Bit Manipulation

0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111
8	1000

Power 2

$0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$

=> 0 + 2 + 1 => 3

Set / Unset

10 11

$2^3 \cdot 0 + 2^2 \cdot 1 + 2^1 \cdot 1 + 2^0 \cdot 0$

10

0 − 1

+ − × / %

& | ^ << >>

12 & 10 = 8

1100
& 1010
1000

20 & 17

10100
& 10001
10001

21 & 14

5 % 2 = 4

5/2 => 2×2
= 4

1 | 1 = 1
1 | 0 = 1
0 | 1 = 1
0 | 0 = 0

20 | 17 →

10100
10001
10101

1010
1100
1110

1 & 1 = 1
1 & 0 = 0
0 & 1 = 0
0 & 0 = 0

1 ^ 1 = 0
0 ^ 0 = 0
1 ^ 0 = 1
0 ^ 1 = 1

20 & 14

10100
& 01101
00101

$((x >> 1) << 1)$

=> x

```
int countOne(int n){
    int cnt = 0;
    while(n!=0){
        n = (n & (n-1));
        cnt++;
    }
    return cnt;
}
```

[2 3 5 2 6 5 3]

O(n) O(1)

1) n^n = 0
2) n^0 = n
3) (a^b)^c = a^(b^c)

2 ^ 3 ^ 5 ^ 2 ^ 6 ^ 5 ^ 3
(2 ^ 2) ^ (3 ^ 3) ^ (5 ^ 5) ^ 6
0 ^ 0 ^ 0 ^ 6
0 ^ 6 = 6

O(n)
O(1)

[2 3 5 2 6 5 3 7]

a^b
6^7
111
110
001
a ^ b

101 => 5
001 = 1
110 => 6
110 => 6
100 => 4
011 ^ 3
001 ^ 1
101 ^ 5

k-th
{5 1 3 5}
3

k-unset
{6 6 4}
4

k = 0

4^3
100
011
111

0 << k

1) Do xor of all elements
2) Find the position of RMSB => K
3) Make 2 sets,
   4 => XOR of all elements having k^{th} bit set
   2 => " " " having k^{th} bit unset

mod 2

1100 => 2