Today's Content

- Find 2 unique elements
- Find 2 missing elements.
- Calculate sum of xor of all pairs.
- Find max A&B

Q1) Given N elements, every element repeats twice, except 2 unique elements. Find 2 unique elements.

arr[i] ≥ 0

Ex1 arr[6] = {3, 6, 4, 4, 3, 8} := 6, 8

Ex2 arr[4] = {4, 9, 9, 8} : ⇒ 4, 8

Approach 1 : Hashmap : TC : O(n)
SC : O(n)

Approach 2 : Sorting : TC : O(n log n)
SC : O(1)

Approach 3 : {3, 6, 4, 4, 3, 8}

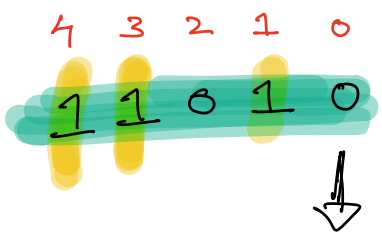
Xor everything ⇒ 6 ^ 8

arr[] = 10 (1010), 8 (1000), 8 (1000), 9 (1001), 12 (1100), 9 (1001), 10 (1010), 17 (10001), 11 (1011), 12 (1100)

E_1 = _ _ _ 1 _

E_2 = _ _ _ 0 _

← after xor of every element

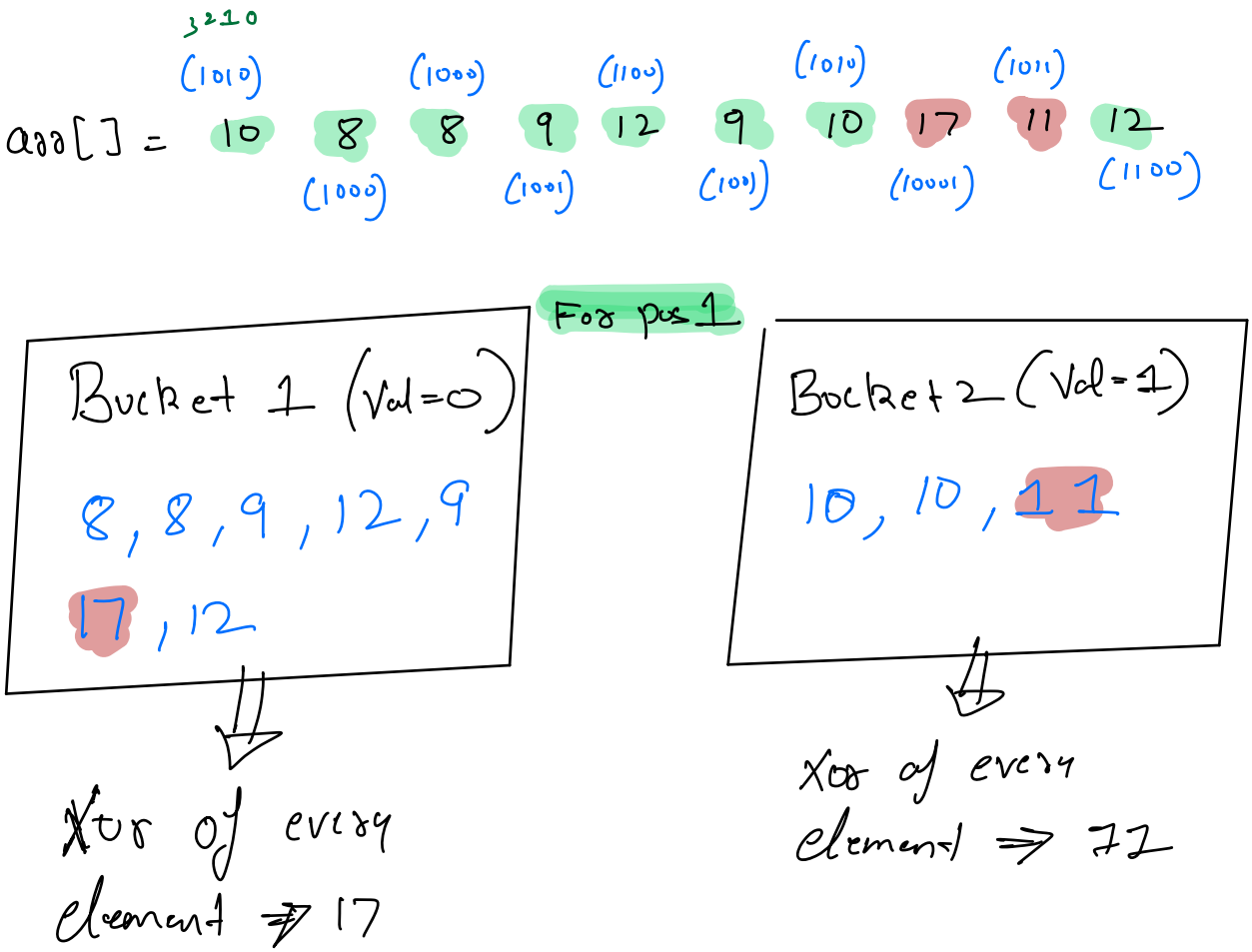


At Bit Pos 1, both unique elements have different bits.

Lets segregate all the array elements int 2 Buckets

Bucket 1 ⇒ Element which has 0 as bit in pos 1.

Bucket 2 ⇒ Element which has 1 as bit in pos 1


3210
(1010)
(1000)
(1100)
(1010)
(1011)
arr[] = 10 8 8 9 12 9 10 17 11 12
(1000)
(1001)
(101)
(10001)
(1100)
For pos 1
Bucket 1 (Val=0)
8, 8, 9, 12, 9
17, 12
Xor of every element ⇒ 17
Bucket 2 (Val=1)
10, 10, 11
Xor of every element ⇒ 11

Pseudo Code

1) xor of all elements

TC: O(n)

SC: O(1)

```
int xor = 0
for (int i = 0; i<n; i++) {
        xor ^= arr[i];
}
```

2) Find a set bit position.

TC: O(K) → no of bit positions to check.

SC: O(1)

```
int pos = -1;
for (int i = 0; i <= 30; i++) {
        if ( checkBit ( xor, i)) {
                pos = i;
                break;
        }
}
```

3) set → 0 // bit val = 1 (Bucket 2)

unset → 0 // bit val → 0 (Bucket 1)

```
for (int i=0 ; i<n ; i++ ) {          → 1
        if (checkBit (arr[i], pos)) {
                set ^= arr[i];
        } else {
                unset ^= arr[i];
        }
}
```

TC: O(n)

SC: O(1)

Return set, unset;

Q2) Given N Array Elements, array contains all elements from [1, N+2] except 2 elements. Find 2 missing elements.

Ex1 arr[4] = {3, 1, 4, 6} ⇒ (2, 5)

Ex2 arr[5] = {1, 6, 4, 7, 5} ⇒ (2, 3)

[1, N+2] [1, N+2]

⇓

missing 2 elements

{3, 1, 4, 6} + {1, 2, 3, 4, 5, 6}

[1, -, 3, 4, -, 6 | 1, 2, 3, 4, 5, 6}

Pseudo Code

1)
```
xor = 0
for (int i=0 ; i<n ; i++) {
    xor ^= arr[i];
}
```

```
for (int i=1 ; i<=(n+2) ; i++) {
    xor ^= i;
}
```

2) Find a set bit position.

```
int pos => -1;
for (int i= 0; i <= 30 ; i++) {
    if ( checkBit ( xor , i)) {
        pos => i;
        break;
    }
}
```

3)
```
set = 0 , unset = 0
for (int i=0 ; i<n ; i++) {
    if ( checkBit (arr[i], pos))
        set ^ = arr[i];
    else
        unset ^= arr[i];
}
```

```
for (int i = 1; i <= (n+2); i++) {
    if (checkBit (i, pos))
        set ^= i
    else
        unset ^ = i;
```

Q3) Given N elements, calculate sum of xor of all pairs.

Ex[5]: {3, 5, 6, 8, 2}

```
Sum = 0
for(int i=0; i<n; i++) {
    for (int j=0; j<n; j++) {
        sum += (arr[i] ^ arr[j]);
    }
}
```

3^3	3^5	3^6	3^8	3^2
5^3	5^5	5^6	5^8	5^2
6^3	6^5	6^6	6^8	6^2
8^3	8^5	8^6	8^8	8^2
2^3	2^5	2^6	2^8	2^2

⇒ TC: $O(n^2)$

Iterate on upper triangle. Find ans.

return 2*ans.

2: 0010 , 3: 0011 , 5: 0101 , 6: 0110 , 8: 1000

	3	2	1	0
3^5	0	1	1	0
3^6	0	1	0	1
3^8	1	0	1	1
3^2	0	0	0	1
5^6	0	0	1	1
5^8	1	1	0	1
5^2	0	1	1	1
6^8	1	1	1	0
6^2	0	1	0	0
8^2	1	0	1	0

6

x_1

x_2

1

$(1 \times 2^3) + (1 \times 2^2) + (1 \times 2^1) + (0 \times 2^0)$

$\Rightarrow 32 + 24 + 12 + 6 \Rightarrow 74$

Bit Pos 0

Set

3: 0011, 5: 0101

Unset

6: 0110, 8: 1000, 2: 0010

$2 \times 3 = 6$ ways $\times 2^0$

Bit Pos 1

Set

2: 0010, 3: 0011

6: 0110

Unset

8: 1000, 5: 0101

$2 \times 3 = 6$ ways $\times 2^1$

Bit Pos 3

Set

8: 1000

Unset

2: 0010, 3: 0011, 5: 0101, 6: 0110

$1 \times 4 \Rightarrow 4 \times 2^3$

Pseudo code.

```
Sum ⇒ 0

for (int i=0 ; i<=30 ; i++) {
    int count ⇒ 0
    for (int j=0 ; j<n ; j++) {
        if (checkBit(arr[j],i)) {
            count++;
        }
    }
}
int contribution ⇒ (count) x (n-count)
contribution ⇒ contribution x (1<<i);
Sum += contribution;
}
return (2*Sum);
```

TC: O(KN)

SC: O(1)

10:57

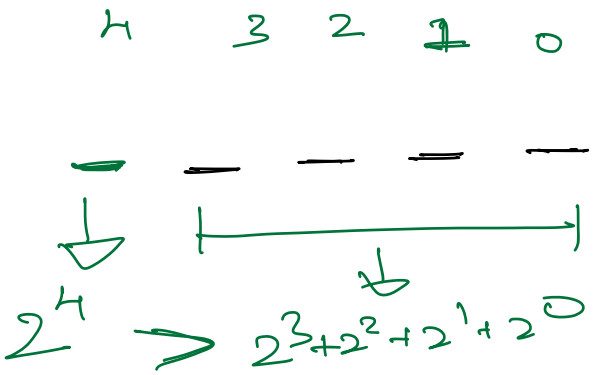
Q4) Given N Array element, choose 2 indices i,j such that $i \neq j$ && [Arr[i] & Arr[j]] is max.

Ex1 arr[3] = {27, 18, 20}

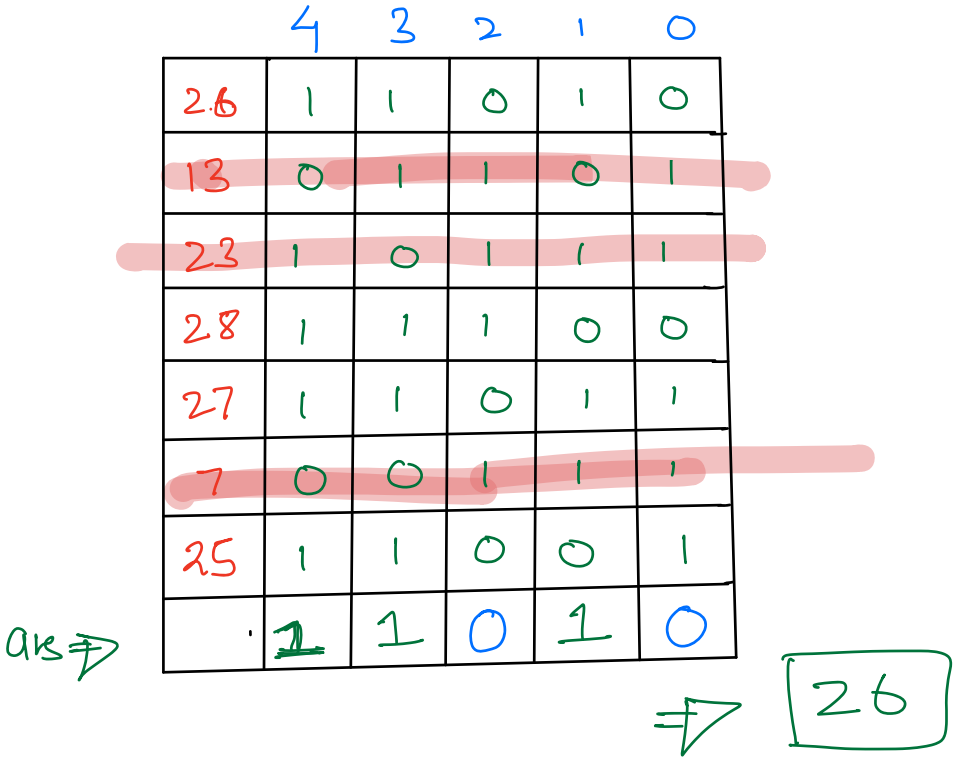
18: 10010
27: 11011
18: 10010

27: 11011
20: 10100
16: 10000

18: 10010
20: 10100
16: 10000



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



Pseudo code

```
ans = 0
for (int i = 30; i >= 0; i--) {
    int c = 0
    for (int j = 0; j < n; j++ {
        if (checkBit(arr[j], i))
            c++
    }
    if (c >= 2) {
        ans = ans + (1 << i);
        for (int j = 0; j < n; j++)) {
            if (!checkBit(arr[j], i))
                arr[j] = 0
        }
    }
}
return ans;
```

TC : O(KN)

SC : O(1)