Bit manipulations.

Bitwise opurators → &, |, ^, ~, <<, >>

some same puppy shame

A	B	A&B	A|B	A^B	~A	~B
0	0	0	0	0	1	1
0	1	0	1	1	1	0
1	0	0	1	1	0	1
1	1	1	1	0	0	0

A = 20
B = 45

A = 0 0 0 1 0 1 0 0
B = 0 0 1 0 1 1 0 1

Print (A & B) = 4 0 0 0 0 0 1 0 0 (4)

print (A | B) = 61 0 0 1 1 1 1 0 1 (61)

Print (A ^ B) = 57 0 0 1 1 1 0 0 1 (57)

Commutative

a & b = b & a
a | b = b | a
a ^ b = b ^ a

Associative

a & b & c = b & a & c = c & a & b
''
''

$a \& 0 = 0$	$a \mid 0 = a$	$a \wedge 0 = a$
$a \& a = a$	$a \mid a = a$	$a \wedge a = 0$

Right shift operator

$a = 50$ 0 0 1 1 0 0 1 0   50 $(2^1+2^4+2^5)$

$a >> 1$ 0 0 0 1 1 0 0 1   25 $(2^0+2^3+2^4)$

$a >> 2$ 0 0 0 0 1 1 0 0   12

$$\underset{}{50} \rightarrow \overset{50>>1}{\frac{50}{2^1}} \rightarrow \overset{50>>2}{\frac{50}{2^2}}$$

$$a >> n = \frac{a}{2^n}$$

Left shift operator

$a = 5$ 0 0 0 0 0 1 0 1 (2^0+2^2)

$a << 1$ 0 0 0 0 1 0 1 0   10 (2^1+2^3)

$20 = (2^2+2^4) =$

$$a << N = a \times 2^n$$

Q. Given a number N and j. Check if j^{th} bit is set or not.

$N = 53$

$$\begin{array}{ccccccc} 6 & 5 & 4 & 3 & 2 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 \end{array}$$

$j = 3$ ans = False

$j = 2$ ans = True

$$\begin{array}{ccccccccc} & & & & & & & & 0^{th} \\ & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1/0 \\ \& & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/0 \end{array}$$

$j = 0$ $N \& 1 == 1$

$j = 2^{th}$ $(N >> 2) \& 1 == 1$

```
if ((N>>j)&1 == 1) return True
else    return False
```

$N = N >> j$

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

Q. Given N elements. Every element repeats twices. Except one. Find unique element

arr[] = {3, 2, 3, 7, 2, 8, 7}

ans = 8

ans = XOR of all elements

$a \wedge a = 0$

$$ans = 3 \wedge 2 \wedge 3 \wedge 7 \wedge 2 \wedge 8 \wedge 7$$
$$= 3 \wedge 3 \wedge 2 \wedge 2 \wedge 7 \wedge 7 \wedge 8$$
$$= 0 \wedge 0 \wedge 0 \wedge 8$$
$$= 0 \wedge 0 \wedge 8$$
$$= 0 \wedge 8$$
$$= 8$$

```
int val = 0
for (i = 0; i < N; i++)
{
    val = val ^ arr[i]
return val
```

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

Q. Every element repeats thrice. Except one[unique]

Find unique element.

arr[] = {5, 7, 5, 4, 7, 11, 11, 9, 11, 7, 5, 4, 4}

ans = 9

```
BF : For each ele, go & count freq
        if freq == 1
            ans
```

TC : $O(N^2)$

SC : $O(1)$

```
Idea2 : Use freq hashmap
    → store freq of each element in hm
    → Iterate array again
        if freq == 1
            return ans
```

TC : $O(N)$

SC : $O(N)$

Idea 3.

{5, 7, 5, 4, 7, 11, 11, 9, 11, 7, 5, 4, 4}

5	=	0	1	0	1
7	=	0	1	1	1
5	=	0	1	0	1
4	=	0	1	0	0
7	=	0	1	1	1
11	=	1	0	1	1
11	=	1	0	1	1
9	=	1	0	0	1
11	=	1	0	1	1
7	=	0	1	1	1
5	=	0	1	0	1
4	=	0	1	0	0
4		0	1	0	0
		4	9	6	10

10 % 3 = 1

1 0 0 1

$2^3 + 2^0 = 9$

At each bit, calculate no of element having this bit set (K)

If (K % 3 == 1) this bit is set in ans

else this bit is unset in ans

```
int ans = 0

for ( i=0 ; i<32 ; i++ )
{
    // i^th bit

    cnt = 0
    for ( j=0; j<N; j++)
    {    if( checkbit( j, i ) == True)
                 cnt++

    if ( Cnt % 3 == 1)
    {    ans = ans + $2^i$
}

return ans
```

TC: O(32N) ≈ O(N)

SC: O(1)

Q. Every element repeats thrice. Except one[twice]

Find unique element.

cnt % 3 == 2

Q. Every element repeats 4 times except one[once]

Find unique element.

cnt%4 == 1

and

ans = XOR of all elements

Q. Given N elements. Every element repeats twice

Except 2 unique elements.

Find those 2 unique numbers.

arr[6] = 3, 4, 6, 4, 3, 8

ans = 6, 8

BF For every, check freq.

If freq == 1

return ans1,ans2

TC: O(N^2)
SC: O(1)

use hashmap

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

arr[6] = 3, 4, 6, 4, 3, 8

3^4^6^4^3^8

= 3^3^4^4^6^8

= 6^8

= 14

```
 0110
 1000
-----
 1110
```

xoR (A ^ B) = 14

3	2	1	0
1	1	1	0

diff diff

0100

3, 4, 6, 4, 3, 8

0011

2th bit is set

4, 6, 4

ans1 = xoR

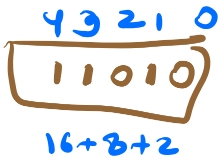
2th bit is unset

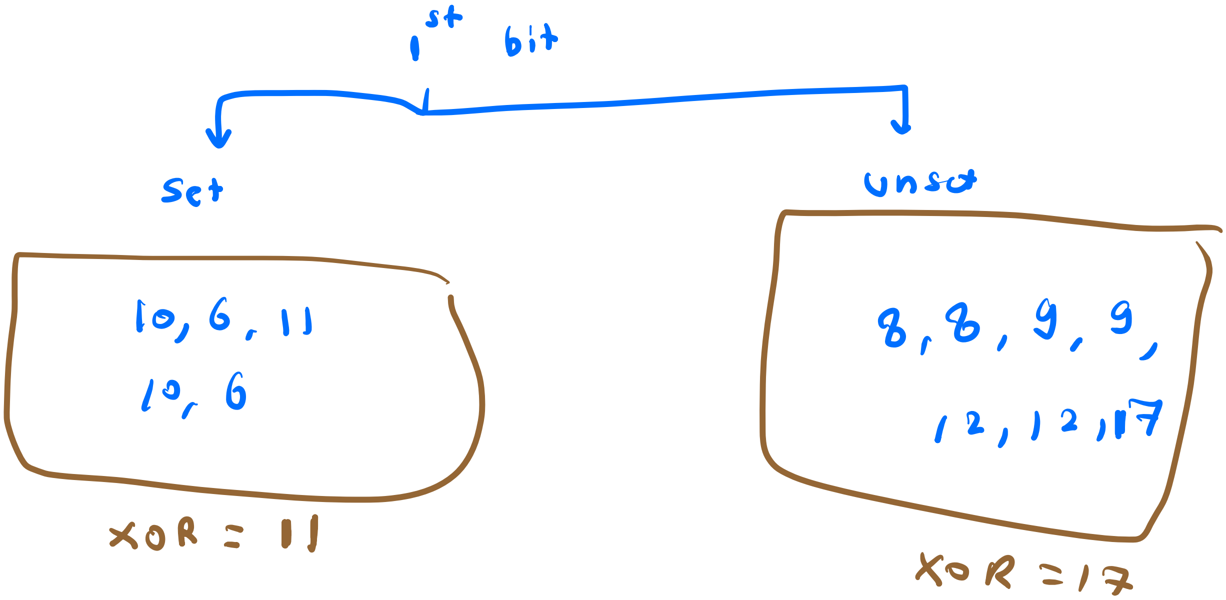
3, 3, 8

ans2 = xoR

10, 8, 8, 9, 12, 9, 6, 11, 10, 6, 12, 17

XOR (array) = 26

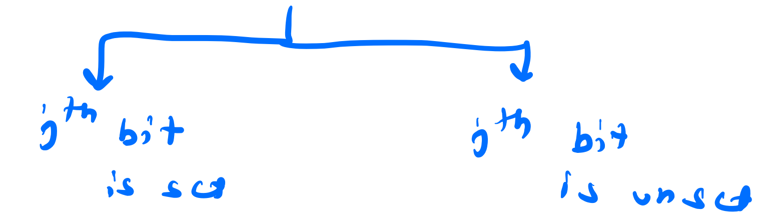




1. Calcule XOR of the array. = K

TODO 2. Take any set bit (K) = j^{th} bit

3. 2 Groups



j^{th} bit is set

j^{th} bit is unset

ans1 = XOR(G1)   ans2 = XOR(G2)

```
XOR1 = 0
XOR2 = 0

for ( i = 0 ; i < N ; i++ )
{
    if ( checkbit( Arr[i], j) == True)
            XOR1 = XOR1 ^ arr[i]
    else
            XOR2 = XOR2 ^ arr[i]
```

ans1 = xOR1

ans2 = xOR2

TC: $O(N)$

SC: $O(1)$

Q. Given N arr elements. Choose i, j

s.T arr[i] & arr[j] is maximised

Find max AND

arr[3] = {27, 18, 20}

27 & 18 = 10010 = 18

27 & 20 = 10000 = 16

18 & 20 = 10000 = 18

27 = 11011

18 = 10010

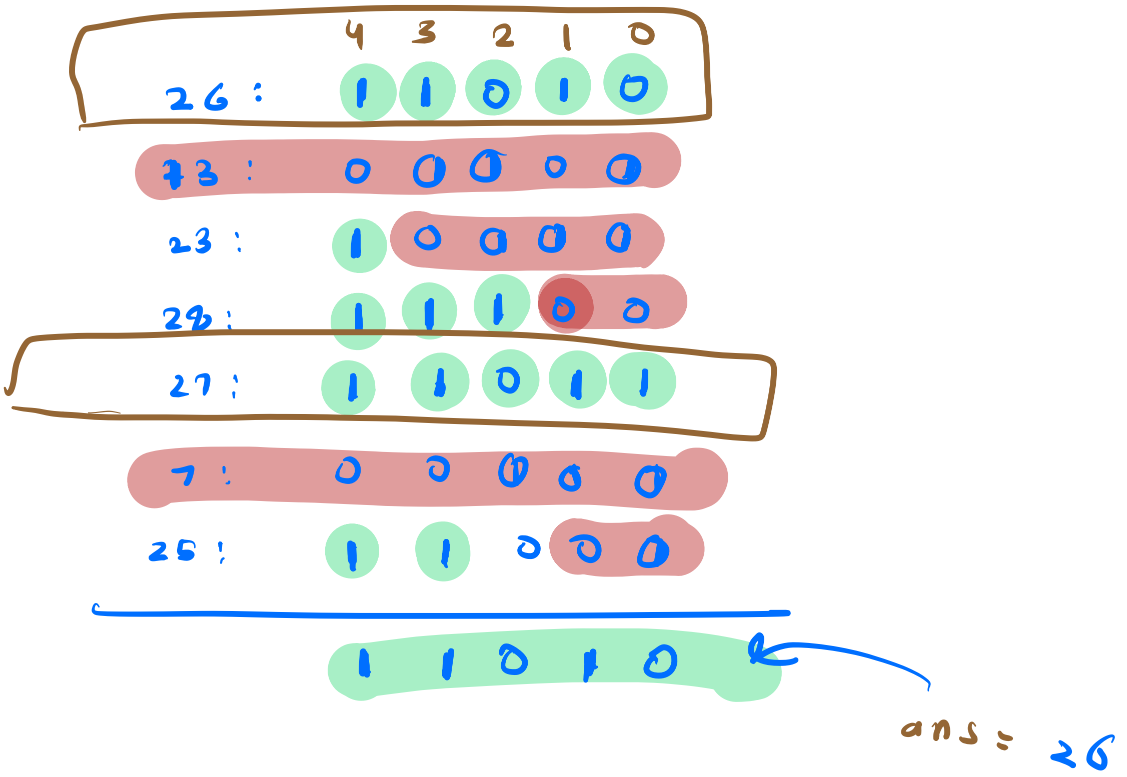
20 = 10100

BF: Consider each pair, pick max AND

TC: $O(N^2)$

SC: $O(1)$

arr[] = {26, 13, 23, 28, 27, 7, 25}



	4	3	2	1	0
26 :	1	1	0	1	0
13 :	0	0	0	0	0
23 :	1	0	0	0	0
28 :	1	1	1	0	0
27 :	1	1	0	1	1
7 :	0	0	0	0	0
25 :	1	1	0	0	0
	1	1	0	1	0

ans = 26

```
ans=0
for( i=31; i≥0; i--)
{
    // ith bit
    cnt=0
    for( j=0; j<N; j++)
    {   if( checkbit( a[j], i) == T)
                cnt++
    }

    if( cnt ≥ 2)
    {
        ans = ans + $2^i$
        for( j=0; j<N; j++)
        {   if checkbit( a[j], i ) == False)
            {   arr[j] = 0
            }
        }
    }
}
return ans
```

TC: O(32N) ~ O(N)

SC: O(1)

Break
10:40 – wednesday
9PM

——— x ——— x ———

→ Bitwise operators $\&, |, \wedge, \sim, <<, >>$

→ Properties $a \& a = a$
$a \& 0 = 0$

→ Checkbit(N, i) $(N >> i) \& 1 == 1$

→ All twice, except unique XOR (Array)

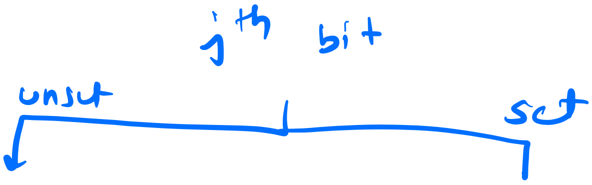
For each bit

→ All thrice except unique
{ cnt no of ele set
if(cnt % 3 == 1)
$ans = ans + 2^i$

→ All twice except 2 unique XOR(Array)
$= XOR(A \wedge B) = K$

↙

Any set bit from xoR
j^{th} bit

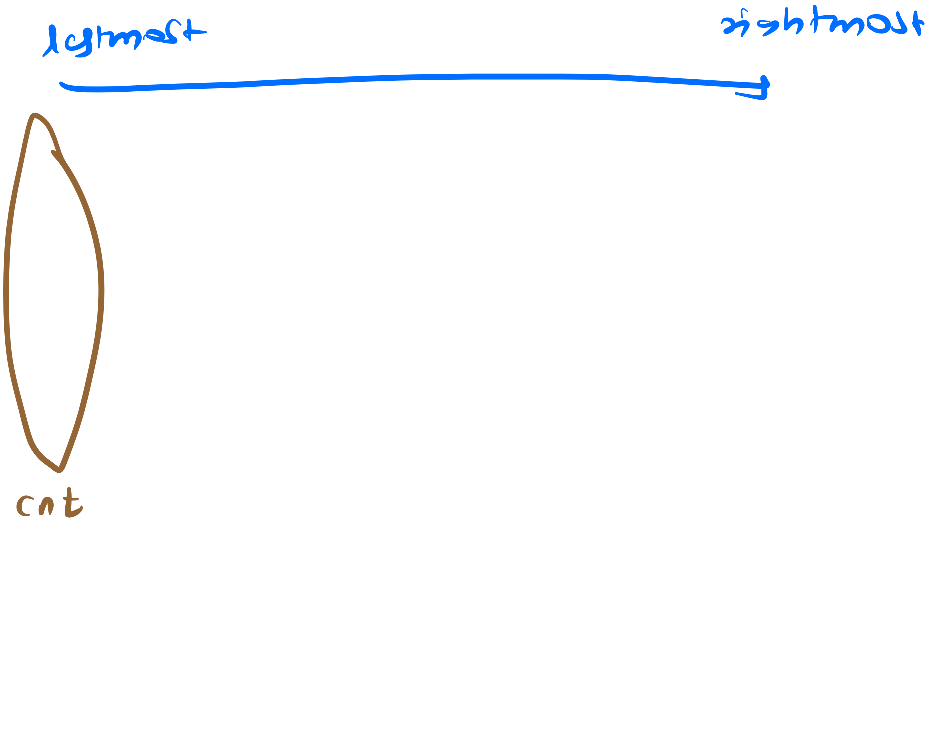


G_1	G_2
P_1	P_2
P_1	P_2
P_3	P_5
P_3	P_5
P_4
P_4

xoR (G_1) = ans1

xoR (G_2) = ans2

Q → Max AND of 2 number.



cnt ≥ 2

take bit in our ans

discard numbers with unset bit

ans