{4,5,6} $\boxed{2^n}$

0 ← { 0 0 0 }
1 ← { 0 0 1 }
2 ← { 0 1 0 }
3 ← { 0 1 1 }
4 ← { 1 0 0 }
5 ← { 1 0 1 }
6 ← { 1 1 0 }
7 ← { 1 1 1 }

$O(2^n)$

```
for(i=0 ; i<2^n ; i++){        O(2^n)
  Arraylist{}
  j=0
  while (j<n){                 O(n)
    if(i&(1<<j) > 0) al.add(arr[j])
    j++;
  }
  Print (Arraylist)
}
```

{ 4, 5, 6 }
0 1 2

{ }
{ 4 }
{ 5 }
{ 4, 5 }

1000101
0001000
0000000

```
recu( ) {
  if(i == n){
    Pri—
  }
  //not pick
  recu( s, i+1)

  // s += a[i]
  recu(s, i+1)
}
```

(2^n)

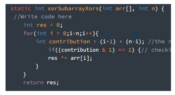
101

s = 1101

~~1^1^3^1^3^6^1^3^6^8^3^3^6^3^6^8^6^6^8^8~~

0 1 3 6 8
res=0 0 1 2 3

$3+3$



```
static int xorSubarrayXors(int arr[], int n) {
 //Write code here
    int res = 0;
    for(int i = 0;i<n;i++){
        int contribution = (i+1) * (n-i); //the r
            if((contribution & 1) == 1) {// checki
            res ^= arr[i];
        }
    }
    return res;
```

Game of XOR

[1, 2, 3] ⇒ 1 ^ 2 ^ 3

$O(n^2)$

[1 ^ 1^2 ^ 1^2^3 ^ 2 ^ 2^3 ^ 3]

$1 \wedge 3$

4 2 1
0 0 0
0 0 1
0 1 0
0 1 1

→ Count contribut

→ if Contri is odd xor it with result

0 1 2 3 4
1 2 3 4 5 5 − 1

count
(i+1) × (n−i)

3
34
345

5 − 2 ⇒ 3

O(1)

n → 63^N 32 64

0 − 31

$2^3 = 8$ ⇒ 1000 − 1

7 ⇒ 111 $\boxed{64}$ → $\boxed{32}$

$\boxed{2^{32}-1}$

$2^{31}-1$

30 25 31

1000 − 1
999

0 − 31 ⇒ $\boxed{32}$

0000 1010
^ 1111 1111
1 1111 0101

& 1 ^

Int_Max

$\boxed{2^{32}-1}$

1 ^ 1 ⇒ 0
0 ^ 1 ⇒ 1

1 ___ −1
31
1111111
31

1010
1100
0110

②

5 →
3 →

5 + 3

→ 8

1010 10 11 11 12 11 13 13 13

1010
1010
1010
1011
1011
~~1100~~
1011
1101
1101
1101
10 | 4 | 6 | 6

$\boxed{3} \to ①$

ans ⇒ $2^2 + 2^3$
⇒ 4+8

⑫ ⑬ ⑩

12 | 7 4 3 3

sum[i] % 3 != 0

1 1 0 1
1 0 1 0
1 1 0 0

40 $\boxed{32}$

0101
0011
0001
0110

Binary

MPMC

0000 0001
1111 1110
1111 1111

(−1)

1 0 0 0 0 0 0
1 0 1 1

㉛ 31 ◻ ← ◻

$\boxed{31}$

$\boxed{63}$ ← 1000000

⑥③ ← ㉛

1 001

13 = 1

1111110 111

2's −1

0000 0001
1111 1110
1111 1111

$\boxed{31}$ (−1)

32 10 1101
26

② ④ ⑥ ⑧

[D]

11 11 1 1111
11 11 1 11 0
00 — 01

x.y