SCALER

Bit Manipulation

TABLE OF CONTENTS

Notes

Decimal Number System

[0 — 9] base 10

$324 = 3*10^2 + 2*10^1 + 4*10^0$

$2563 = 2*10^3 + 5*10^2 + 6*10^1 + 3*10^0$

Binary Number System

[0 — 1] base 2

$110 = 1*2^2 + 1*2^1 + 0*2^0 \rightarrow \underline{6}$

$1011 = 1*2^3 + 0*2^2 + 1*2^1 + 1*2^0 \rightarrow \underline{11}$

1. Binary to Decimal Conversion

- $(\overset{3}{1}\overset{2}{1}\overset{1}{0}\overset{0}{1})_2 = (\quad)_{10}$

$$1 * 2^3 + 1 * 2^2 + 0 * 2^1 + 1 * 2^0 = \underline{13}$$

$$\begin{array}{cccc} 3 & 2 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{array}$$

$$2^3 + 2^2 + 2^0 = 8 + 4 + 1 = 13$$

$$\begin{array}{ccccc} 4 & 3 & 2 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{array}$$

$$2^4 + 2^2 + 2^0 = 16 + 4 + 1 = \underline{21}$$

$$\begin{array}{ccccccc} 6 & 5 & 4 & 3 & 2 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \end{array}$$

$$2^6 + 2^4 + 2^3 + 2^1 = 64 + 16 + 8 + 2 = \underline{90}$$

2. Decimal to Binary

- $(20)_{10} = (\quad)_2$

		rem.
2	20	0
2	10	0
2	5	1
2	2	0
2	1	1
	0	

LSB

MSB

$$\begin{array}{ccccc} 4 & 3 & 2 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \end{array}$$

$$2^4 + 2^2 = 16 + 4 = \underline{20}$$

- $(45)_{10} = (\quad)_2$

2	45	1
2	22	0
2	11	1
2	5	1
2	2	0
2	1	1
	0	

$\underline{1\,0\,1\,1\,0\,1}$

Addition of two decimal numbers -

```
    1   1
    3   6   8
+   1   4   5
-------------
    5   1   3   (Ans)
-------------
```

Addition of two binary numbers -

```
  1          1
     1   1   0   1   → 13
+    1   0   0   1   → 9
------------------------
  1  0   1   1   0   → 22
------------------------
```

0 → 0
1 → 1
$\boxed{2 \rightarrow 10}$
3 → 11

```
  1           1
     1   1   0   1   0   1
+    1   0   0   1   1   0
---------------------------
  1  0   1   1   0   1   1
---------------------------
```

```
         1   1
     1   0   1   1   0
+    0   0   1   1   1
-----------------------
     1   1   1   0   1
-----------------------
```

Bitwise Operators

0 → false / unset bit

1 → true / set bit

! , & , | , ^ , << , >> - Advanced BM

NOT AND OR XOR

res = 1, if all bits = 1

1) addition without carry

2) res = 1, if odd # 1's

3) "same same puppy shame."

a	b	a&b	a|b	a^b	~a/!a
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

res = 1, if any bit = 1

flip the bit

5 & 6 = ?

5 →	1	0	1
6 →	1	1	0

&

---

1 0 0 → 4

20 & 45 = ?

20 →	0	1	0	1	0	0
45 →	1	0	1	1	0	1

0 0 0 1 0 0 → 4

20 | 45 = ?

20 →	0	1	0	1	0	0
45 →	1	0	1	1	0	1

1 1 1 1 0 1 → 61

5 4 3 2 1 0

20 ^ 45 = ?

20 →	0	1	0	1	0	0
45 →	1	0	1	1	0	1

1 1 1 0 0 1 → 57

5 4 3 2 1 0

Negative Numbers

int → 32 bits long → 64 bits

for our understanding → 8 bit system

MSB (signed bit) ← LSB

	7	6	5	4	3	2	1	0	
45 →	0	0	1	0	1	1	0	1	
Flip bits →	1	1	0	1	0	0	1	0	[1's complement]

$$
\begin{array}{lcccccccc}
 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\
\text{Add 1} \rightarrow + & & & & & & & & 1 \\
\hline
-45 \rightarrow & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\
\hline
 & 7 & 6 & 5 & 4 & 3 & 2 & 1 & 0
\end{array}
$$

[2's complement]

$$-2^7 + 2^6 + 2^4 + 2^1 + 2^0 = -128 + 64 + 16 + 2 + 1$$

$$= \cancel{211} \quad -128 + 83$$

$$= \underline{-45}$$

MSB

0

↓

No. is positive ✓

1

↓

No. is negative ✓

Binary representation of -3

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

```
  1 1 1 1 1 1 0 0
+               1
-----------------
  1 1 1 1 1 1 0 1  → -3
```

	7	6	5	4	3	2	1	0
10 →	0	0	0	0	1	0	1	0
flip →	1	1	1	1	0	1	0	1

```
              1
  1 1 1 1 0 1 0 1
+               1
-----------------
  1 1 1 1 0 1 1 0  → -10
```

8 bit system → Min → 1 0 0 0 0 0 0 0

$= -2^7 = -128$

$[-2^7 \quad (2^7-1)]$

$[-128 \quad 127]$

MSB

Max → 0 1 1 1 1 1 1 1

$= 2^6+2^5+2^4+2^3+2^2+2^1+2^0$

$= 2^7-1 = 127$

Integer (32 bit system)

Min → $\overset{31}{1}\ \overset{30}{0}\ \overset{29}{0}\ \ldots\ \overset{0}{0}$ $= -2^{31} = -2\,147\,483\,648$

$\approx \underline{-2*10^9}$

Max → $0\ 1\ 1\ \ldots\ 1$ $= 2^{30}+2^{29}+\ldots+2^0$

$= 2^{31}-1 = 2\,147\,483\,647$

$\approx \underline{2*10^9}$

$[-2^{31} \quad 2^{31}-1]$

$[-2*10^9 \quad 2*10^9]$ ✓

Long (64 bit system)

Min → $\overset{63}{1}\ \overset{62}{0}\ \ldots\ \overset{0}{0}$ $= -2^{63} = -9\,223\,372\,036\,854\,775\,808$

$\approx \underline{-9*10^{18}}$

Max → $0\ 1\ \ldots\ 1 = 2^{63}-1$

$= 9\,223\,372\,036\,854\,775\,807$

$\approx \underline{9*10^{18}}$

$[-2^{63} \quad 2^{63}-1]$

$[-9*10^{18} \quad 9*10^{18}]$ ✓

Importance of Constraints (also used to predict TLE) ✓

Q → Given 2 integers a & b find their product.

`int c = a * b;` ✗

$a = 10^5 \quad b = 10^5$

$a*b = 10^{10}$ (outside int range)

long c = $\boxed{a * b}$; ✗ right side calculation → overflow

2 1 → overflow

long c = (long) (a * b) ✗

long c = long(a) * long(b) ✓

long c = long(a) * b ✓

int * int → int

long * int → long

Q → Given an integer array, find sum of elements.

$1 <= N <= 10^5$
$1 <= A[i] <= 10^6$

```
~~int~~ long sum = 0
for i → 0 to (N-1) {
    sum += A[i]
}
return sum
```

max sum = $10^5 * 10^6 = \underline{10^{11}}$

Agenda :-

Bitwise Operators
↳ Properties
↳ few Problems.

int a = 5;
int b = 10;

10 → 1010

no' → Binary

a (Bitwise operator) b

The operators which act upon the binary equivalent.

Truth table of Bitwise Operators

a	b	a & b	a | b	a ^ b (XOR)	~a
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

Basic And Properties

1) Even / odd number

10 → 1 0 1 0

9 → 1 0 0 1

_ _ _ _

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

In binary representation, if a number is even, then its least significant bit (LSB) is 0.

Conversely, if a number is odd, then its LSB is 1.

1) odd / Even Condn.

A&1 → x y z a b c 0/1 (0th bit)

&1 0 0 0 0 0 1

0 0 0 0 0 0/1

A&1 → 1 (A is odd)

A&1 → 0 (A is even)

```
if ((A&1) == 1) {      // no. is odd
}
```

2) $A \& 0 = 0$

$$\begin{array}{lcccccc} A \rightarrow & x & y & z & \alpha & \beta & \gamma \\ \& \; 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

3) $A \& A \longrightarrow A$

$$\begin{array}{lccccc} A \rightarrow & 1 & 0 & 1 & 1 & 0 \\ \& \; A \rightarrow & 1 & 0 & 1 & 1 & 0 \\ \hline A \rightarrow & 1 & 0 & 1 & 1 & 0 \end{array}$$

OR properties

1) $A \mid 0 = A$

$$\begin{array}{lccccc} A \rightarrow & 1 & 0 & 1 & 1 & 1 \\ OR \; 0 \rightarrow & 0 & 0 & 0 & 0 & 0 \\ \hline & 1 & 0 & 1 & 1 & 1 \end{array}$$

2) $A \mid A = A$

$$\begin{array}{lccccc} A \rightarrow & 1 & 0 & 1 & 1 & 0 \\ A \rightarrow & 1 & 0 & 1 & 1 & 0 \\ \hline & 1 & 0 & 1 & 1 & 0 \end{array}$$

XOR Properties

1) $A \wedge 0 = A$

$$\begin{array}{r} A = 1 \\ 0 \\ \hline 1 \end{array} \qquad \begin{array}{r} A = 0 \\ 0 \\ \hline 0 \end{array}$$

$$\begin{array}{lr} A \rightarrow & 1\,0\,1\,1\,1 \\ \text{XOR } 0 \rightarrow & 0\,0\,0\,0\,0 \\ \hline & 1\,0\,1\,1\,1 \end{array}$$

2) $A \wedge A = 0$

$$\begin{array}{r} A = 0 \\ 0 \\ \hline 0 \end{array} \qquad \begin{array}{r} A = 1 \\ 1 \\ \hline 0 \end{array}$$

$$\begin{array}{lr} A \rightarrow & 1\,0\,1\,1\,0 \\ A \rightarrow & 1\,0\,1\,1\,0 \\ \hline & 0\,0\,0\,0\,0 \end{array}$$

Commutative Property

↳ Order doesn't change the result.

$a \& b = b \& a$

$a | b = b | a$

$a \wedge b = b \wedge a$

Associative Property

↳ grouping doesn't impact the overall result.

$(A \& B) \& C = A \& (B \& C)$

$(A | B) | C = A | (B | C)$

$(A \wedge B) \wedge C = A \wedge (B \wedge C)$.

Ques

Evaluate the expression: a ^ b ^ a ^ d ^ b

$a \wedge b \wedge a \wedge d \wedge b$

$\underline{a \wedge a} \wedge b \wedge b \wedge d$

$\underline{0 \wedge b} \wedge b \wedge d$

$\underline{b \wedge b} \wedge d$

$\underline{0 \wedge d} \rightarrow \underline{d}$

Ques

Evaluate the expression: 1 ^ 3 ^ 5 ^ 3 ^ 2 ^ 1 ^ 5

$\underbrace{1 \wedge 1}_{0} \wedge \underbrace{3 \wedge 3}_{0} \wedge \underbrace{5 \wedge 5}_{0} \wedge 2$

$\underline{2}$

Left shift Operator (<<)

Let's say we have 8 bit numbers,

$a = \underline{10}$

	7	6	5	4	3	2	1	0	
a = 10 =	0	0	0	0	1	0	1	0	
dis									
a << 1 =	0	0	0	1	0	1	0	0	=> 20 => 10×2^1
dis									
a << 2 =	0	0	1	0	1	0	0	0	=> 40 => 10×2^2
a << 3 =	0	1	0	1	0	0	0	0	=> 80 => 10×2^3
a << 4 =	1	0	1	0	0	0	0	0	=> 160 => 10×2^4
disc									
a << 5 =	0	1	0	0	0	0	0	0	=> ~~320~~ 64

$$\boxed{a << n = a * 2^n}$$

(assuming no overflow)

$$\boxed{1 << n = 2^n}$$

2^5

$\underline{1 << 5}$

Right Shift Operator (>>)

		7	6	5	4	3	2	1	0		
a = 20	=>	0	0	0	1	0	1	0	0		
a >> 1	=>	0	0	0	0	1	0	1	0	discarded	=> 10
a >> 2	=>	0	0	0	0	0	1	0	1	dis	=> 5
a >> 3	=>	0	0	0	0	0	0	1	0		=> 2
a >> 4	=>	0	0	0	0	0	0	0	1		=> 1
a >> 5	=>	0	0	0	0	0	0	0	0		=> 0

$$\boxed{\begin{aligned} a >> n &= \frac{a}{2^n} \\ 1 >> n &= \frac{1}{2^n} \end{aligned}}$$

$\underline{1 << 3} = \underline{2}^3$.

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

$\rightarrow 2^0$

$\rightarrow 2^1$

$\rightarrow 2^2$

$\rightarrow 2^3$

$1 \times 2^1 + 1 \times 2^3 \Rightarrow \underline{1}0$.

* Set ith bit.

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

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

To set ith bit of a number

N = N | (1<<i)

Toggle ith bit

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

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

N = N ^ (1<<i)

Check bit at Particular idx.

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

$\longrightarrow$ 0

(N>>4) & (1)

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

$\longrightarrow$ non-zero number

N>>3 → 0 0 0 1 0 1
& 1   0 0 0 0 0 1

```
if ( (N & (1<<i)) == 0) {
        ith bit was unset
}
else {
        it was set.
}
```

```
function checkBit (N, i) {

    if ( (N & (1<<i)) == 0) {
        // bit was unset
            return false
    }
    else {
            return True;
    }
}
```

T.C → O(1)
S.C → O(1)

Ques Count no. of set bits in N.

N = 12 ⟶ 1100 → Ans → 2

Approach 1 :-

N = 12

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

T.C → O(1)
S.C → O(1)

Approach 2 :-

N = 1010 ⟶ 8 bit no

N =		0	0	0	0	1	0	1	0
& 1	=	0	0	0	0	0	0	0	1
		0	0	0	0	0	0	0	0

N >> 1 ⟹	0	0	0	0	0	1	0	1
& 1	0	0	0	0	0	0	0	1
	0	0	0	0	0	0	0	1

N >> 2 0 0 0 0 0 0 1 0
& 1 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0

N >> 3 0 0 0 0 0 0 0 1
& 1 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 1

N >> 4 => 0 0 0 0 0 0 0 0 => 0

10 → 5 → 2 → 0

```
ans = 0;
while (N > 0) {
    if ((N & 1) != 0)
        ans++;
    N = (N >> 1);
}
```

constant

3 const

/2 .

Theoretical → T.C → O(log N) | T.C → O(1)

S.C → O(1) .

Ques

IRCTC (India's train ticketing system) wants to improve how it shows train options to its users. They've decided that trains which run more **frequently** should appear higher up in the search results. To figure this out, they look at a **28-day period** to see how often each train runs.

For **each** train, they've come up with a **special number**. This isn't just any number, though. If you were to write it down in binary form (which is like a special code of 0s and 1s), each of the 28 **digits** corresponds to a day in that **period**. A **'1'** means the train runs on that day, and a **'0'** means it doesn't.

Your task is to help **IRCTC** by writing a program. Given a list **A** of these **special numbers** for different **trains**, your program should find the train that runs the most.

ex:- A = [4369, 8738, 8495252]

0 → 3, → 3 set, → 14 set bits

28 bit, 28 bit, 28 bit

000 ——— 1 000 —— . . . 1 00 (5th 0)

2nd train.

Ques unset i^{th} bit of a no' n.

$N = 6$, $\longrightarrow$ $\overset{2}{1}\ \overset{1}{1}\ \overset{0}{0}$ $\quad i = 2$

$\longrightarrow$ 0 1 0 $\longrightarrow$ 2 .

$$
\begin{array}{rccc}
N = & 1 & 1 & 0 \\
1<<2 \ \wedge & 1 & 0 & 0 \\
\hline
 & 0 & 1 & 0 \\
\hline
\end{array}
$$

① check i^{th} bit.

② if bit is set then do XOR.

T.C → O(1)
S.C → O(1).

```
func unset (N, i) {
    if (checkbit (N, i)) {
        N = N ^ (1<<i)
    }
    return N;
}
```

---

wrong, read about it.

$$
\begin{array}{l}
\text{sign} \begin{cases} \underline{1}\ \underline{1}\ \underline{0}\ \underline{1} \rightarrow -5 \\ \underline{0}\ \underline{1}\ \underline{0}\ \underline{1} \rightarrow 5 . \end{cases}
\end{array}
$$

Ques Set Bits in a Range.

A group of computer scientists is working on a project that involves encoding binary numbers. They need to create a binary number with a specific pattern for their project. The pattern requires A 0's followed by B 1's followed by C 0's. To simplify the process, they need a function that takes A, B, and C as inputs and returns the decimal value of the resulting binary number. Can you help them by writing a function that can solve this problem efficiently?

$A=4,\ B=3,\ C=2$ → 28 Ans.

⇒ 0 0 0 0 1 1 1 0 0

e.g

$A=4,\ B=3,\ C=2$

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

C^{th} bit

$A=2,\ B=4,\ C=3$

8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0

```
ans = 0;
for (i=0; i<B; i++) {        → B times.
    N = setBit(N, C+i);
}
```

T → O(B) ≃ O(1).

A = 2, B = 4, C = 3 → 1111000

11 → 100

111 → 1000

1111 → 10000

11111 → 100000

10000 → 1111

1000000 → 111111

1 << B ⇒ 10000

((1 << B) - 1) ⇒ 1111

((1 << B) - 1) << C ⇒ 1111000

Q → Given an integer array where every element occurs twice except for 1 element that occurs once. Find that unique element.

A = [4 5 5 4 1 6 6] Ans = 1

A = [7 5 1 7 5 6 1] Ans = 6

$$\boxed{\begin{array}{c} x \wedge x = 0 \\ x \wedge 0 = x \end{array}}$$

$7 \wedge 5 \wedge 1 \wedge 7 \wedge 5 \wedge 6 \wedge 1$

$= 7 \wedge 7 \wedge 5 \wedge 5 \wedge 1 \wedge 1 \wedge 6$

$= 0 \wedge 0 \wedge 0 \wedge 6 = 6$

```
ans = A[0]
for i → 1 to (N-1) {
    ans ^= A[i]
}
return ans
```

TC = O(N) SC = O(1)

A = [2 3 5 6 3 6 2]

	2	1	0
2 →	0	1	0
3 →	0	1	1
5 →	1	0	1
→ 6 →	1	1	0
3 →	0	1	1
→ 6 →	1	1	0
2 →	0	1	0

# of set bits → even → 0 (2x)

# of set bits → odd → 1 (2x + 1)

#1's → 3 6 3

1 0 1 → 5 (Ans)

Q→ Given an integer array where every element occurs thrice except for 1 element that occurs once. Find that unique element.

A = [4 4 5 4 5 1 6 6 5 6]   Ans = 1

A = [5 7 5 7 3 7 5]

$5 \wedge 7 \wedge 5 \wedge 7 \wedge 3 \wedge 7 \wedge 5$

$= 5 \wedge 5 \wedge 5 \wedge 7 \wedge 7 \wedge 7 \wedge 3$

$= 0 \wedge 5 \wedge 0 \wedge 7 \wedge 3 = \underline{5 \wedge 7 \wedge 3}$ (Not helpful)

A = [5 7 5 7 3 7 5]

	2	1	0
5 →	1	0	1
7 →	1	1	1
5 →	1	0	1
7 →	1	1	1
$\boxed{3}$ →	0	1	1
7 →	1	1	1
5 →	1	0	1
#1's →	6	4	7
(3x)	0	1	1 → 3 (Ans)

(3x+1)

∀ bits, count of 1's

(3x+1) → 1

(3x) → 0

```
ans = 0
for b → 0 to 31 {
    cnt = 0
    for i → 0 to (N-1) {
        if ((A[i] & (1 << b)) > 0)   cnt++
    }
    if ( cnt % 3 == 1)
        ans = ans | (1<<b)   // set b^th bit
}
return ans
```

TC = $\underline{O(N)}$ SC = $\underline{O(1)}$

Find unique element if all other elements are present K times.   K (even) → XOR of all elements

#1's → Kx → 0

Kx + 1 → 1

Q → Given an integer array, every element occur twice except for 2 elements. Find those two unique elements.

A = [4 5 4 1 5 2]   Ans = $\underline{\{1, 2\}}$

Will XOR be helpful?

4 ^ 5 ^ 4 ^ 1 ^ 5 ^ 2 = $\boxed{\underline{1 \wedge 2}}$ = 3

$(x != y) \rightarrow \boxed{x \wedge y > 0}$ → There exist a set bit.

2 1 0

$1 \wedge 2 = 3 \rightarrow 0\ 1\ 1$

one of the two nos. only has this bit as set.

A = [4 5 4 1 5 2]

0^{th} bit $\rightarrow$ 1 {5 1 5} $\rightarrow$ XOR of all elements

$\rightarrow$ 0 {4 4 2} $\rightarrow$

```
xor = A[0]
for i → 1 to (N-1) {
    xor ^= A[i]
}
// Find position of anyone set bit
b = -1
for i → 0 to 31 {
    if ((xor & (1<<i)) > 0) {  // check i^th bit
        b = i
        break
    }
}
x = 0     y = 0
for i → 0 to (N-1) {
    if ((A[i] & (1<<b)) > 0)     x ^= A[i]
    else                         y ^= A[i]
}
return {x, y}
```

TC = O(N + 32 + N) = O(N)

SC = O(1)

Q → Given an integer array find max value of $(A[i] \,\&\, A[j])$ s.t $i \,!= j$.

$$A = \begin{array}{cccc} 0 & 1 & 2 & 3 \\ [5 & 4 & 6 & 8] \end{array}$$

5 → 101

4 → 100

6 → 110

8 → 1000

i	j	A[i] & A[j]
0	1	5 & 4 = 4
	2	5 & 6 = 4
	3	5 & 8 = 0
1	2	4 & 6 = 4
	3	4 & 8 = 0
2	3	6 & 8 = 0

Ans = 4

$$A = \begin{array}{cccc} 0 & 1 & 2 & 3 \\ [21 & 18 & 24 & 17] \end{array}$$

21 → 10101

18 → 10010

24 → 11000

17 → 10001

21 & 18 = 16

21 & 24 = 16

21 & 17 = 17 Ans

18 & 24 = 16

18 & 17 = 16

24 & 17 = 16

Bruteforce → TC = $O(N^2)$ SC = $O(1)$

x: 1 0 0 0 0 y: 0 1 1 1 1

16 > 15

$$A = [\overset{0}{26}\quad \overset{1}{13}\quad \overset{2}{23}\quad \overset{3}{28}\quad \overset{4}{27}\quad \overset{5}{7}\quad \overset{6}{25}]$$

	4	3	2	1	0
26 →	1	1	0	1	0
~~13~~ 0 →	0	1	1	0	1
~~23~~ 0 →	1	0	1	1	1
~~28~~ 0 →	1	1	1	0	0
27 →	1	1	0	1	1
~~7~~ 0 →	0	0	1	1	1
~~25~~ 0 →	1	1	0	0	1
	1	1	0	1	0

1 1 0 1 0 → 26 (Ans)

```
ans = 0
for b → 31 to 0 {
    cnt = 0
    for i → 0 to (N-1) {
        if ((A[i] & (1 << b)) > 0)   cnt++
    }

    if (cnt >= 2) {
        ans = ans | (1<<b)
        for i → 0 to (N-1) {
            if ((A[i] & (1<<b)) == 0)
                A[i] = 0
        }
    }
}  return ans
```

TC = O(N)   SC = O(1)

$$A = \begin{array}{ccc} 0 & 1 & 2 \\ [8 & 9 & 10] \end{array}$$

$8 \rightarrow 1000$

$9 \rightarrow 1001$

$10 \rightarrow 1010$

max & value = 8

H.W → Count # pairs with max AND value.

K non-zero elements → $^{K}C_2 = \frac{K*(K-1)}{2}$