Day - 8

Sunday, 18 January 2026 11:08 AM

Q Power of LS.

A -> (45) 0 0 1 0 1 1 0 1

(1) 0 0 0 0 0 1 0 0 (1<<2)

0 0 1 0 1 1 0 1 => 45(A)

A => 0 0 1 0 1 1 0 1

(1) 0 0 0 1 0 0 0 0 (1<<4)

0 0 1 1 1 1 0 1 => 45+16 = (61)

N | (1<<i)

(N) => i^{th} bit is already set

N + (2^i) => i^{th} bit is unset.

A -> 0 0 1 0 1 1 0 1

(^) 0 0 1 0 0 0 0 0

0 0 0 0 1 1 0 1 => 45 - 32

=> (13)

A => 0 0 1 0 1 1 0 1

(^) 0 0 1 0 0 0 0 0

0 0 1 1 1 1 0 1 => 45 + 16

=> (61)

N ^ (1<<i)

N - 2^i ; if i^{th} bit is set

N + 2^i ; if i^{th} bit is unset.

A -> 0 0 1 0 1 1 0 1

(&) 0 0 1 0 0 0 0 0

0 0 1 0 0 0 0 0 => 32.

A => 0 0 1 0 1 1 0 1

(&) 0 0 1 0 0 0 0 0

0 0 0 0 0 0 0 0 => 0

N & (1<<i)

2^i ; if i^{th} bit is set

0 ; ——"—— unset.

Q Given an int 'N'

.. ... i^{th} bit of 'N'

(i) Set ...

$N = [N \mid (1 << i)]$

(ii) Toggle i^{th} bit (0 → 1, 1 → 0)

$N = N \wedge (1 << i)$

(iii) Check if i^{th} bit is set or not.

(i) $N \& (1 << i) \; != 0$ // i^{th} bit is set.

(ii) $[N \wedge (1 << i)] < N$ // ——— " ———

(iii) $N \mid (1 << i) == N$ // ——— " ———

Q. Check if i^{th} bit set or not w/o LS operator.

N = 45   0010 1101
            3210

i = 3 ⇒ True.

$N >> 3$

0010 1101
0001 0110
0000 1011
0 0000 1011 (N>>3)

if $[(N >> i) \& 1] \; != 0$
    // set
else
    // unset.

Q. Given an int 'N', count # of set bits

N = 45 ⇒ 0010 1101   o/p ⇒ 4

M-1   while (N > 0) {

```
if (N & 1) != 0 {
    count++
}
N = N >> 1
}
```

int ⇒ 4B
- 4.8 bit
- 32 bit

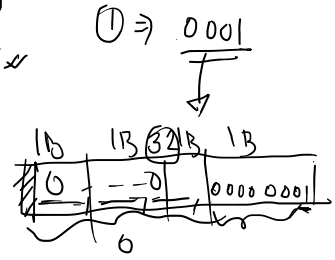
M-2

```
for i ← 0...31
    if (N & (1<<i)) != 0 {
        count++
    }
}
```

int { $-(2^{31})$, $+(2^{31}-1)$ }

0 | 0 1

① ⇒ 0001



Q Given an int arr of size 'N'
where every no. appears twice except 2 nos.
Find those 2 nos.

eg [2, 5, 6, 7, 5, 7] o/p ⇒ 2, 6

eg [1, 3, 3, 4, 1, 8] o/p ⇒ 4, 8

$1 \wedge 1 \wedge 3 \wedge 3 \wedge 4 \wedge 8$

$= 4 \wedge 8 = 0100\ (4)$

$= 1000\ (8)$

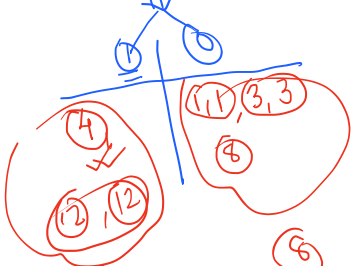
$1100\ (12)$ ⇒ TotalXOR

setIdx = ②

num1 = 0 $(0 \wedge 1 \wedge 3 \wedge 3 \wedge 8) = 0 \wedge 8 = 8$

num2 = 0 $(0 \wedge 4 \wedge 12 \wedge 12) = 0 \wedge 4 = 4$

2 1 0
1 - 0 0 0 1
1 - 0 0 0 1
3 - 0 0 1 1
3 - 0 0 1 1
4 - 0 1 0 0
8 - 1 0 0 0



12 - 1 1 0 0

12 1 1 0 0

④

```
totalXOR = 0
for  i ← 0 ... n-1
     totalXOR ^= arr[i]
```

O(n)

```
setIdx = 0
for i ← 0 ... 31 {
    if (totalXOR & (1 << i) != 0) {
        setIdx = i
        break;
    }
}
```

Find 1st set bit from RHS side

```
num1 = 0
num2 = 0
for  i ← 0 ... n-1 {
    if (arr[i] & (1 << setIdx)) == 0
        num1 ^= arr[i]
    else
        num2 ^= arr[i]
}
return (num1, num2)
```

O(n)

TC = O(n)

SC = O(1) ✓

Q Given an arr of size N where every no. appears thrice except '1' find the elem.

eg [1, 2, 1, 3, 4, 3, 1, 3, 2, 2] o/p ④

[illegible]

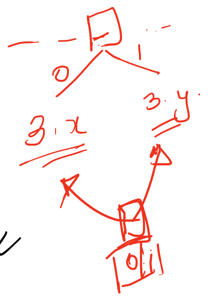
[2, 5, 7, 4, 7, 2, 5, 2, 7] O/P (4) ✓

2 ^ 2 ^ 2 ^ 5 ^ 5 ^ 5 ^ 4 ^ 7 ^ 7 ^ 7

(2^2^2 → 0, 5^5^5 → 0, 7^7^7 → 0)

⇒ 2 ^ 5 ^ 7 ^ 4

unique → 3*



	3	2	1	0
2 -	0	0	1	0
2 -	0	0	1	0
2 -	0	0	1	0
5 -	0	1	0	1
5 -	0	1	0	1
5 -	0	1	0	1
7 -	0	1	1	1
7 -	0	1	1	1
7 -	0	1	1	1
4 -	0	1	0	0

am = 0 1 0 0 ⇒ (4) ✓

```
am = 0
for i ← 0 ... 31
    c_0 = 0
    for j ← 0 ... n-1 {
        if (N & (1<<i) == 0) {
            c_0++;
        }
    }
    if (c_0 % 3 == 0) {
        am = [am | (1<<i)]
    }
```

```
c_1 = 0
for j ← 0 ... n-1 {
    if (N & (1<<i) != 0) {
        c_1++
    }
}
if (c_1 % 3 != 0) {
    am = [am | (1<<i)]
}
```