Q Given an array of non-ve integers. Find the pair of integers in this array which has minimum XOR

Google

max XOR ?

A: 0, 2, 5, 7

0 ^ 2 ⟶ 2*

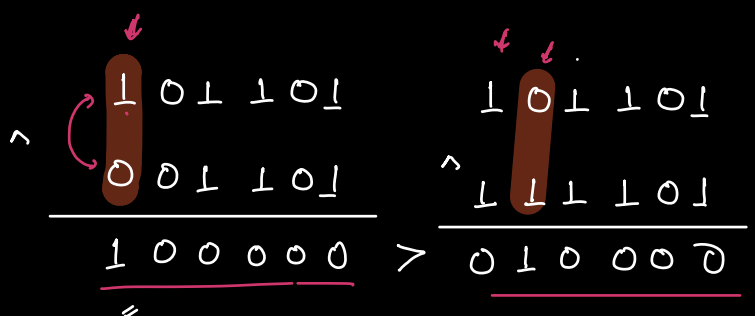
5 ^ 7 ⟶ 2

Brute force : $O(N^2)$

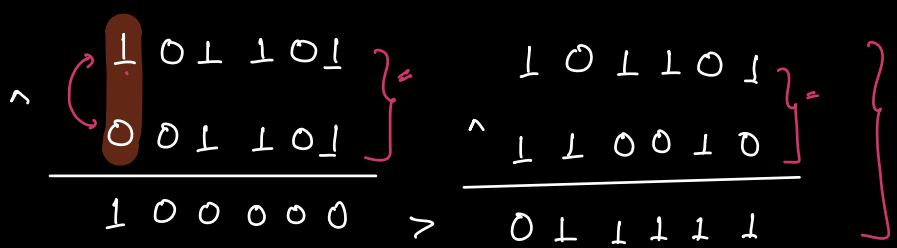
1 ^ 1 ⟶ 0

0 ^ 0 ⟶ 0

a ^ a ⟶ 0



1000 > 0111



a < b < c < d < e

$3 \wedge 4 \rightarrow 7$ ✓

$4 \wedge 5 \rightarrow 6$

$5 \wedge 6 \rightarrow 3$

$6 \wedge 7 \rightarrow 1$

$3 \wedge 5 = 6$

$3 \wedge 5 \geq \begin{cases} 3 \wedge 4 \\ 4 \wedge 5 \end{cases}$ ✓

Assume

$A > C > B$

A : —— 1 ——

C : —— 0/1 ——

B : —— 0 ——

A : 1 0 1 1 0

C : 1 0 0 0 1

B : 1 0 0 0 0

C → 0 C → 1

A : —— 1 ——

C : —— 0 ——

B : —— 0 ——

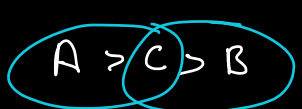
$C \wedge B \rightarrow$ Smaller

A : —— 1 ——

C : —— 1 ——

B : —— 0 ——

$A \wedge C \rightarrow$ Smaller



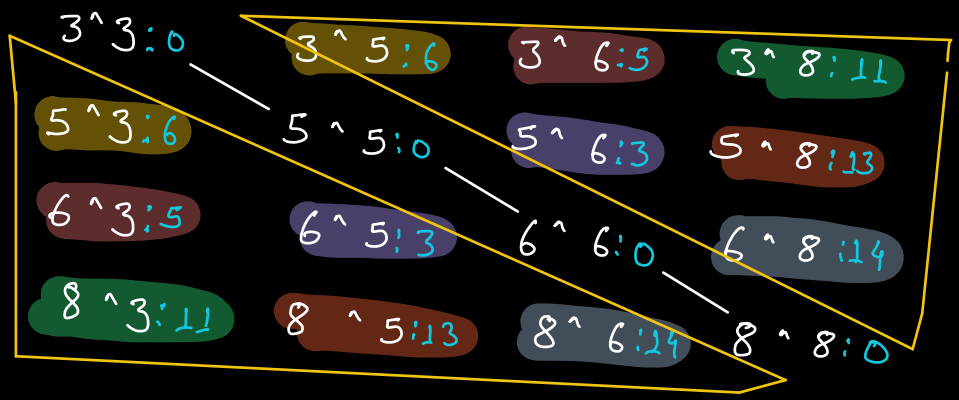
If elements are arranged in Asc order

the min XOR will be found in adjacent elements only.

TC: $O(N \log N)$

Google Given an array of N non-ve integers. Calculate the sum of XOR of all possible pairs $(i, j) \neq (j, i)$

A : 3, 5, 6, 8 (0, 1, 2, 3)



⇒ 104

```
ans = 0;
for (i=0; i<N; i++) {
    for(j=i+1; j<N; j++) {
        ans += (A[i] ^ A[j]);
    }
}
ret 2x ans;
```

TC : $O(N^2)$

2 : 010 3 : 011 5 : 101 6 : 110 8 : 1000

	3	2	1	0
2^3 :	0	0	0	1
2^5 :	0	1	1	1
2^6 :	0	1	0	0
2^8 :	1	0	1	0
3^5 :	0	1	1	0
3^6 :	0	1	0	1
3^8 :	1	0	1	1
5^6 :	0	0	1	1
5^8 :	1	1	0	1
6^8 :	1	1	1	0
Sum :	4	6	6	6
	2^3	2^2	2^1	2^0

	2^2	2^1	2^0
	2^2		
2^3		2^1	
	2^2	2^1	
	2^2		2^0
2^3		2^1	2^0
		2^1	2^0
2^3	2^2		2^0
2^3	2^2	2^1	
	$2^1 \times 6$		6×2^0

$4 \times 2^3 \rightarrow 32$

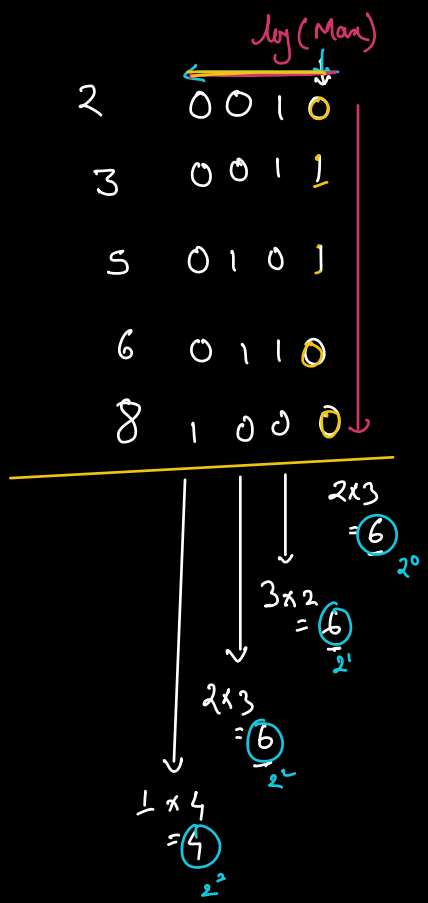
$6 \times 2^2 \rightarrow 24$

$6 \times 2^1 \rightarrow 12$

$6 \times 2^0 \rightarrow 6$

Σ 74

Final Ans = 74×2



$0 \wedge 1 \longrightarrow 1$

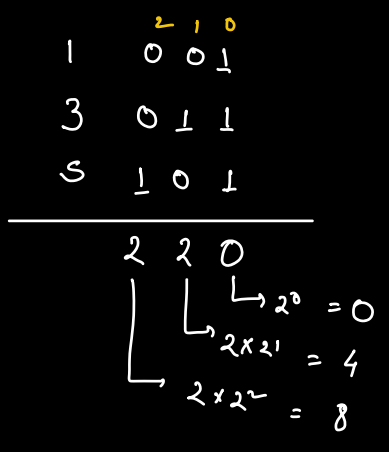
No. of set bits at pos i (in array of XOR pairs) = No of (0,1) pairs at pos i in the original arr.

⇒ No of 1's × No of 0's

TC : $O(N \log Max)$

fn (0 → log(Max))

fn (0 → N)



⇒ 12

Final Ans → 12 x 2 ⇒ 24

```
Sum = 0
for (i = 0 ; i <= MaxBit ; i++) {
        SetBitCount = 0;
        for (j = 0 ; j < N ; j++) {
                if ( CheckBit(A[j], i)) {
                }       SetBitCount++;
        }

        unSetBitCount =  N - SetBitCount;
        Sum  =  sum  +  ( SetBitCount  x  unSetBitCount);
}
ret (Sum x 2);
}
```

Google

Q Given an array of N non-ve integers.
Return the max '&' value of any pair.

Ret max (A[i] & A[j]) [i != j]

A : 27, 18, 20

```
27 : 11011
18 : 10010
---------
&    10010
       18
```

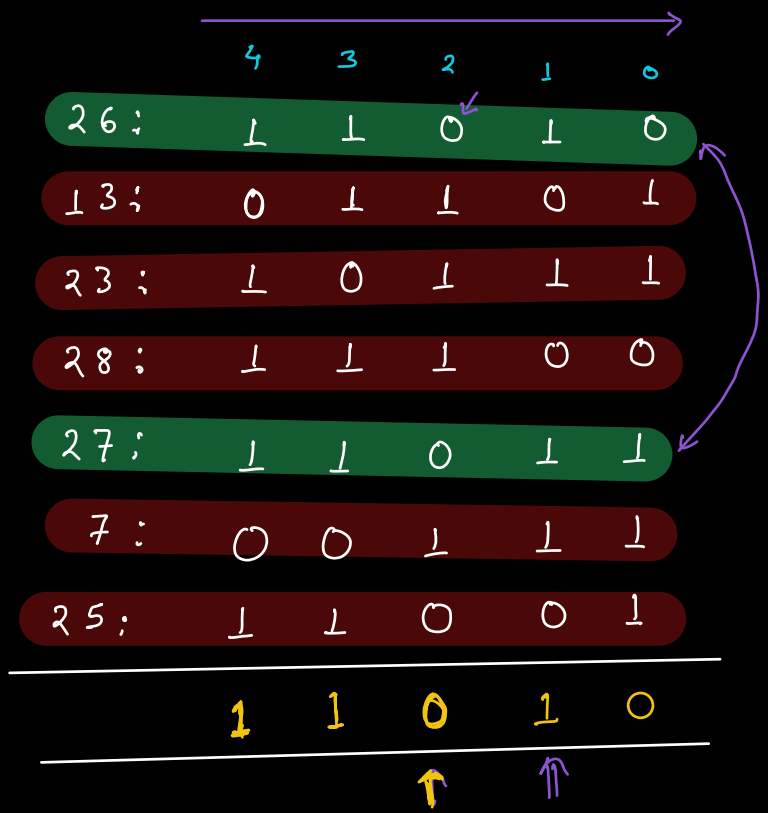
```
27 : 11011
20 : 10100
---------
&  : 10000
       16
```

```
18 : 10010
20 : 10100
---------
&  : 10000
       16
```

Break till 10.37p

Brute force : $O(N^2)$

A : 26, 13, 23, 28, 27, 7, 25



1 & 1 ⟶ 1

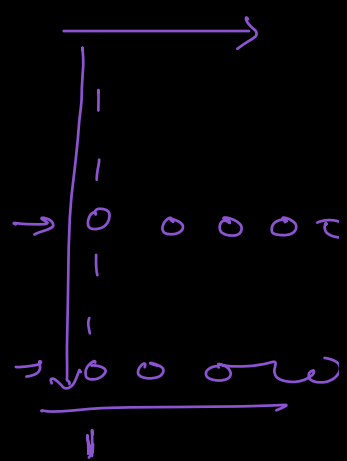
0 & 1
1 & 0 ⟶ 0
0 & 0

10000 > 01111

if Count (setbit) at pos i
≥ 2
→ Ans will have a set bit
at pos i

```
int ans = 0;
for( i = MaxBit ; i >= 0; i--){
    SetBit = 0;
    for(j= 0; j< N; j++ ) {
        if ( checkBit (A[j], i)){
            SetBit ++;
        }
    }
    if ( SetBit >= 2) {
        // Set ith bit in ans
        ans = ans | (1<<i);
        // Remove all no. which have unset bit at ith pos
        for (j=0; j<N; j++) {
            if ( ! CheckBit( A(j) , i)){
                A(j) =0;
            }
        }
    }
}
ret ans;
}
```

A[i] datatype	MaxBit = log(max)
⇒ int	31
⇒ long	63



TC : $O(\log Max \times (N+N)) \Rightarrow O(N \log Max)$

SC : $O(1)$

HW: * Return any one pair with max '&'

* A & B & C ? A & B & C & D ?

Google