Todays Content

1. Maximum AND Pair.
2. Maximum AND Pair, Count Pairs.

Recap:

Check i^{th} Bit set in N : $(N >> i) \& 1 == 1$: Set.

Set i^{th} Bit in N : $N = N | (1 << i)$

Flip i^{th} Bit in N : $N = N \wedge (1 << i)$

$a \wedge a = 0$

Q4: Given ar[N], Return max & between any pair ar[i] & ar[j]

$$\begin{array}{lccc} & 0 & 1 & 2 \\ \text{Ex: } ar[\,] = \{ & 27 & 18 & 20 \} \end{array}$$: ans =

$$\begin{array}{lccccc} & 0 & 1 & 2 & 3 & 4 \\ \text{Ex: } ar[\,] = \{ & 21 & 18 & 24 & 20 & 16 \} \end{array}$$ ans =

#Idea1:

#Idea2:

Obs: 1. a
& b

2. a : 0 0 1
& b : 0 1 0

0 0 0 0

2. ele1 ele2

2^4 2^3 2^2 2^1 2^0 2^4 2^3 2^2 2^1 2^0

_ _ _ _ _   _ _ _ _ _

	0	1	2	3	4	5	6	7	8
Ex: ar[9] =	{24	12	23	25	7	26	27	31	29}

	4	3	2	1	0
24:	1	1	0	0	0
12:	0	1	1	0	1
23:	1	0	1	1	1
25:	1	1	0	0	1
7:	0	0	1	1	1
26:	1	1	0	1	0
27:	1	1	0	1	1
31:	1	1	1	1	1
29:	1	1	0	1	1
cnt:					
Ans:					

Note:

Max pair

	4	3	2	1	0
a :					
b :					
and :					

	0	1	2	3	4	5	6	8
Ex: arr[ ] =	{10	14	1	6	11	10	8	6}

ans =

val:	3	2	1	0
10:	1	0	1	0
14:	1	1	1	0
1:	0	0	0	1
6:	0	1	1	0
11:	1	0	1	1
10:	1	0	1	0
8:	1	0	0	0
6:	0	1	1	0
cnt:				
&val:				

Max pair

y	3	2	1	0
a:				
b:				
and:				

```
int maxpair(int arr[], int N){ TC: O(      ) = O( ) SC: O()

}
```

Q4: Given ar[N], Return min ^ between any pair ar[i] ^ ar[j] i ≠ j

	0	1	2	3	4	
ar[] = {	7	4	6	9	10 }	ans = 1

Ex: 7^4 = 3 7^6 = 1

4^6 = 2

9^10 = 3

#Idea1: Generate all pairs, calculate xor & get overall min.

```
ans = INT_MAX;      TC: O(N^2)  SC: O(1)
i=0; i<N; i++) {
    j = i+1; j<N; j++) {
        ans = Min(ans, ar[i] ^ ar[j]);
    }
}
return ans;
```

#Idea2: Sort ar[] & calculate xor between adjacent elements & get overall min.

	0	1	2	3	4
ar[] = {	7	4	6	9	10 }

#sort

	i →	i →	i →	i →	i
	0	1	2	3	4
ar[] = {	4	6	7	9	10 }
xor	2	1	14	3	

if i == last_index stop

ans = 1.

```
int minXor(vector<int> &ar) {      TC: O(N log N + N) = O(N log N)
    sort(ar.begin(), ar.end());    Inbuilt sorting
    int ans = INT_MAX, N = ar.size();
    for(int i=0; i<N-1; i++) {
        ans = min(ans, ar[i] ^ ar[i+1]);
    }
    return ans;
}
```

Statement? Min Xor value will be among adjacent element in sorted arr[]

Why?

Say we have 2 ele: a b

Assume a < b

	-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
a :	0	1	1	0	0	1	....	
b :	0	1	1	0	1	0	....	

Say we have 3 ele: a b c

Assume a < b < c

	-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
a :	0	1	1	0	0	1	....	
b :	0	1	1	0	1	0	....	
c :	0	1	1	0	1	1		

In c 4th Bit = 1

a < b < c # a^c is never min

	-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
a :	0	1	1	0	0	1	....	
b :	0	1	1	0	1	0	....	
c :	0	1	1	1	0	1	....	

	-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
a^b :	0	0	0	0	1	1	...	
b^c :	0	0	0	1	1	1	...	
a^c :	0	0	0	1	0	0	....	

In c 2nd Bit = 1

a < b < c # a^c is never min

	-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
a :	0	1	1	0	0	1	....	
b :	0	1	1	0	1	0	....	
c :	0	1	1	0	1	1		

	-2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
a^b :	0	0	0	0	1	1	...	
b^c :	0	0	0	0	0	1	...	
a^c :	0	0	0	0	1	0	....	

#Con: In this way we can prove that min lies among adjacent elements.

i $\qquad\qquad$ i $\qquad\qquad$ i

ab : __ __ __ 0

~b : __ __ __ 1

c : __ __ __ 1

ab : __ __ __ 0

~b : __ __ __ 0

c : __ __ __ 1