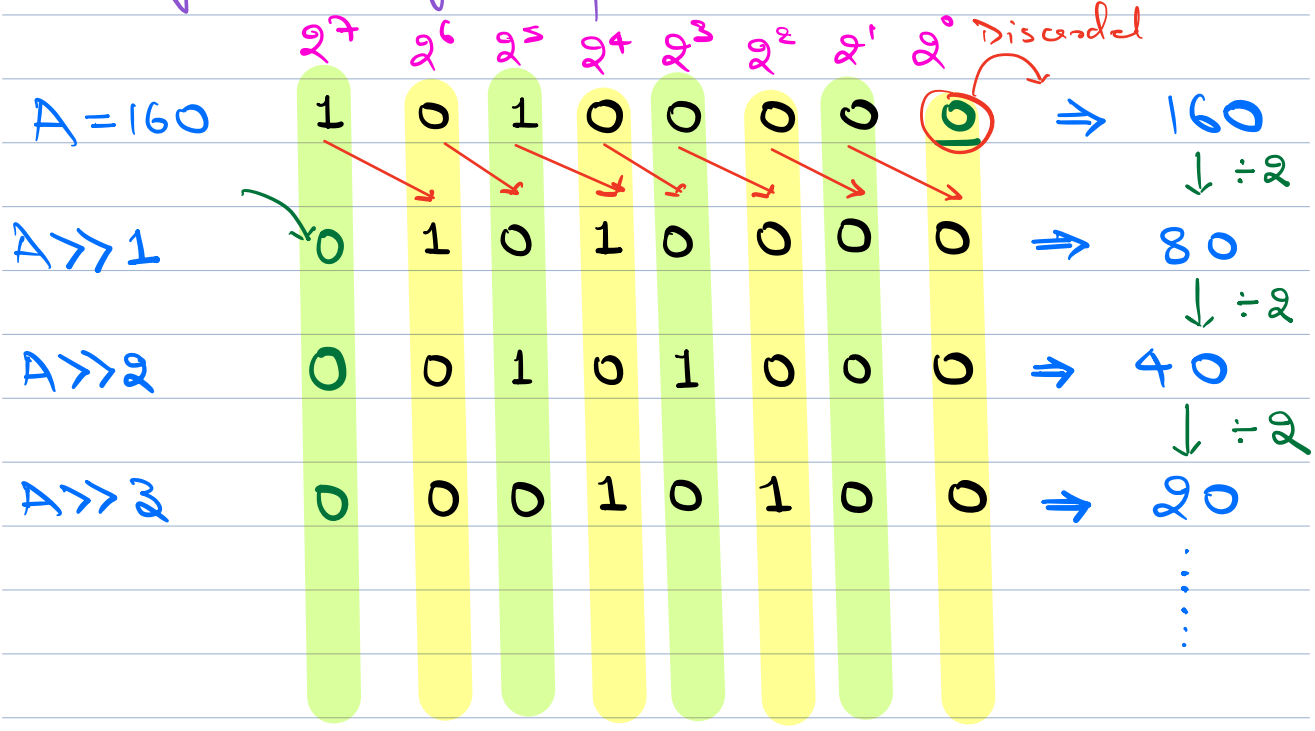
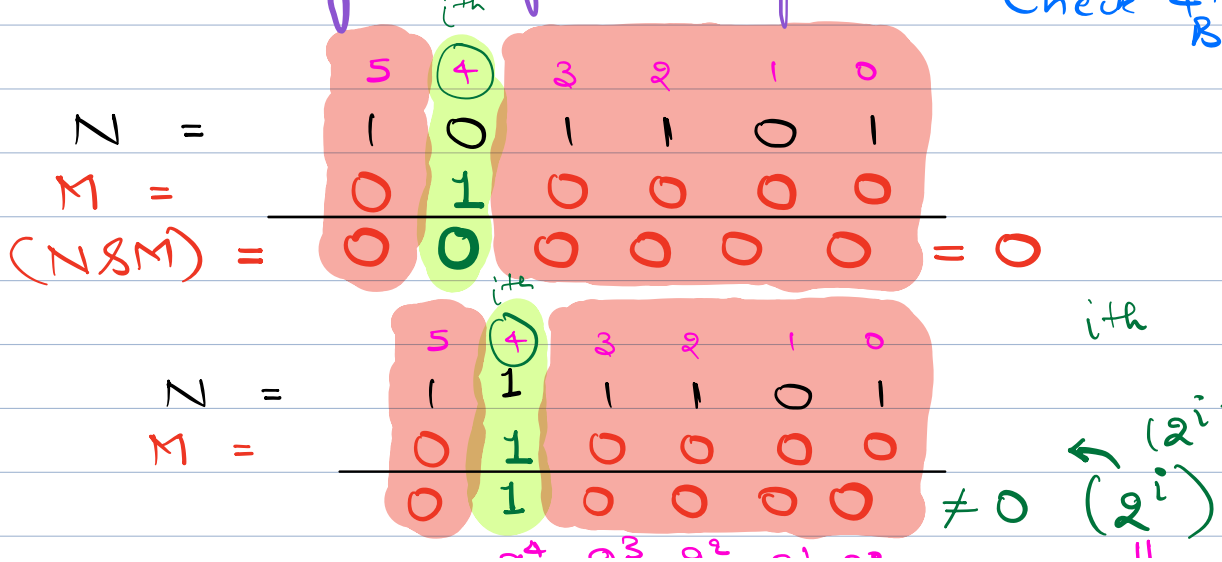
Right Shift Operator (>>)



$$(A >> n) = A / 2^n$$

Power of Left Shift

Check 4^{th} Bit



$2^0\ 2^1\ 2^2\ 2^3\ 2^4 \ldots$ ($\log i$)

$(A << i) = A \times 2^i$

put $A = 1$

$(1 << i) = 2^i$

$N \,\&\, (1 << i)$ → 0 ⇒ i^{th} bit is not set

→ >0 ⇒ i^{th} bit is set

	5	4	3	2	1	0
N =	1	0	1	1	0	1

Given a no. N ⇒ Set i^{th} bit of N.

$i = 4$

	5	4	3	2	1	0
N =	1	0	1	1	0	1
M = OR	0	1	0	0	0	0
	1	1	1	1	0	1

$i = 3$

	5	4	3	2	1	0
N =	1	0	1	1	0	1
M = OR	0	0	1	0	0	0
	1	0	1	1	0	1

$1 \_\_ 0 = 1$

$0 \_\_ 0 = 0$ > (OR) or (XOR)

$1 \_\_ 1 = 1$

$0 \_\_ 1 = 1$ > OR

$N = N \mid (1 << i)$ ⇒ Set i^{th} bit of N

```
int a = 4;
a + 4;
print(a); ⇒ 4
a = a + 4;
print(a) ⇒ 8
```

Given a no. N ⇒ Toggle i^{th} bit of N.

$(1 \to 0)$ & $(0 \to 1)$

$i = 4$

	5	4	3	2	1	0
N =	1	0	1	1	0	1
M = XOR	0	1	0	0	0	0
	1	1	1	1	0	1

$i = 2$

	5	4	3	2	1	0
N =	1	0	1	1	0	1
M = XOR	0	0	1	0	0	0
	1	0	0	1	0	1

$1 \_\_ 0 = 1$
$0 \_\_ 0 = 0$ > (OR) & (XOR)

$1 \_\_ 1 = 0$
$0 \_\_ 1 = 1$ > XOR

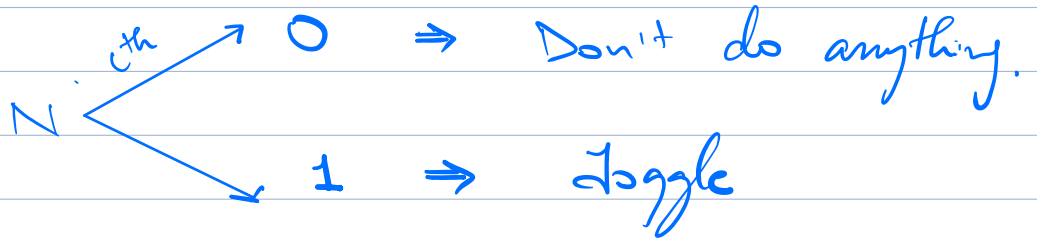
Given a no. N ⇒ Unset the i^{th} bit of the no. N.

$i = 4$

	5	4	3	2	1	0
N =	1	0	1	1	0	1
M =	0	1	0	0	0	0
	1	0	1	1	0	1

$i = 3$

	5	4	3	2	1	0
N =	1	0	1	1	0	1
M =	0	0	1	0	0	0
	1	0	0	1	0	1




```
if ( N & (1<<i) != 0) {  // ith bit is set
    N = N ^ (1<<i);
}
```

Given an integer N. Count the no. of set bits in N.

Eg N = 12 ⇒ 1100 ⇒ 2

```
bool checkBit (N, i) {
    if((N & (1<<i)) == 0){
        return false;
    } else {
        return true;
    }
}
```

```
count = 0;
for (i=0; i<32; i++) {
        if ( checkBit (N, i)) {
                count ++;
        }
}
return count;
```

(N & 1)

N = 1010 00101011

0101 00010101

```
int countBit (N) {
    count = 0;
    while (N>0) {
        if (N&1 == 1) {
            count ++;
        }
        N = (N>>1);
    }
    return count;
}
```

Sam has a cat
Every day the cat behaves good, sam

feeds the cat.

Day 1 ⇒ 1 unit of food if cat behaves good
Day 2 ⇒ 2 units " " " "
3 ⇒ 4 units " " " "
4 ⇒ 8
⋮

Given the total units of food sam has given to the cat. Can you find out the no. of days the cat behaved good.

Total food = 79 units

2^n . . . . . 2^3 2^2 2^1 2^0

1 0 1 0 1 1 0 0 0 0 1 0 1

2^4 2^3 2^2 2^1 2^0

1 0 1 0 1

$1\times2^0 + 0\times2^1 + 1\times2^2 + 0\times2^3 + 1\times2^4$

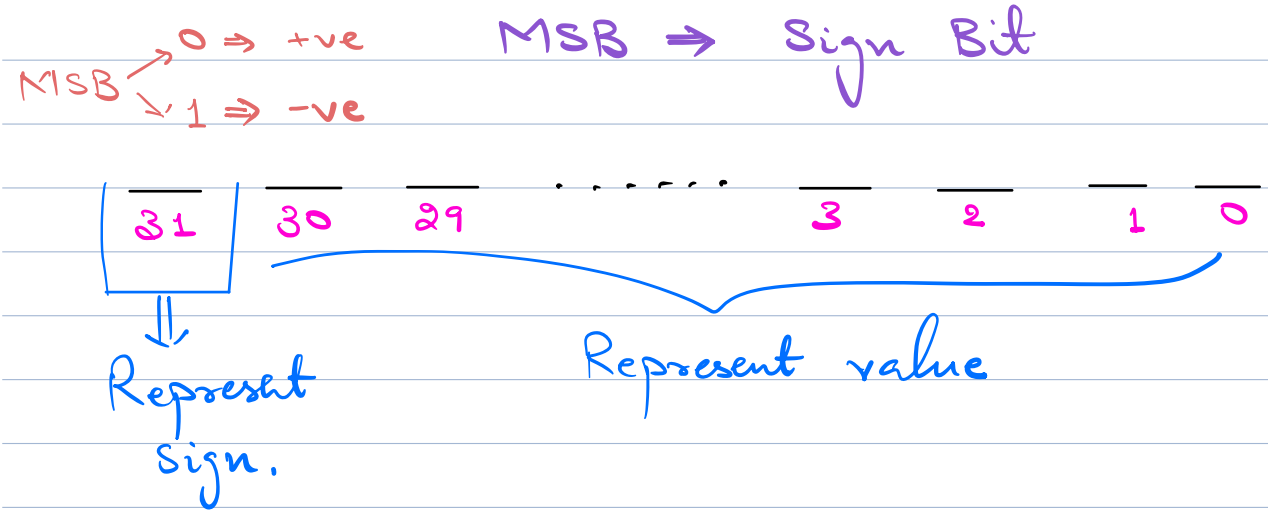
=

Binary Representation of Negative No.

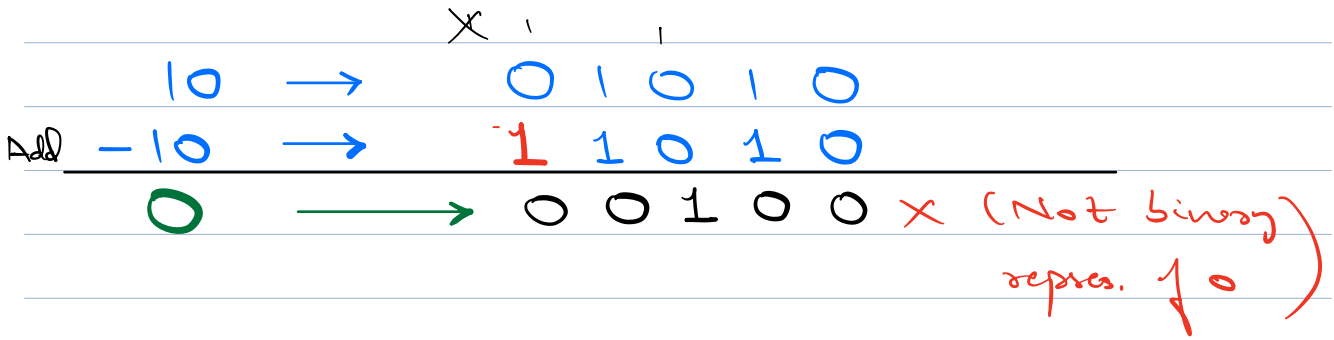
Assume 5 bits (MSB)

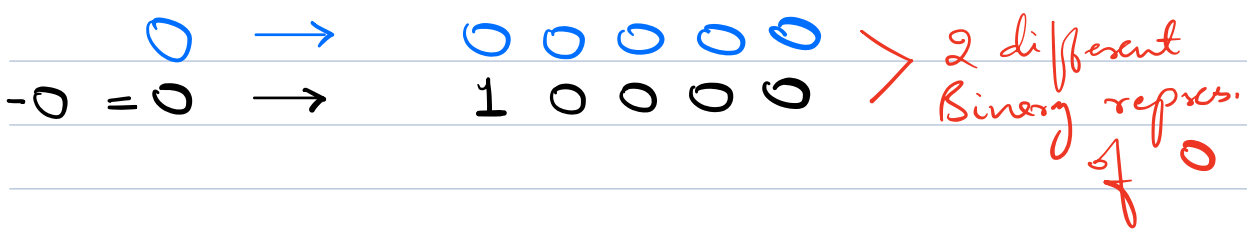
2^4 2^3 2^2 2^1 2^0

10 → 0 1 0 1 0




```
int          x = 5
unsigned int x = 5
```





2's compliment

-10 in binary

1) Take binary representation of 10

$10 \rightarrow 01010$

2) Invert all bits

10101

3) Add 1 to the result of step 2

$$\begin{array}{r} 10101 \\ 00001 \\ \hline -10 \rightarrow 10110 \end{array}$$

$$\begin{array}{rcl} -10 & \Rightarrow & 10110 \\ 10 & \Rightarrow & 01010 \\ \hline 0 & \Rightarrow & 00000 \end{array}$$

Sign Bit

	2^4	2^3	2^2	2^1	2^0
$-10 \Rightarrow$	1	0	1	1	0

$$(-2^4) \times 1 + 2^3 \times 0 + 2^2 \times 1 + 2^1 \times 1 + 2^0 \times 0$$
$$= -16 + 0 + 4 + 2 + 0$$
$$= -10$$

00101

Range of Data Types

N	Signed	Unsigned
2	-2^1 2^0 Max → 0 1 = 1 Min → 1 0 = -2 Range = [-2, 1]	2^1 2^0 Max → 1 1 = 3 Min → 0 0 → 0 Range = [0, 3]
4	-2^3 2^2 2^1 2^0 Max → 0 1 1 1 = 2^3-1 = 7 Min → 1000 → -8 Range = [-8, 7] = $[-2^3, 2^3-1]$	2^3 2^2 2^1 2^0 Max → 1 1 1 1 = 2^4-1 = 15 $2^3+2^2+2^1+2^0$ Min → 0000 = 0 Range → [0, 15] = $[0, 2^4-1]$
32	-2^{31} 2^{30} ..... 2^2 2^1 2^0 Max → 0111...11 → $2^{31}-1$ Min = 100000 → -2^{31} Range = $[-2^{31}, 2^{31}-1]$	2^3 2^{30} ..... 2^2 2^1 2^0 Max = 11...111 = $2^{32}-1$ Min = 000000 = 0 Range = $[0, 2^{32}-1]$
N	-2^{N-1} 2^{N-2} ..... 2^2 2^1 2^0 Max = 0111...111 → $2^{N-1}-1$	2^{N-1} 2^{N-2} ..... 2^2 2^1 2^0 Max = 111111 = 2^N-1

Min = 1000...00 $\Rightarrow -2^{N-1}$

Range = $[-2^{N-1}, 2^{N-1}-1]$

Min = 0000..0 = 0

Range = $[0, 2^{N}-1]$

Importance of Constraints

$2^{31} \simeq 10^9$

```
int a = 10^5
int b = 10^6
int c = (a*b)
```

→ ALU (Arithmetic Logic Unit)

??? (10^{11} ??)

→ Overflow.

How to avoid overflow.

1) `long c = (long)(a*b)` ✗

2) `long c = ((long)a) * b`

2) `long c = a * (long) b`

2) `long c = (long)a * (long)b`

✓

Doubt

20B, 14R

v 2

Same colr → Blue
diff → Red.

2B / 2R ⇒ 1 Blue
↓ ↓ +1 Blue
-1

1R + 1B ⇒ 1 Red Ball ⇒ -1 Blue