Q → $a \wedge b \wedge c \wedge b \wedge d \wedge c \wedge a$ $\quad\quad$ $\wedge \rightarrow$ XOR

$= a \wedge a \wedge b \wedge b \wedge c \wedge c \wedge d$

$= 0 \wedge 0 \wedge 0 \wedge d = d$

Q → We are given an integer array where every number occurs twice except for 1 number. Find that number.

	0	1	2	3	4	5	6
A = [	4	5	5	1	4	6	6]

Ans = 1

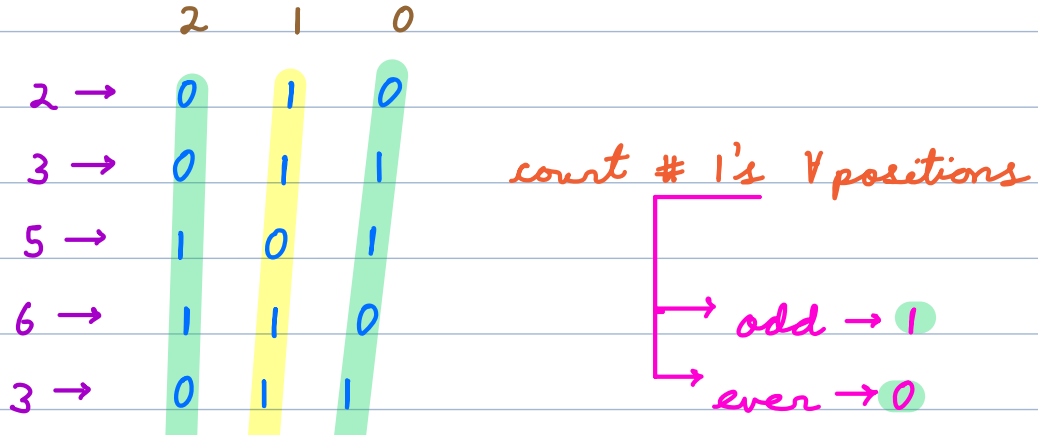
Bruteforce → TC = $O(N^2)$ $\quad$ SC = $O(1)$

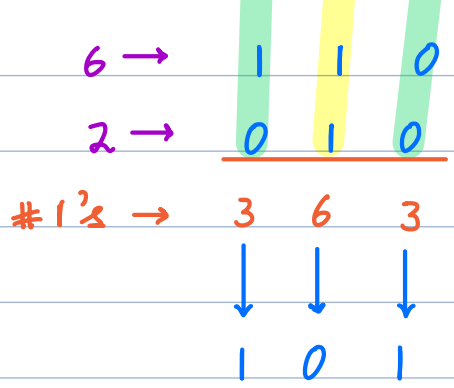
Sol → Ans = $\wedge_{\forall i}$ A[i]

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

TC = O(N) $\quad$ SC = O(1)

	0	1	2	3	4	5	6
A = [	2	3	5	6	3	6	2]






```
ans = 0
for i → 0 to 31 {
    cnt = 0
    for j → 0 to (N-1) {
        // check i^th bit in A[j]
        if ((A[j] & (1<<i)) > 0) cnt++
    }
    if ((cnt & 1) == 1)  // (cnt % 2 == 1)
        // set i^th bit in ans
        ans |= (1<<i)
}
return ans
```

$32 * N \rightarrow N$

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

Q → Given an integer array where all the elements occur thrice except 1 element which occur once. Find that unique element.

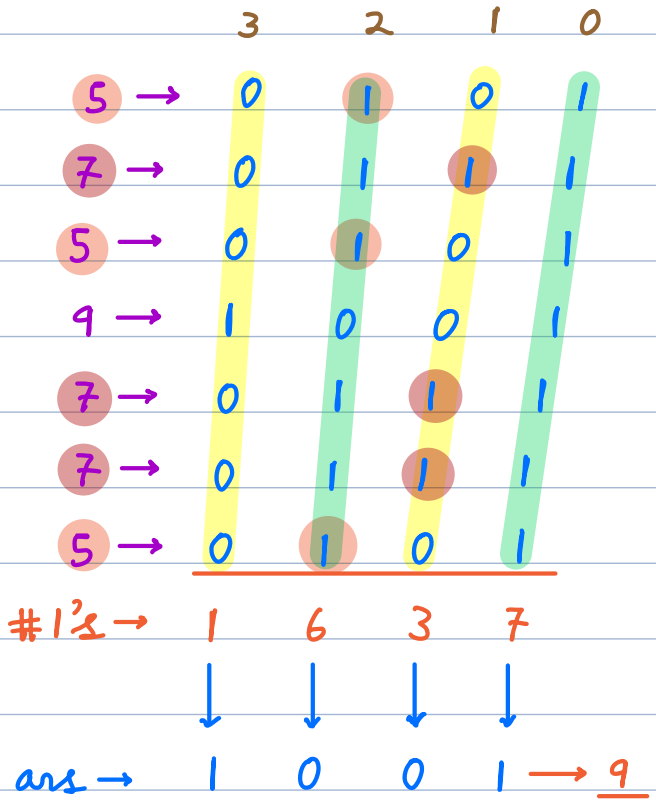
	0	1	2	3	4	5	6	7	8	9
A = [	4	5	5	4	1	6	6	4	5	6]

Ans = 1

Bruteforce → TC = $O(N^2)$ SC = O(1)

$5 \wedge 5 \wedge 5 = 5$

A = [5 7 5 9 7 7 5]



count of 1's = cnt

cnt % 3 → 0 → 0

→ 1 → 1

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

32 * N → N

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

cnt % K

If all numbers occurs K times & 1 unique element is present in the array.

Q → Given an integer array where every element occurs twice except for 2 elements, find those 2 elements which occur once.

```
      0  1  2  3  4  5  6  7
A = [ 4  5  4  1  6  6  5  2 ]      Ans → 1, 2
```

Bruteforce → TC = O(N²) SC = O(1)

XOR of all → ~~4~~ ^ ~~5~~ ^ ~~4~~ ^ 1 ^ ~~6~~ ^ ~~6~~ ^ ~~5~~ ^ 2 = 1 ^ 2 = 3

always have a set bit

0 1 1

$x \wedge y > 0$ $(x != y)$

9 ^ 10

```
  1 0 0 1
^ 1 0 1 0
---------
  0 0 1 1
```

```
      0  1  2  3  4  5  6  7
A = [ 4  5  4  1  6  6  5  2 ]
```

xor of all → 3

pos → 1 (or 0) (2 1 0: 0 1 1)

set → 6 6 2 ⟶ XOR of all = 2

unset → 4 5 4 1 5 " = 1

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

```
p = 0
for i → 0 to 31 {
    if ((x & (1 << i)) > 0) {
        p = i
        break
    }
}
```

```
n1 = 0      n2 = 0
for i → 0 to (N-1) {
    if ((A[i] & (1 << p)) > 0)
        n1 ^= A[i]
    else
        n2 ^= A[i]
}
return {n1, n2}
```

A = [5 10 5 6 10 9] (indices 0 1 2 3 4 5)

x = 6 ^ 9

	0	1	1	0
	1	0	0	1
	1	1	1	1 → 15
				0

p = 0

n1 = 0 ^ 5 ^ 5 ^ 9 = 9

n2 = 0 ^ 10 ^ 6 ^ 10 = 6

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

Q → Given N array elements, choose 2 indices (i, j) s.t (i != j) and (A[i] & A[j]) is maximum.

A = [5 4 6 8] (indices 0 1 2 3)

5 →	1 0 1	5 & 4 = 4	4 & 6 = 4
4 →	1 0 0	5 & 6 = 4	4 & 8 = 0
6 →	1 1 0	5 & 8 = 0	6 & 8 = 0
8 →	1 0 0 0		

Ans = 4

A = [21 18 24 17 16]
0 1 2 3 4

5 4 3 2 1 0

21 → 0 1 0 1 0 1

18 → 0 1 0 0 1 0

24 → 0 1 1 0 0 0

17 → 0 1 0 0 0 1

16 → 0 1 0 0 0 0

Ans = 21 & 17 = 17

A = [5 4 3 2 1]
0 1 2 3 4

Ans = 5 & 4 = 4

Bruteforce → ∀ pairs (i, j) check A[i] & A[j] s.t i != j

TC = O(N²) SC = O(1)

Observations → 1) If i^{th} bit is set in both numbers, then only its set in & of both.

2)

(1 0 0 0) > (0 1 1 1)

8 > 7

check bits from left to right.

A = [26 13 23 28 27 7 25]
0 1 2 3 4 5 6

4 3 2 1 0

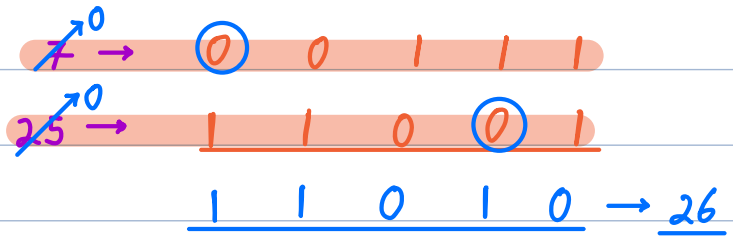
26 → 1 1 0 1 0

13 → 0 1 1 0 1

23 → 1 0 1 1 1

28 → 1 1 1 0 0

27 → 1 1 0 1 1




```
ans = 0
for i → 31 to 0 {
    cnt = 0
    for j → 0 to (N-1) {
        if ((A[j] & (1<<i)) > 0)
            cnt++
    }
    if (cnt >= 2) {
        ans |= (1<<i)
        for j → 0 to (N-1) {
            if ((A[j] & (1<<i)) == 0)
                A[j] = 0
        }
    }
} return ans
```

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

HW → Calculate the count of pairs for which bitwise & is maximum.