Bit Manipulation - 2

Question 1 :- We are given an integer array where every Number occurs twice except for one number which occurs just once. Find that number.

Ex1 :- [4, 5, 5, 4, 1, 6, 6]
↳ Ans

Ex2 :- [7, 5, 5, 1, 7, 6, 1, 6, 4]
↳ Ans

BRUTE FORCE

↳ Traverse over array & check if element is repeated or Not.

TC - O(N²)
SC - O(1)

↳ OPTIMIZATION

IDEA 1 :- HASHMAP

↳ Traverse the array & store count corresponding to each element.

TC - O(N)
SC - O(N)

IDEA2 :- Based on two properties.

① $A \wedge A = 0$

② $a \wedge b \wedge c = a \wedge c \wedge b$ [Cummulative property]

Quiz! :- Value of $120 \wedge 5 \wedge 6 \wedge 6 \wedge 120 \wedge 5$

$120 \wedge 120 \wedge 5 \wedge 5 \wedge 6 \wedge 6$

$0 \wedge 0 \wedge 0 \Rightarrow 0$ Ans

Observation :- Since (^) help to cancel out similar elements. So we can use it to find XOR of entire array to get unique element.

PSEUDO CODE

```
int x = 0;
for (i = 0; i < arr.length; i++){
    x = x ^ arr[i];
}
return x;
```

TC	→ O(N)
SC	→ O(1)

IDEA 3 :- Here, we will work on Bit level.

Ex :- A = [2 , 3 , 5 , 6 , 3 , 6 , 2]

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
		3	6	3

→ unique.

← Count of Set Bits.

Observation :- ① If Single element is Not there, the Count of Set Bits will be Multiple of 2

② If Count of Set Bits is odd, it depicts that their is one extra Set Bit.

③ So, Let analys Count of Set Bits.

⇒ 3 6 3

↓ ↓ ↓

1 0 1 ← Ans

PSEUDO CODE

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

    if (cnt %2 != 0){
        ans += 2^i;
          ↳ (1<<i)
    }
}
```

TC → O(32 N) ≈ O(N)

SC → O(1)

Question :- Given an integer array, all the elements will occur thrice except one element which occur once. Find the unique element.

Eg :- [4, 5, 5, 4, 1, 6, 6, 4, 5, 6]
↳ One

BRUTE FORCE

↳ Traverse over array & check if element is repeated or Not.

TC - O(N²)
SC - O(1)

↳ OPTIMIZATION

IDEA 1 :- HASHMAP

↳ Traverse the array & store count corresponding to each element.

TC - O(N)
SC - O(N)

IDEA 2 :- Can we solve this using XOR?

↳ No, because A ^ A = 0

but, A ^ A ^ A != 0

IDEA 3:- Previous question's idea 3.

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

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
Count	4 (3+1)	6 (6+0)	6 (6+0)	10 (9+1)

If Count → 3x → (unique No bit is unset)

→ 3x+1 → (unique No. bit is set)

PSEUDO CODE

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

```
        if (cnt % 3 != 0) {
            ans += 2^i ;
                   ↳ (1<<i)
        }
    }
    return ans;
```

Final observation

① The remainder on %3 = 1 only if the bit is set in unique element.

Extention ↳ ② If question is Number appearing 2 time to be found when other are appearing 3 times then the remainder on %3 == 2.

TC - O(N)
SC - O(1)

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

Ex:- arr[] = { 4, 5, 4, 1, 6, 6, 5, 2 }

↳ Ans [1, 2]

Ex:- arr[] = { 4, 9, 9, 8 }

↳ Ans [4, 8]

OBSERVATION

IDEA :- when we see the first 2 repeated element, though is XOR

Ex:- [~~3~~ ~~7~~ 6 ~~7~~ 8 ~~3~~ ~~9~~]

↳ Ans = 6 ^ 8

↳ XOR might be helpful but not directly because Ans is XOR of two elements.

IDEA2 :- Can we somehow identify elements from its XOR.

Ex:-

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

↳ If we take XOR of entire Array, then the result will be XOR of unique elements because similar will be cancelled out.

11 → 0 1 0 1 1

17 → 1 0 0 0 1

1 1 0 [1] 0

Observation :- In the two unique nos. this Bit will be different.

⇒ Based on this observation

⇒ Divide entire array on the basis of 1st Bit level

Set

10, 6, 11, 10, 6

↳ XOR = 11

unset

8, 8, 9, 12, 9, 12, 17

↳ XOR = 17

⇒ Divide entire array on 3rd bit level

Set

10, 8, 8, 9, 12, 9, 11

10, 12

↳ XOR = 11

unset

6, 6, 17

↳ XOR = 17

PSEUDO CODE

① Step1 :- XOR of entire array

```
V=0
for (i=0; i < arr. length ; i++){
    V = V ^ arr[i];
}
```

② Step2 :- from 'V' get the First Set Bit level from right.

```
Pos = 0;
for (i=0; i < 31 ; i++){
    if (checkBit(V, i)){
        Pos = i;
        break;
    }
}
```

③ Step3 :- Split the entire array on the basis of 'Pos' bit level

```
set=0 , unset = 0
for ( i --> 0 to n-1 ){
    if ( checkBit(arr[i], Pos)){
        set = set ^ arr[i];
    }
    else {
        unset ^= arr[i];
    }
}
Print ( set + " " + unset );
```

Quiz2 :- Time Complexity of above Code?

TC → O(N)

SC → O(1)

Question4 :- Given N array elements, choose two indices (i, j) such that (i != j) & (arr[i] & arr[j]) is maximum.

Ex :- [5, 4, 6, 8, 5]

Indices.

Ans = [0, 4] ← Maximum

Ex :- [27, 18, 20]

(27, 18) → 18

(27, 20) → 16

(18, 20) → 16

Ans

Quiz3 :- Max & Pair in array :-

arr[5] = { 21, 18, 24, 17, 16 }

↳ 21 & 17

Quiz4 :- Max & Pair in array :-

arr[5] = { 5, 4, 3, 2, 1 }

↳ 5 & 4

Brute force

↳ Calculate AND of all pairs.

TC → O(N²)
SC → O(1)

↳ OPTIMIZATION

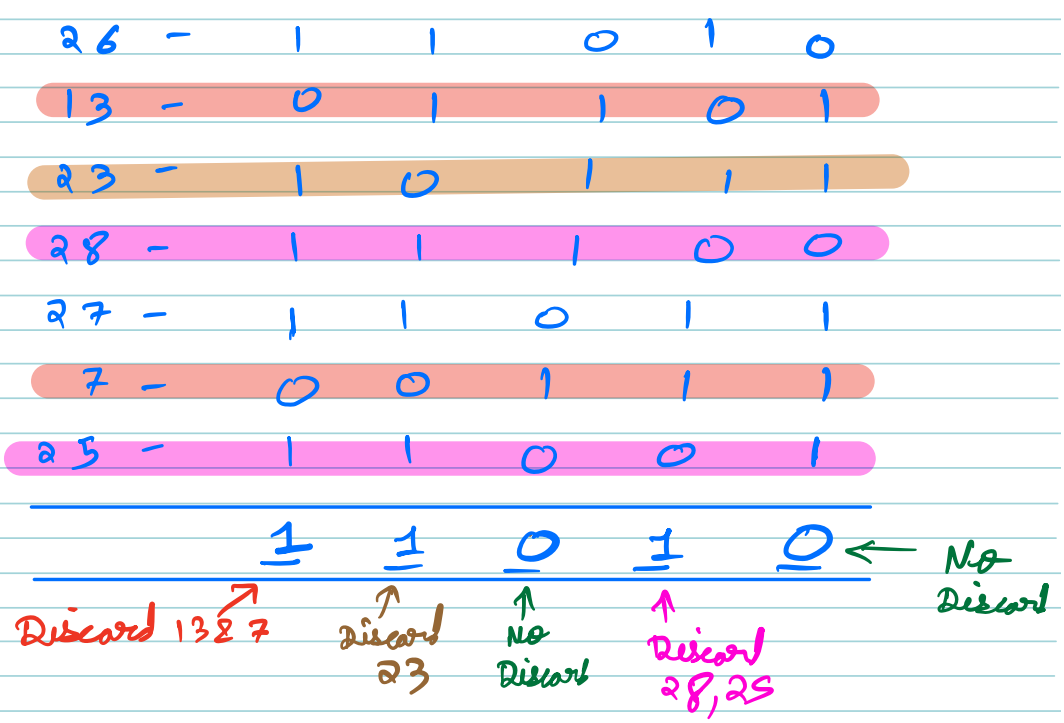
IDEA :- ① AND gives 1 when both bits are set

② Select Nos. which gives max set Bits starting from MSB.

(10000) ← { z → 1 0 0 0 0
x → 1 0 1 1 1
y → 0 0 1 1 1 } AND
0 0 1 1 1

EXAMPLES

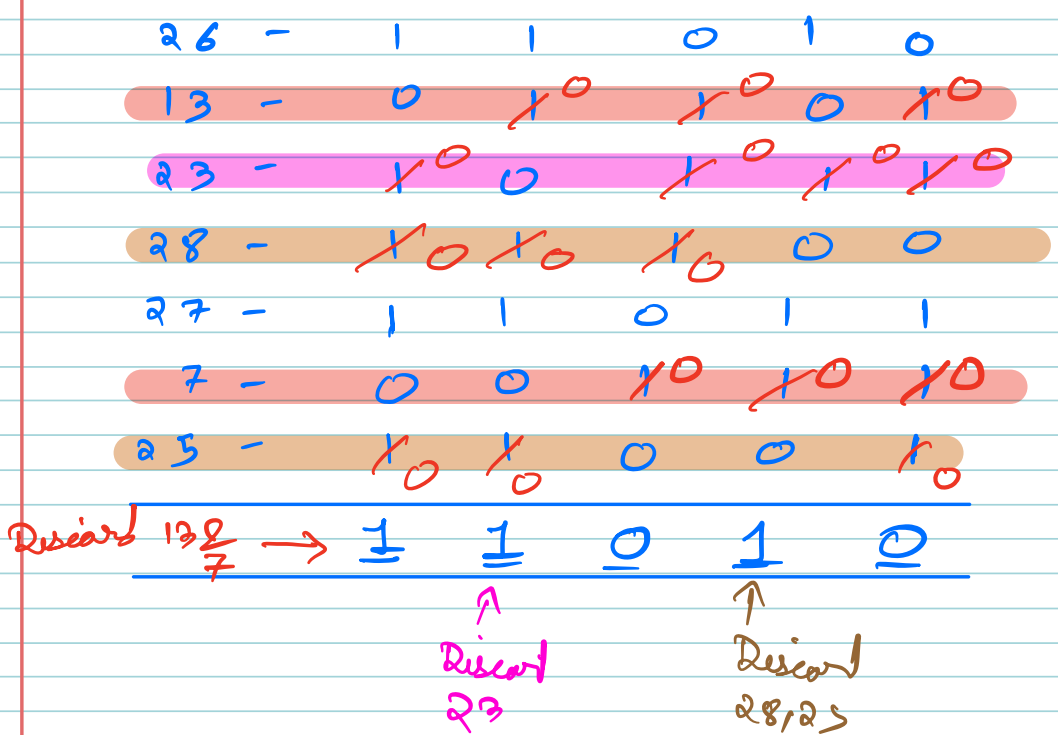
E :- {26, 13, 23, 28, 27, 7, 25}



Observations

1. If we are looking for max AND, we need to start from left (MSB)
2. If there are NO 2 ones then we need to keep 0 at that bit level. And will Discard Nothing.

Ex:- {26, 13, 23, 28, 27, 7, 25}



PSEUDO CODE

```
ans = 0
for ( i = 30 ; i >= 0 ; i--) {
    // find count of set Bits on ith level.
    cnt = 0
    for (j = 0; j < N ; j++) {
        if ( checkBit ( arr[j], i)) {
            cnt++
        }
    }
    if ( cnt >= 2) {
        // we can form a pair & have 1 on ith level.
        ans = ans + 2^i

        // Discard unwanted elements
        for (j = 0; j < N ; j++) {
            if ( checkBit (arr[j], i) == F) {
                arr[j] = 0;
            }
        }
    }
}
return ans;
```

TC → O(N)
SC → O(1)

[GOOGLE]

Question :- Calculate the count of pairs for which bitwise & is maximum.

Ex:- (a, b, c, d)

Pair → (a, b) (a, c) (a, d) (b, c) (b, d) (c, d)

Generic formula

Sum of N Natural Nos with No. of terms as 'N-1'

$$\Rightarrow \frac{n(n+1)}{2}$$

$$\Rightarrow \frac{(N-1)(N-1+1)}{2} \quad [n \to (N-1)]$$

$$\Rightarrow \frac{(N-1)N}{2}$$ Ans