Bit Manipulation 2

Content

— Single No. 1 2 3

— MAX AND pair.

---

last PSP: 66.6 %

current: 64.6 % ⟶ 75 %

get as close to 100 %

Rs 1000 for a single question

↓

500 + questions. ≈ JB 5 lacs

Q> Given int[] A. Every no. occurs twice except one number. Find that unique number.

1	3	5	3	2	1	5	=	ans 2

Approach ⟶ XOR all elements.

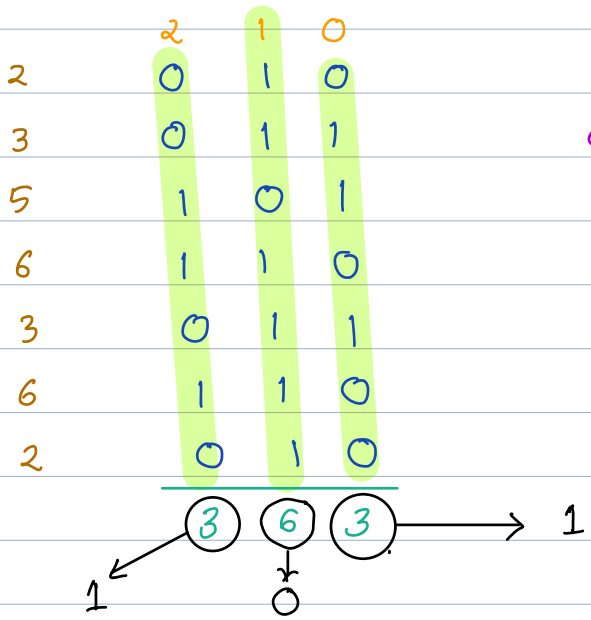
Pseudo code

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

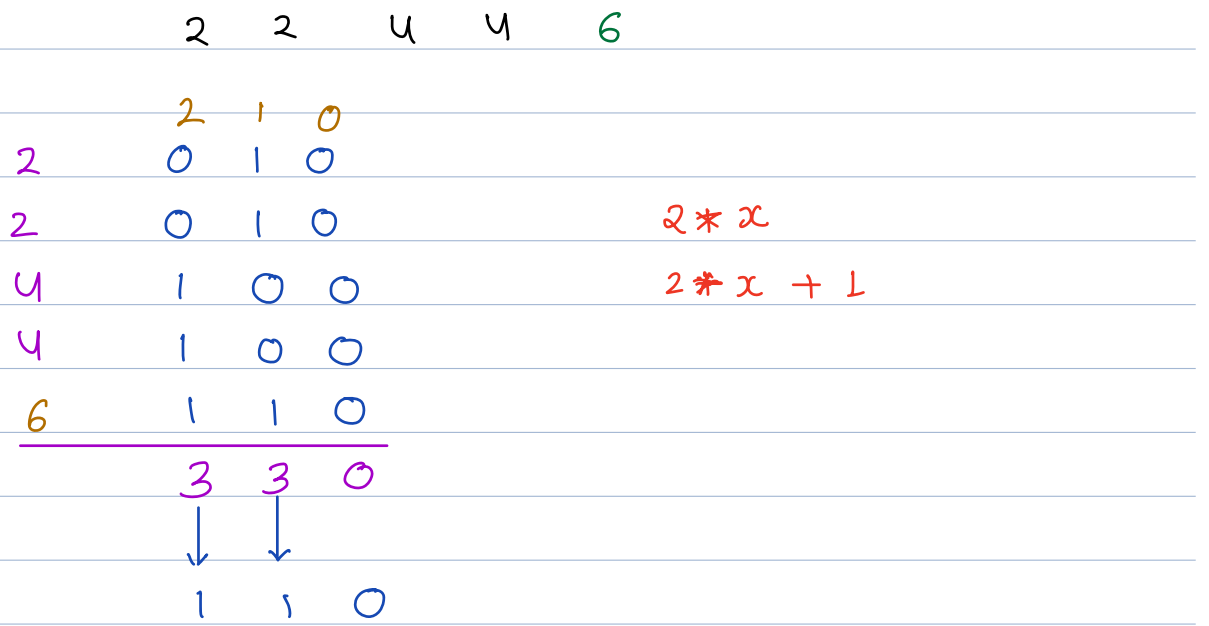
TC : O(N)
SC : O(1)

Approach 2 ⟶ Bit by Bit

A = 2 3 5 6 3 6 2 ans = 5



ans = 31 . . 3 2 1 0
0 0 0 .. 0 0 0 0
1 1
101



Pseudocode

```
ans = 0
for bit ⟶ 0 to 31 {
    count = 0
    for i ⟶ 0 to N-1 {
        if (checkBit(A[i], bit) {
            count++
        }
    }
    if(count % 2 == 1) { // count & 1 == 1
        ans |= (1 << bit)
    }
}
print(ans)
```

N & (1<<i) > 0

TC: O(N)

SC: O(1)

Single Number 2

Given an int[] A, all the elements will occur thrice but one. Find the unique element.

A = 4 5 5 4 1 6 6 4 5 6

ans = 1

Brute force ⟶ ∀i check freq A[i] == 1

TC : O(N²)

SC : O(1)

Approach 2 ⟶ use hashmap

TC : O(N)

SC : O(N)

A = 5 7 5 9 7 11 11 7 5 11

	3	2	1	0
5	0	1	0	1
7	0	1	1	1
5	0	1	0	1
9	1	0	0	1
7	0	1	1	1
11	1	0	1	1
11	1	0	1	1
7	0	1	1	1
5	0	1	0	1
11	1	0	1	1

3 6 6 9 ⟶ multiple of 3 ∵ no. repeat 3 times

9 → 1 0 0 1

4 6 6 10

1 0 0 1

Pseudocode

```
ans = 0
for bit ⟶ 0 to 31 {
    count = 0
    for i ⟶ 0 to N-1 {
        if ( checkBit( A[i], bit) {        // N & (1<<i) > 0
            count++
        }
    }
    if( count % 3 == 1 ) {                 // count & i == 1
        ans |= (1 << bit)
    }
}
print (ans)
```

TC: O(N)

SC: O(1)

⟶ Every no. repeats (4) (K) times except 1 ⟶ take XOR

if K is even ⟶ take XOR

if K is odd ⟶ count % K == 1

Q> *** Given int[] A. Every no. occurs twice except two numbers. Find those two unique numbers.

A = 4 5 4 (1) 6 6 5 (2)

ans = 1, 2

Brute force ⟶ ∀ i calculate freq A[i]

if freq == 1

print (A[i])

A = 5 5 4 4 1 2

1 0 0 1
2 0 1 0
0 1 1

XOR = 5 ^ 5 ^ 4 ^ 4 ^ 1 ^ 2 = 1 ^ 2 = 3

This implies 0th bit for unique no. are diff at 0th bit.

5	1	0	1
5	1	0	1
4	1	0	0
4	1	0	0
1	0	0	1
2	0	1	0

0th bit is 0

4	1	0	0
4	1	0	0
2	0	1	0

2

0th bit is 1

5	1	0	1
5	1	0	1
1	0	0	1

1

A = 4 5 4 8 6 6 5 2

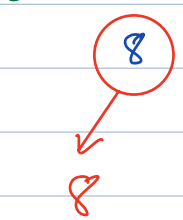
XOR = 8 ^ 2 =

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

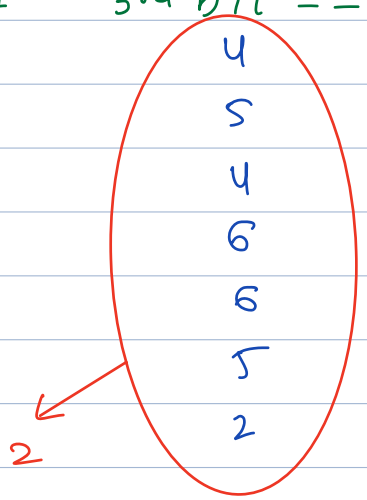
8 1000
2 0010
1010

1st & 3rd bit is diff.

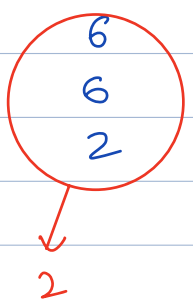
3rd bit == 1



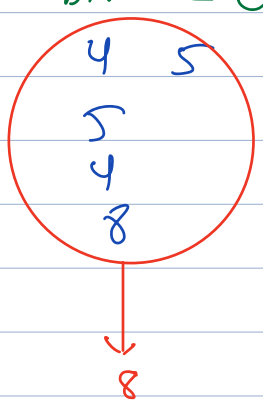
3rd bit == 0



1st bit == 1



1st bit == 0



Pseudocode

```
// XOR all
xor = 0
for i ——→ 0 to N-1 {
    xor ^= A[i]
}
```

```
// find any set bit in xor
pos = -1
for bit ⟶ 0 to 31 {
    if (checkBit(xor, bit)) {
        pos = bit
        break
    }
}
```

```
ans1 = 0
ans2 = 0
for i ⟶ 0 to N-1 {
    if (checkBit(A[i], pos)) {
        ans1 ^= A[i]
    }
    else {
        ans2 ^= A[i]
    }
}
```

TC : O(N)

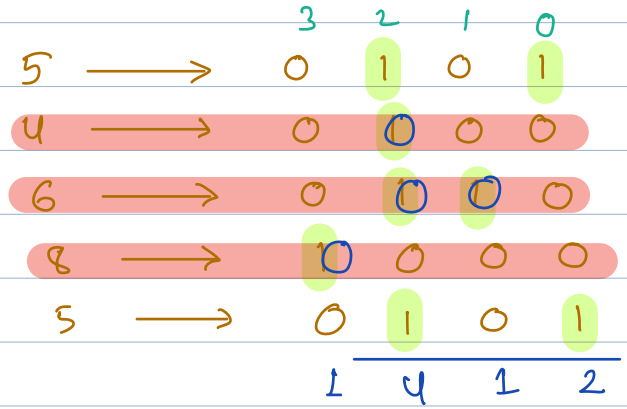
SC : O(1)

```
print (ans1, ans2)
```

Break : 22 : 33

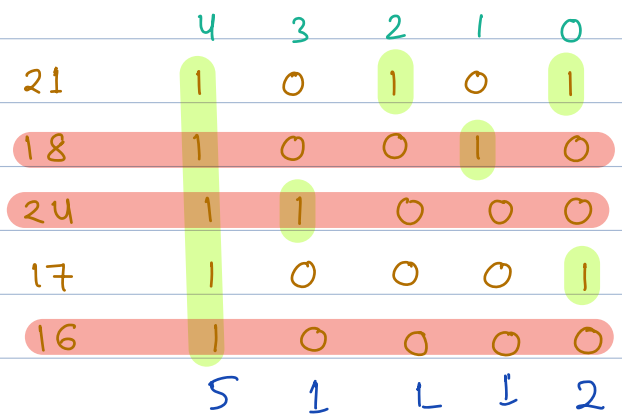
Q> ***** Given an int[] A. Find max value of A[i] & A[j] ∀(i,j) pairs. (i != j)

A = [0:5, 1:4, 2:6, 3:8, 4:5] ⟶ ans = 5 & 5 = 5



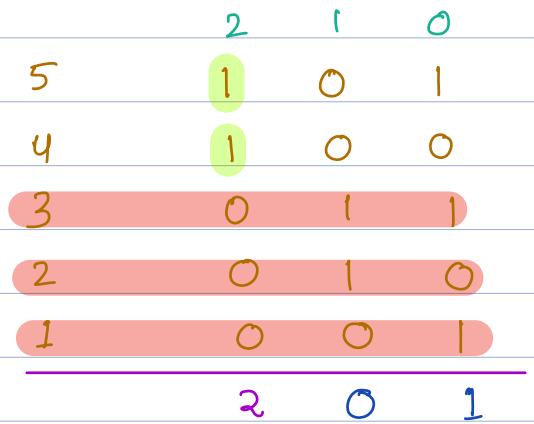
5 0 0 0 5

A = 21 18 24 17 16



21 & 17 = 17

A = 5 4 3 2 1



5 & 4 = 4

Bruteforce $\forall i, j$ $\max(A[i] \& A[j])$

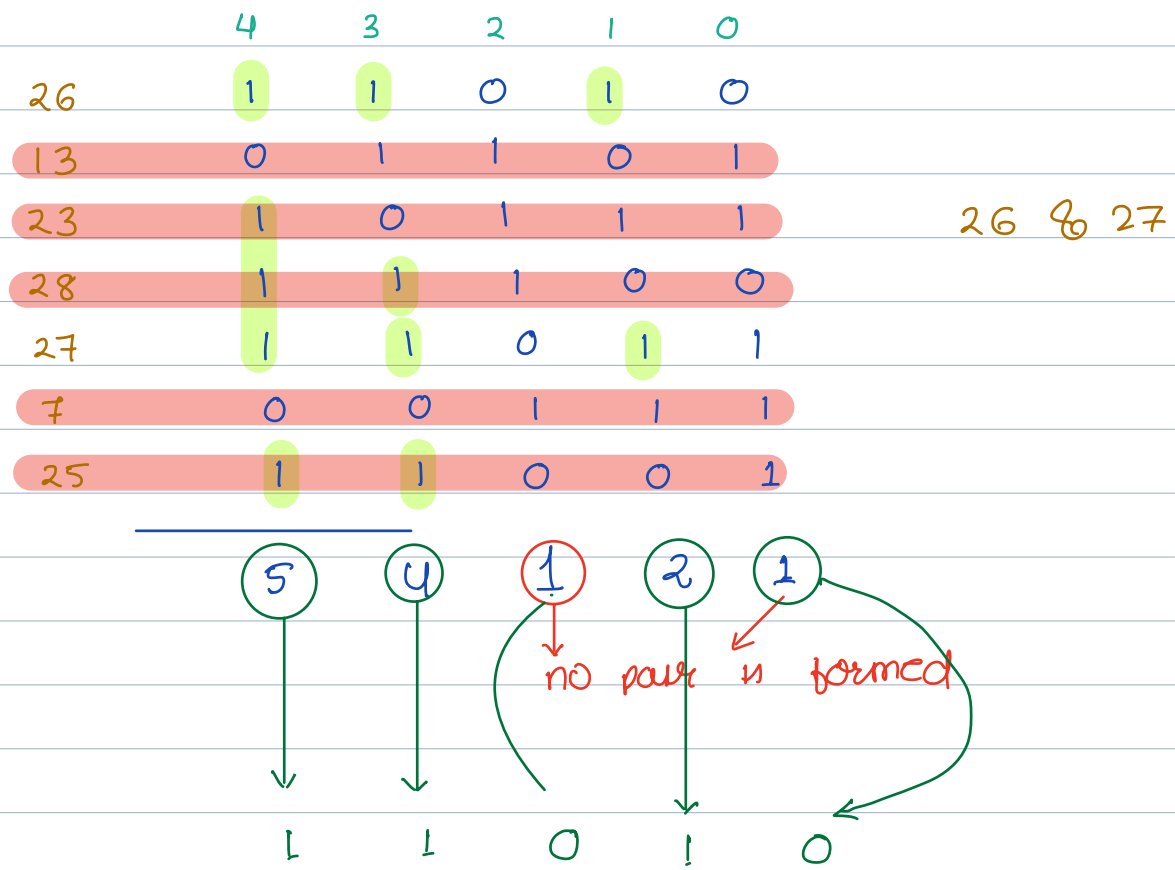
TC: $O(N^2)$

SC: $O(1)$

1 0 0 0 > 0 1 1 1

We always want to set the MSB.

A = 26 13 23 28 27 7 25



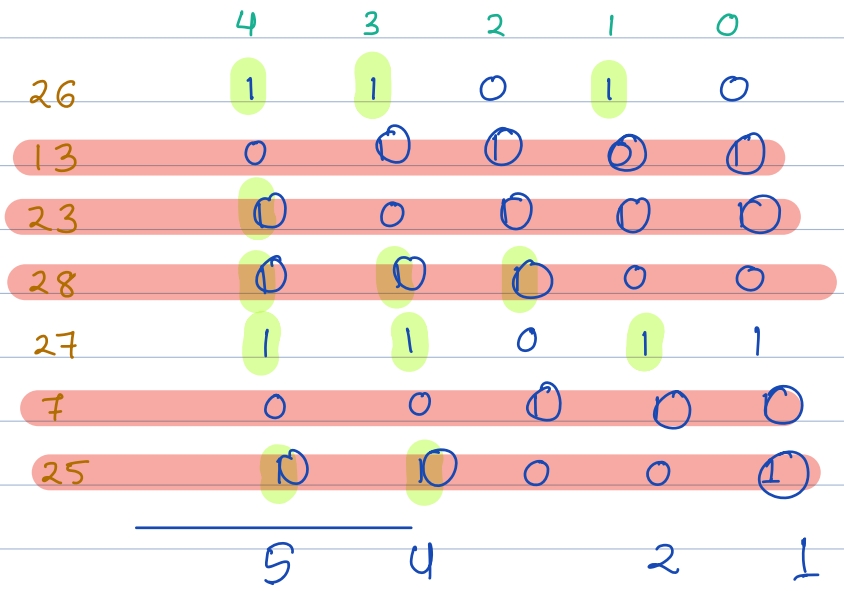
Pseudo ⟶

```
ans = 0

for bit ⟶ 31 to 0 {
    count = 0

    for i ⟶ 0 to N-1 {
        if ( checkBit ( A[i], bit) ) {
            count++
        }
    }

    if ( count >= 2 ) // we can form a pair
        ans |= (1 << bit) // set bit in ans

        // mark A[i] == 0 if mc bit is unset

        for j ⟶ 0 to N-1 {
            if ( !checkBit(A[j], bit)) {
                A[j] = 0
            }
        }
}
print (ans)
```

TC: O(N)

SC: O(1)

A = 26 13 23 28 27 7 25



ans = 26

~~0~~	~~0~~	0	~~0~~	0
1	1		1	

Calculate the no. of Max AND pairs (H.W.)

A = [9 9 9]

ans = 3

From N elements I need to choose 2

$= {}^{N}C_{2} = \frac{n*(n-1)}{2}$

A = 9 10 12

9 & 10 = 8

9 & 12 = 8

10 & 12 = 8

	0	0	0	0
9	1	0	0	1
10	1	0	1	0
12	1	1	0	0
	3			