

Bit Manipulation



Real World

1st Key Takeaway

- ↳ Efficient & fast
- ↳ Hardware
- ↳ Compression Algorithms
- ↳ Encryption

Bitwise

2nd Key Takeaway
Left shift (<<)

num = 4

① Operators

$4 \gg 1$

$100 = 4$
 $010 = 2$

Division
by 2

Right shift (>>)

num = 4

$4 \ll 1$



multiplication
by 2

$0100 = 4$
 $1000 = 8$
 $10000 = 16$

num = 4
↓
Decimal

Binary

		1	0	0
		MSB		LSB
2	4			
2	2	0		
	1	0		

$1 \times 2^2 = 4$
 0×2^1
 0×2^0

Set \rightarrow '1'

3 Key Takeaway

<u>OR ()</u>			<u>AND (&)</u>			<u>XOR (^)</u>		
A	B	O/P	A	B	O/P	A	B	O/P
0	0	0	0	0	0	0	0	0
0	1	1	0	1	0	0	1	1
1	0	1	1	0	0	1	0	1
1	1	1	1	1	1	1	1	0

$\underline{A} \wedge \underline{A} = \underline{0}$
 $\underline{A} \wedge \underline{0} = \underline{A}$

$1 \wedge 0 = 1$
 $4 \wedge 0 = 4$
 $7 \wedge 0 = 7$

4th Key Takeaway

Interview Problems

①

0	1	2	3	4	5	6	7	8
2	4	2	4	5	6	6	5	4

$$2 \wedge 4 \wedge 2 \wedge 4 \wedge 5 \wedge 6 \wedge 6 \wedge 5 \wedge 4$$

$$0 \wedge 4 = 4$$

① Duplicate element

$$2 \wedge 2 = 0$$

②

2 ——— 2
4 ——— 3
5 ——— 2
6 ——— 2
9

odd
num of
times

$$5 \wedge 5 = 0$$

$$\text{Result} = 4$$

$$6 \wedge 6 = 0$$

XOR

$$4 \wedge 4 \wedge 4$$

$$0 \wedge 4 = 4$$

getOddFreqElem(nums)

for (int num: nums) {

XOR = XOR \wedge num;

}

return XOR;

time complexity $\rightarrow O(n)$

Space complexity $\rightarrow O(1)$

2

A handwritten diagram on a black background. On the left, the text "num = 16" is circled in white. An arrow points from this circle to the word "PowerOfTwo" which is underlined. Below "PowerOfTwo", the word "True" is written and underlined with two yellow lines. To the left of "True", the text "2^4" is written. A curved arrow points from "2^4" up to the "True" result.

And

num = 15 False

16 — 1 0 0 0 0

15 — 0 1 1 1 1

—————

0 0 0 0 0 \Rightarrow 0

$$(num \& num-1) == 0$$

num \rightarrow Power of Two

③ num = 15 1111

num = 4 → 100

o/p = 1

(1's)

count of the Result

Shift
Operators

number of set bits = 4

while (num > 0) {

count = 0

num = 15

1 1 1 1 ✓

0 0 0 1

① num & 1
count

② num >> 1

count + = 0 0 0 1

(/2)

0 1 1 1

0 0 0 1

0 0 0 1

0000

0001

0001

0001

0 0 1 1

0 0 0 1

0 0 0 1

↳