Ques

We are given an integer array where every number occurs twice except for one number which occurs just once. Find that number.

Ex 1) 4, 5, 5, 4, 1, 6, 6 → 1

Ex 2) 7, 5, 5, 1, 7, 6, 1, 6, 4 → 4

Idea 1 :- Brute force soln.

Traverse the array and count frequency of every element one by one.
T.C - O(N^2)
S.C - O(1)

Idea 2 :- use hashmap

Traverse the array and store frequency of every element.
T.C - O(N)
S.C - O(N)

Idea 3 :- A ^ A = 0, a ^ b ^ c = b ^ c ^ a.

Value of :- **120 ^ 5 ^ 6 ^ 6 ^ 120 ^ 5**

↓

$\underbrace{120 \wedge 120}_{0} \wedge \underbrace{6 \wedge 6}_{0} \wedge \underbrace{5 \wedge 5}_{0} \rightarrow 0.$

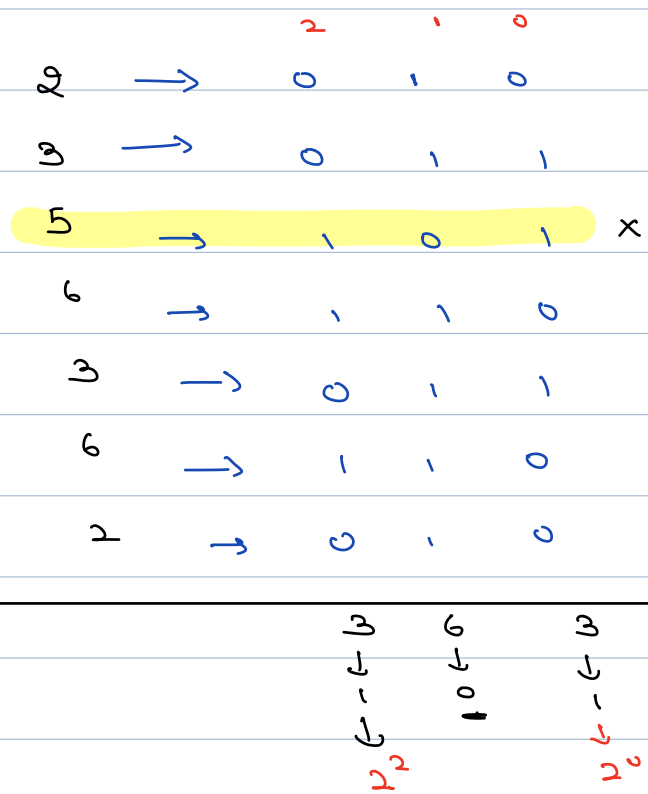
Since ^ helps to cancel out same pairs, we can use it.
Take XOR of all the elements.

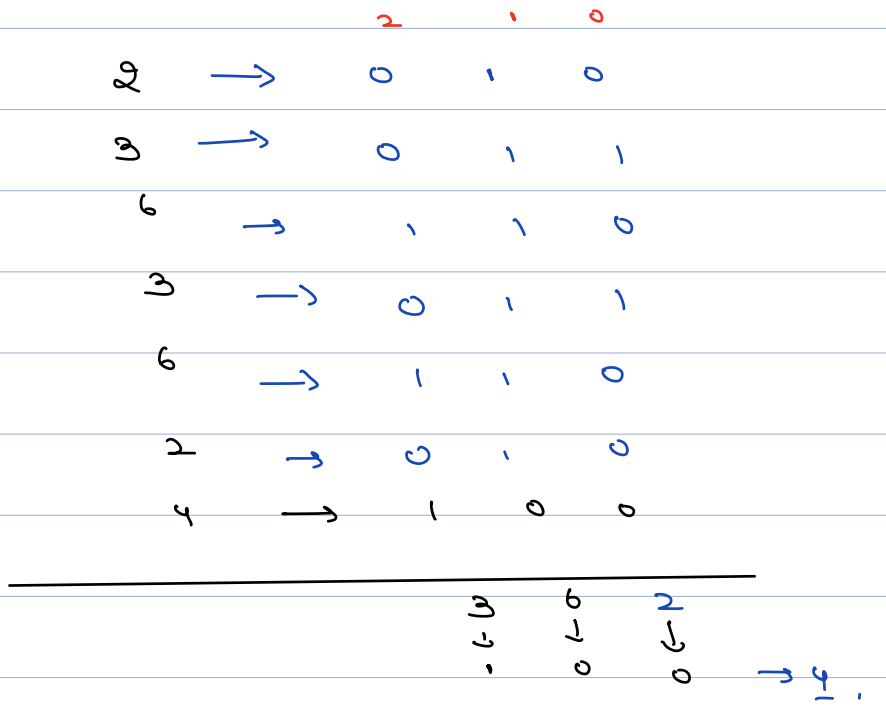
```
int x = 0;

for (int i = 0; i < arr.size(); ++i) {
    x = x ^ arr[i]; // XOR operation
}

print(x);
```

T.C → O(n)
S.C → O(1)

Idea 4 :-

A = [2, 3, 5, 6, 3, 6, 2]






```
int ans = 0;
for (i=0; i<31; i++) {
    int cnt = 0;
    for (j=0; j<arr.size(); j++) {
        if (checkBit(arr[j], i)) {
            cnt++;
        }
    }
    if (cnt % 2 != 0) {
        ans = 2^i
    }
}
```

T.C → O(n)
S.C → O(1).

(cnt%2) != 0

↳ 1<<i

Ques

Given an integer array, all the elements will occur thrice but one. Find the unique element.

→ [4, 5, 5, 4, 1, 6, 6, 4, 5, 6] → Ans : 1

Idea 1 :- Brute force Soln.

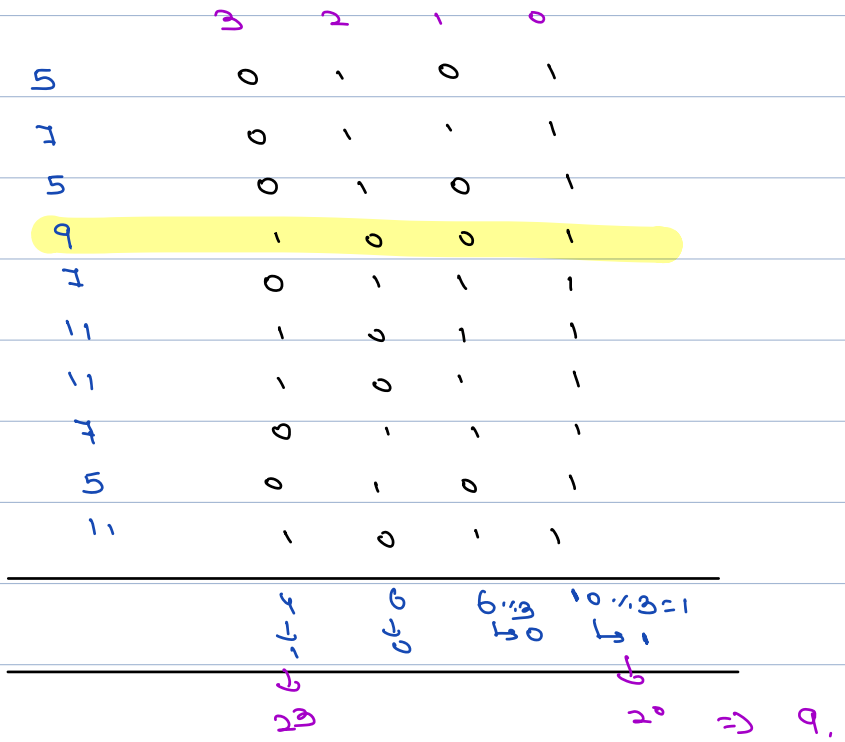
Traverse the array and count frequency of every element one by one.
T.C - O(N^2)
S.C - O(1)

Idea 2 :- use hashmap

Traverse the array and store frequency of every element.
T.C - O(N)
S.C - O(N)

Approach 3 :-

arr[] = {5, 7, 5, 9, 7, 11, 11, 7, 5, 11}



```
int ans = 0;                                   T.C → O(n)
for (i=0; i<31; i++) {                         S.C → O(1)
    int cnt = 0;
    for (j=0; j<arr.size(); j++) {
        if (checkBit(arr[j], i)) {
            cnt++;
        }
    }
    if (cnt % 3 != 0) {
        ans = 2^i
              ↳ 1<<i
    }
}
```

Every no. is coming 3 time, except 1 unique element that is coming 2 time.

Break 9:55 - 9:58pm Revision,

9:58pm - 10:05pm Rest

Ques Single number - 3.

Given an integer array, all the elements will occur twice except two. Find those two elements.

distinct elements

arr[] = {4, 5, 4, 1, 6, 6, 5, 2} → 1 & 2.

arr[] = {4, 9, 9, 8} → 4 & 8

Idea 1 → Take XOR. 3, 7, 6, 7, 3, 8, 9, 9 → 6 & 8

↓

14

arr[12] =

(1010)		(1000)		(1100)		(0110)		(1010)	(1100)	
10	8	8	9	12	9	6	11	10	6	12	17
	(1000)		(1001)		(1001)		(1011)		(0110)		(10001)

Take XOR

↓

Array

↓

	4	3	2	1	0
11 →	0	1	0	1	1
^ 17 →	1	0	0	0	1
	1	1	0	1	0

Divide the array based on 1st bit

set: 10, 10, 6, 6, 11 → 11

unset: 8, 8, 9, 9, 12, 12, 17 → 17

3rd bit

set: 12, 12, 10, 10, 8, 8, 9, 9, 11

unset: 6, 6, 17

① Step 1 :- XOR of Array.

```
v = 0;
for (i=0; i<n; i++) {
    v = v ^ arr[i];
}
```

② Step 2 :- from v get, get a set bit pos.

```
for (i=0; i<31; i++) {
    if (checkbit(v,i)) {
        pos = i
        break;
    }
}
```

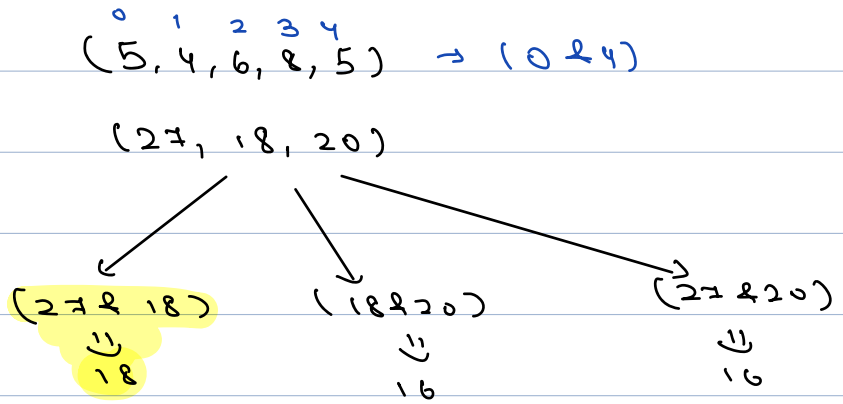
③ Split array based on pos idx, into set & unset.

```
set = 0; unset = 0;
for i = 0 to n-1
    if (checkbit(arr[i], pos))
        set ^= arr[i];
    else {
        unset ^= arr[i];
    }
}
```

print (set + " " + unset);

Ques

Given N array elements, choose two indices(i, j) such that **(i != j)** and **(arr[i] & arr[j])** is maximum.

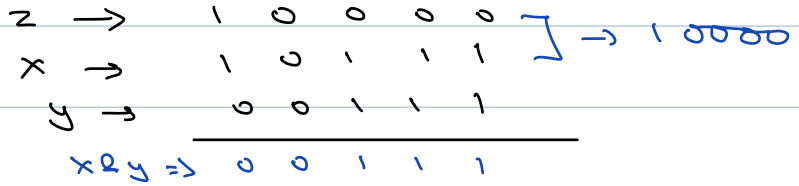


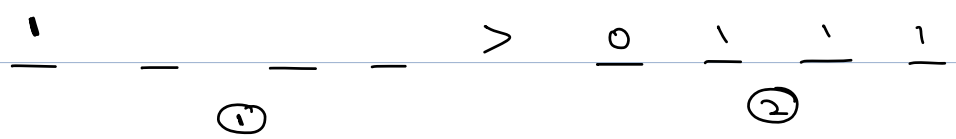
ex) arr[5] = {21, 18, 24, 17, 16}
↳ (21, 17)

idea → brute force → calculate AND of all pairs.

T.C → O(n²)
S.C → O(1).

idea having more no. of set bits at same place. ✗





arr[] = {26, 13, 28, 27, 7, 25}

```
      4  3  2  1  0
26 -> 1  1  0  1  0
13 -> 0  0  0  0  1
28 -> 1  1  1  0  0
27 -> 1  1  0  1  1
 7 -> 0  0  0  0  1
25 -> 1  1  0  0  1
-------------------
      1  1  0  0  1   <- Max And value.
-------------------
```

```
ans = 0;
for (i = 30; i >= 0; i--) {
    // find count of set bits at this idx
    int c = 0;
    for (j = 0; j < n; j++) {
        if (checkBit(arr[j], i)) {
            c++;
        }
    }
    if (c >= 2) {
        // we can form a pair, whose and will have a 1 at this place
        ans = ans + (2^i)   // 1 << i
        for (j = 0; j < n; j++) {
            if (checkBit(arr[j], i) == 0)
                arr[j] = 0;
        }
    }
}
return ans;
```

T.C → O(n)

S.C → O(1).

Ques Calculate count of pairs which give Xorm And.

→ after above. (n) → $\frac{n(n-1)}{2}$

we have, 4 non zero elements,

- - - - ↓

$\frac{4 \times (4-1)}{2}$

a, b, c, d

(a,b) (a c) (a d) (b c) (b d) (c d)