SCALER

Bit Manipulation - 2

TABLE OF CONTENTS

Amreshwar
Anil
Arunava Basak
Bhavesh Pandey
Deepshikha Arora
Dhasthagiri Reddy
Dipika Malik
Harikrishnan A
jeevanantham
M S Haseeb Khan
Mahesh Baswaraj
Manohar A N
MOHAMMAD ALI
Pallavi V Rao
Pranjul Kesharwani
Shivam Shiv
Shivanand Patil
Srikant Kumar Pratihary
Surabhi Kumari
Suresh
Uddeepta Saikia

N & (N-1) == 0

2^0

n = 8 → 1 0 0 0

n-1 = 7 → 0 1 1 1

0 0 0 0

if (n & (n-1) == 0)

n is a power of 2

3 → 11

2 → 10

10

4 → 100

3 → 011

000

TC of counting set bits of an integer N

Scenario:

In a large data center, servers communicate with each other and with the outside world by sending and receiving packets of data. Each packet has a unique identifier (ID), and due to the nature of transmission protocols, every packet is supposed to be sent and then acknowledged (ACK) by the receiving end. In an ideal scenario, for every packet ID, there should be exactly two occurrences in the network log: one for the packet being sent and another for its acknowledgment.

Problem:

Due to network issues, a packet got lost. Now we need to find out which packet got lost.

You are given an array with packet IDs from sender and receiver ends, only one packet was lost and hence it occurs only once in the array. Find this element.

S A S A S A S

1 1 2 2 5 5 6

< *Question* > : Given arr[N] where every number is present two times except one unique number. Find that unique number.

XOR arr[ ] → [4 , 5 , 5 , 1 , 6 , 4 , 6]

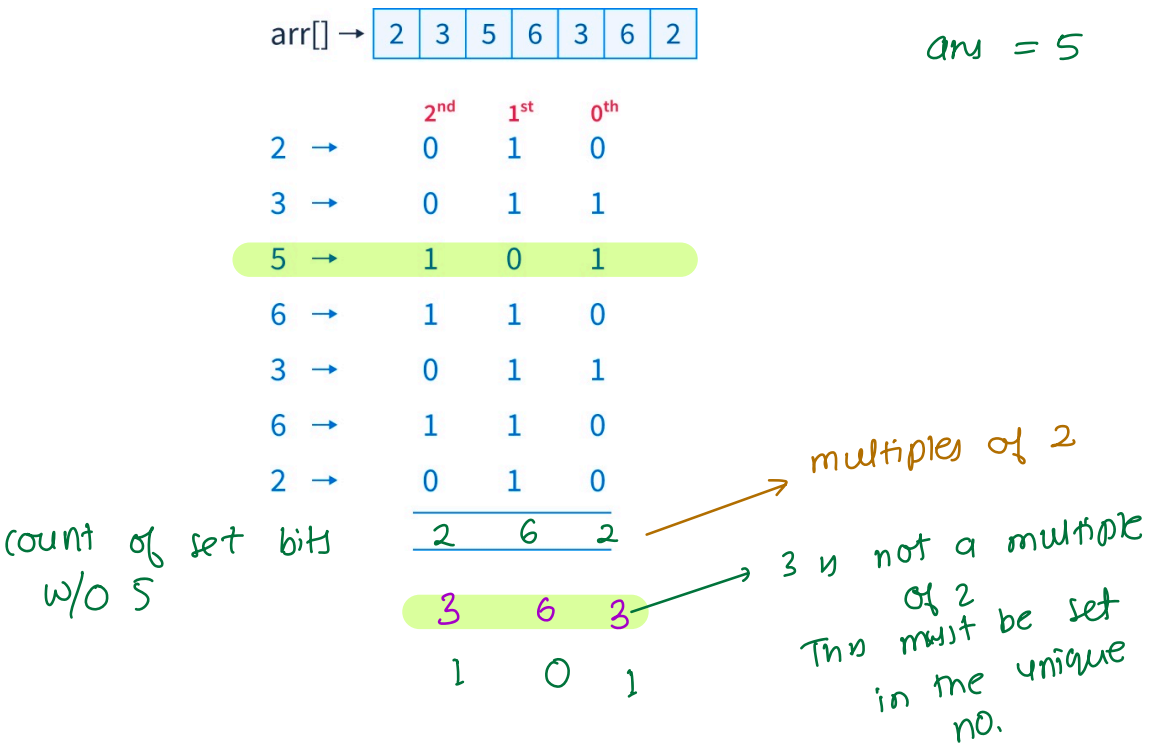
```
for i ⟶ 0 to N-1 {
    xor ^ = A[i]
}

print (xor)
```

TC : O(N)

SC : O(1)

[Another Approach]



< / > Code

```
unique = 0

for bit ⟶ 0 to 31 {
    cnt = 0
    for i ⟶ 0 to N-1 {
        // if bit is set in A[i]
        if ((A[i] & (1 << bit)) > 0) {
            cnt += 1
        }
    }

    if (cnt % 2 != 0) {
        unique = unique | (1 << bit)
    }
}

print(unique)
```

log(max_int)

TC : O(32 * N)

SC : O(1)

< *Question* > : Given an integer array of size N, where all the elements occur thrice except one element. Find that unique element.

$(1 \le N \le 10^6)$

arr[ ] → [4 , 5 , 5 , 4 , 11 , 6 , 6 , 4 , 5 , 6] ans = 11

0 1 2 3 4 5 6 7 8 9

BF Idea

For every val ⟶ Find the count of val in A
if count == 1 ⟶ return val

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

Idea 0 ⟶ use hash map
maintain freq map and check if
freq == 1

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

Idea -2

Xor all the values

~~1~~ ~~1~~ 1 ~~2~~ ~~2~~ 2 3

ans = 1 ^ 2 ^ 3

= 3 ^ 3 = 0

ans = 9

Idea -4

arr[ ] → [5 7 5 4 7 11 11 9 11 7 5 4 4]

0 1 2 3 4 5 6 7 8 9 10 11 12

3 2 1 0

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 % 3

1 0 0 1

multiple of 3

< / > Code

```
unique = 0

for bit ⟶ 0 to 31 {
    cnt = 0
    for i ⟶ 0 to N-1 {
        // 9t bit is set in A[i]
        if ((A[i] & (1<<bit)) >0 ) {
            cnt += 1
        }
    }

    if (cnt % 3 != 0) {
        unique = unique | (1<<bit)
    }
}

Print(unique)
```

log(max_int)

TC: O(32 * N)

SC: O(1)

Repeats

2 ⟶ XOR all values

3 ⟶ % 3 != 0

5 ⟶ % 5 != 0

4 ⟶ XOR all values

< *Question* > : Given an integer array of size N, where all elements repeat twice except two. Find those two elements.

{ T.C - 0(N) S.C - 0(1) }

arr[ ] → [4 , 5 , 4 , 1 , 6 , 6 , 5 , 2]

any 1 , 2

Bruteforce ⟶ for every val ⟶ Find the count of val in A

if count == 1

print (val)

TC: $O(N^2)$

SC: O(1)

Idea 2 ⟶ we HashMap to store freq

if (freq == 1) print (val)

TC: O(N)

SC: O(N)

arr[ ] → [4 , 5 , 4 , 1 , 6 , 6 , 5 , 2]

Xor all values ⟶ 1 ^ 2

```
  0 1
^ 1 0
-----
  1 1
-----
```

when is xor for a bit == 1

when both the bits are diff.

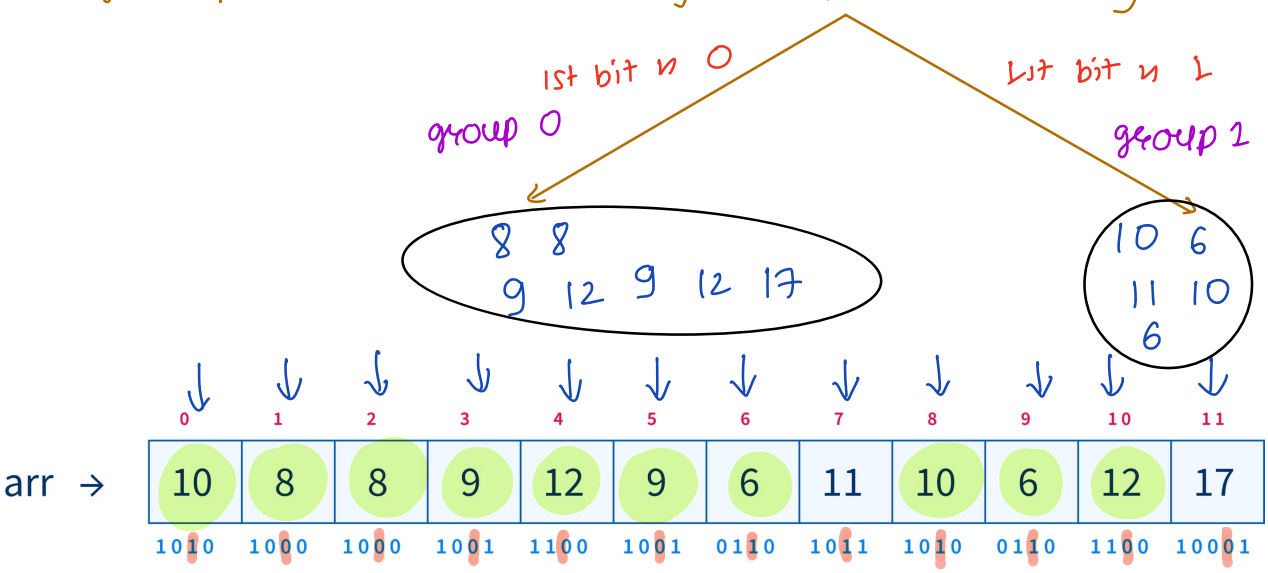
XOR of all the elements → 11 ^ 17 = 26

1st, 3rd, 4th bits are set

```
  01011
^ 10001
-------
  11010
-------
  43210
```

⇒ The two unique no. differ at 1st, 3rd, 4th bit

Let pick 1st bit and try to split the array



XOR the individual group to get unique no 0

XOR the individual group to get unique no 1

TC : O(N)

Step- 1 : Take XOR of all the elements.

```
val = 0
for i ⟶ 0 to N-1 {
    val ^= A[i]
}
```

TC : O(N)

Step- 2 : Find any set bit position in val.

```
pos = 0
for i ⟶ 0 to 31 {
    if ((val & (1<<i)) > 0) {
        pos = i
        break
    }
}
```

TC: O(32)

Step- 3 : Split the array on the basis of posth bit.

group0 = 0 , group1 = 0

```
for i ⟶ 0 to N-1 {
    if ( A[i] & (1 << pos) > 0 ) {
        group1 ^= A[i]
    } else {
        group0 ^= A[i]
    }
}
```

TC: O(N)

Step- 4 : Print the unique numbers.

```
print ( group0 , group1)
}
```

TC: O(1)

Break 22 : 37

Maximum and Pair

< *Question* > : Given N +ve array elements. Find the maximum value of (arr[i] & arr[j]) where i ! = j (indices must be different for 2 number)

arr[ ] → [27 18 20]

0 1 2

ans = 18

```
27 → 1 1 0 1 1
                &
18 → 1 0 0 1 0
---------------------------------
        1 0 0 1 0
```

```
27 → 1 1 0 1 1
                &
20 → 1 0 1 0 0
---------------------------------
        1 0 0 0 0
```

```
18 → 1 0 0 1 0
                &
20 → 1 0 1 0 0
---------------------------------
        1 0 0 0 0
```

BF Idea

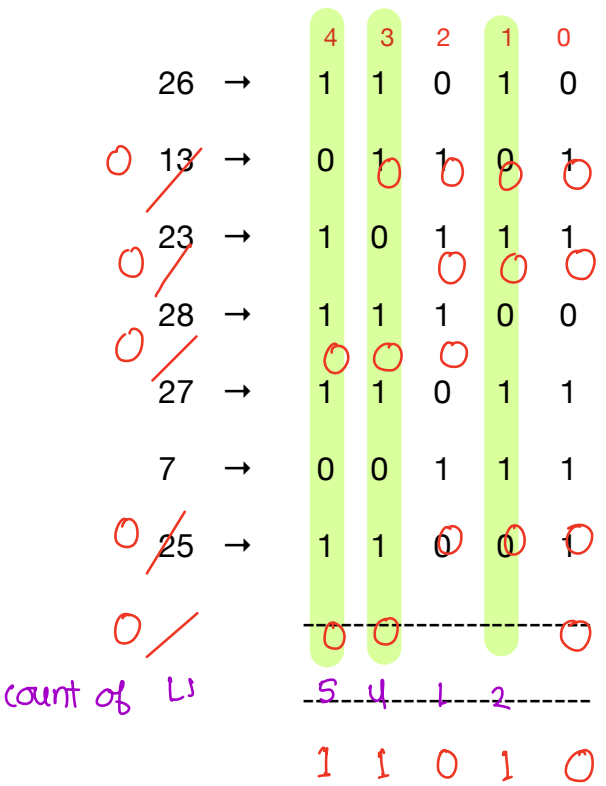
TC: $O(N^2)$

SC: $O(1)$

```
ans = 0
for i ⟶ 0 to N-1 {
    for j ⟶ 0 to N-1 {
        if (i != j) {
            ans = max(ans, A[i] & A[j])
        }
    }
}
print(ans)
```

arr[ ] → [26 13 23 28 27 7 25]

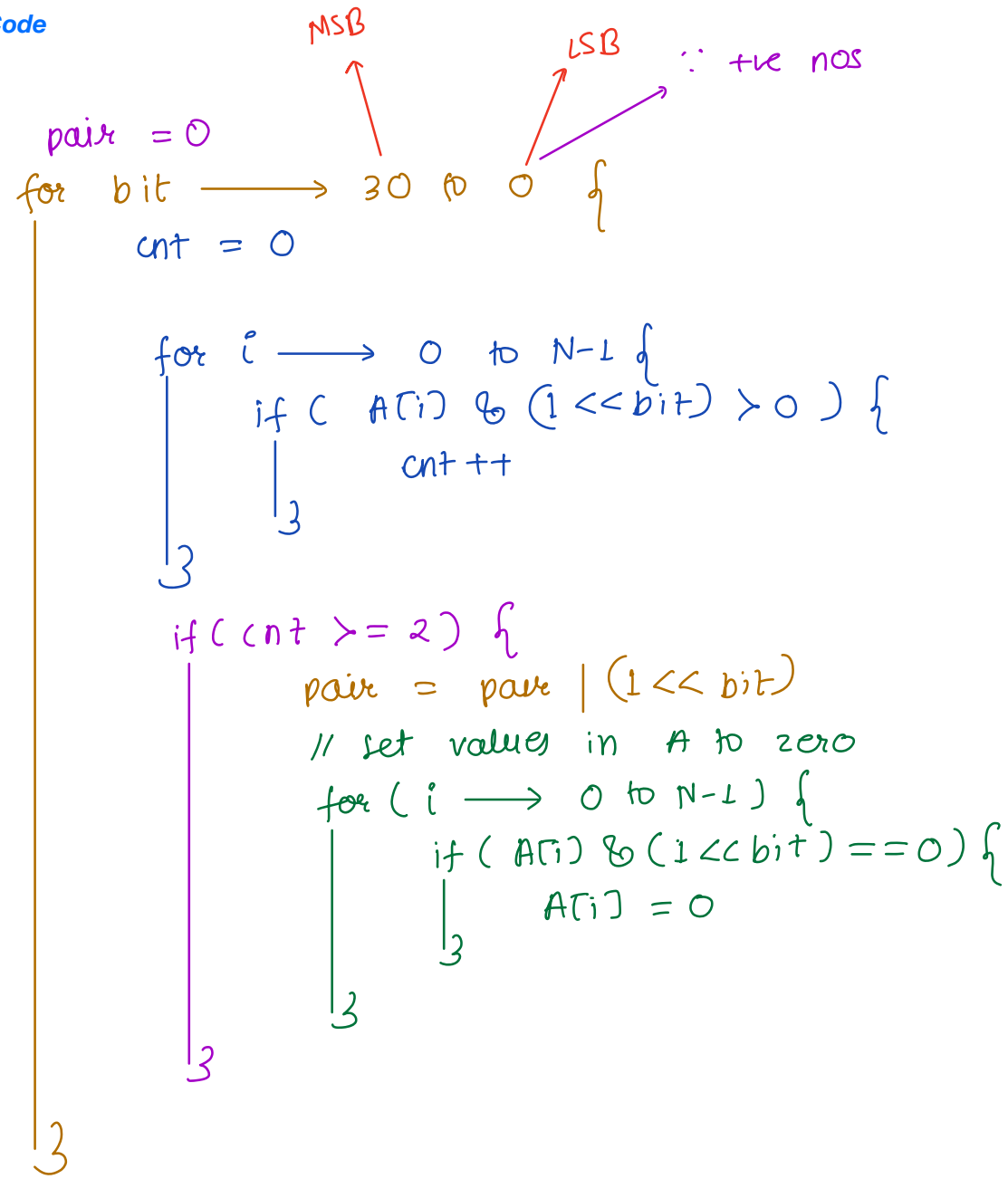
0 1 2 3 4 5 6



26 & 27

11010

< / > Code



TC : $O(31 * N) = O(N)$

$O(\log(max\_int) * N)$

Google Interview Question

Given N +ve nos. Count total no. of pairs whose bitwise & is maximum. pair $A_i \& A_j$ $(i != j)$

ans = 3

A = 26 26 27

pairs → 26 & 26 = 26
26 & 27 = 26
26 & 27 = 26 } 3

A = 9 10 12

9 & 10

```
 1001
&1010
 1000
```

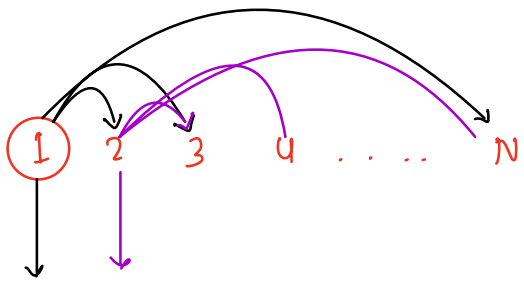
10 & 12

```
 1010
&1100
 1000
```

9 & 12

```
 1001
 1100
 1000
```

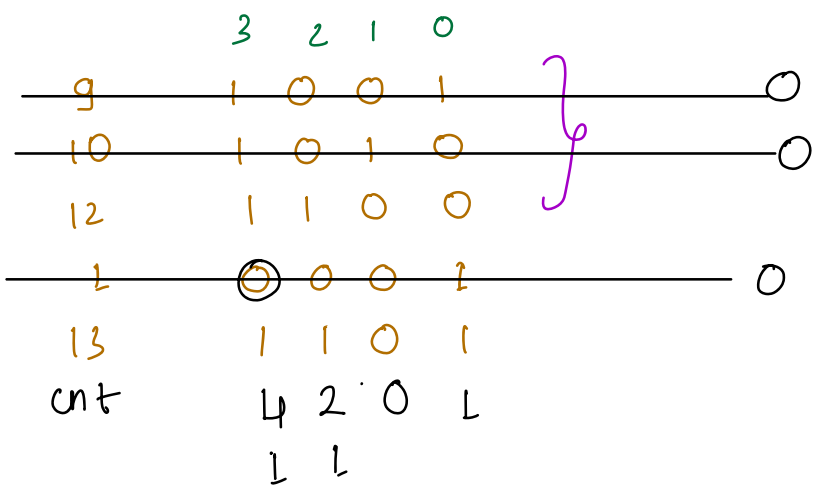
	3	2	1	0	
9	1	0	0	1	} 3
10	1	0	1	0	
12	1	1	0	0	
cnt	3	1	1	1	
	1	0	0	0	



n-1 n-2 . . . . 1 0

$\frac{(n-1) * (n-1+1)}{2}$ $\quad {}^{n}C_{2}$

$\frac{n(n-1)}{2}$



X = cnt of non zero values in A after above algo

final ans = $\frac{x(x-1)}{2}$