1. Search in a row wise & col wise sorted matrix
2. Count of AND Pairs
3. Decreasing Dishes

① Discuss
② Revise notes

Contest 1 → Practice Mode

Score ≤ 60 → Reattempt

Reattempt 1 Oct 14 12 midnight
↓
Oct 15 11:59 pm

Given a matrix of integers A of $N \times M$ and an integer B. Every row and col is sorted in non-decreasing order, return position of B $(i * 1009 + j)$ in matrix. If not present, return -1.

$$A = \begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 1 & 2 & 3 \\ 2 & 4 & 5 & 6 \\ 3 & 7 & 8 & 9 \end{array} \qquad B = 2$$

$i \to 1$
$j \to 2$

$ans = (1 * 1009) + 2$
$= 1011$

$$A = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1 & 3 & 3 & 4 \\ 2 & 3 & 3 & 3 & 5 \end{array} \qquad B = 3$$

$i \to 1 \quad j \to 2$

$ans = 1011$

$\min \to (i * 1009 + j)$

$$A = \begin{array}{c|ccc} & 1 & 2 & 3 \\ \hline 1 & 1 & 2 & 3 \\ 2 & 4 & 5 & 6 \\ 3 & 6 & 8 & 9 \end{array} \qquad B = 7$$

$ans = -1$

$ans = \cancel{1012} \;\; 1011$

$$A = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1 & 3 & 3 & 4 \\ 2 & 3 & 3 & 3 & 5 \end{array} \qquad B = 3$$

2019

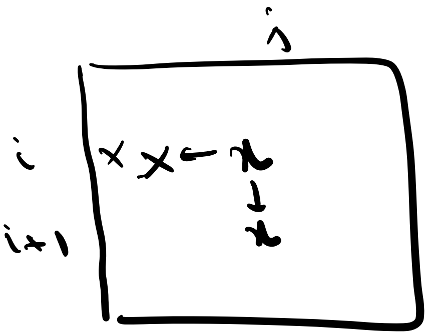
i, j	
1,3	$1 * 1009 + 3$
1,2	$1 * 1009 + 2$
	$\downarrow$ 1011

Minimize $i*1009+j$

Left (j--)
1, 3 → 1, 2

Down i++
1, 3 → 2, 3 $2*1009+3$

$\boxed{2021}$



// A[m][n]   // B

```
int i=0, j=n-1
int ans = INT_MAX

while (j>=0 && i<m) {
    if (A[i][j] == B) {
        ans = min(ans, ((i+1)*1009) + j+1)
        j--
    }
    else if (A[i][j] < B) {
        i++
    }
    else {   // A[i][j] > B
        j--
    }
}
if (ans == INT_MAX)
    return -1
return ans
```

TC : O(M+N)

SC : O(1)

Given an array of N integers, and Q queries in array B. For every query B[i], find count of pairs from array A such that bitwise AND of them has the $B[i]^{th}$ bit set.

$A = [2, 5, 6, 7]$

$B = [1, 2]$

$\;\;\;\;\;\;\;\;\;\;\downarrow\;\;\downarrow$

$\;\;\;\;\;\;\;\;\;\;3\;\;\;3$

4	2	5	6	7
3	1	2	0	

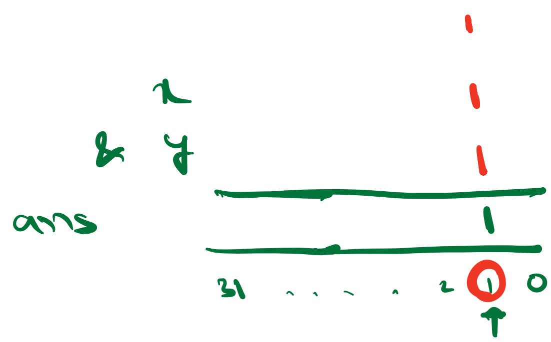
$1 \le N \le 10^5$

$1 \le Q \le 50$

$0 \le B[i] \le 31$

$1 \le A[i] \le 10^9$

	2	1	0
2	0	1	0
5	1	0	1
6	1	1	0
7	1	1	1



Nos. which has 1^{st} bit pos set $= 2, 6, 7$

2 6 7

$2 + 1 = 3$ pairs

$c = 3$

$ans = \frac{c-1 \times c}{2}$

$= \frac{2 \times 3}{2} = 3$

Nos. which has 2^{nd} bit pos set $= 5, 6, 7$

5 6 7

$2 + 1$

B → [2]   A = [x, y, z, a, b, d]

	..4	3	2	1	0
x			1		
y			0		
z			1		
a			1		
b			1		
d			0		

x   z   a   b

3 + 2 + 1

= 6 pairs

B[i] bit

1. Cnt nos. which have $B[i]^{th}$ bit set = C

2. Cnt of pairs = $\dfrac{(C-1)C}{2}$

sum of 1st N natural nos. = $\dfrac{N(N+1)}{2}$

3. Max bit pos = 32   0 ≤ bit pos ≤ 31

int cntpairs [32]

0 1 ... 31

```
int cntpairs [32]

// idx i -> ith bit pos -> store
cnt of pairs where & value has
ith bit set

for (int pos=0 ; pos <32 ; pos++) {
    int count = 0
    for (int i=0 ; i < N ; i++) {
        if ((A [i] & (1 <<pos)) != 0)
            count ++
    }
    cntpairs [pos] = ((count-1) * count) / 2
}

list <int> ans
for (int i=0 ; i < Q ; i++) {
    // query -> B [i]

    ans. insert ( cntpairs [B [i]])
}
return ans
```

32 N

Q

TC: O(N+Q)
SC: O(1)

32 size array

Given an array of N +ve integers representing weight of ingredients in dish. Find max possible sum of subarray with decreasing weights.

$1 \le N \le 10^5$

$0 \le A[i] \le 10^5$

A = [3, 2, 1] ans = 6

A = [3, 3, 5, 0, 1]
ans = 5

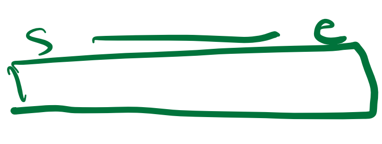
3, 3 X Not strictly decreasing

① Subarrays with decreasing elements
② +ve integers

BF →

```
ans = 0
for (s = 0 ; s < n ; s++)
    for (e = s ; e < n ; e++) {
        sum = a[s], flag = 1
        for (i = s+1 ; i <= e ; i++) {
            sum += a[i]
            if (a[i] >= a[i-1]) {
                flag = 0
                break
            }
        }
        if (flag == 1)
            ans = max(ans, sum)
    }
```

TC: O(N³) → N² CF



Optimized → O(N)

A =	[7,	6,	3,	3,	2,	4,	5,	11,	10]
sum	7	13	16	3	5	4	5	11	21
ans	7	13	16	16	16	16	16	16	21

```
int sum = A[0], ans = A[0]

for (int i=1 ; i<N ; i++) {
    if (A[i] ≥ A[i-1]) {
        sum = 0
    }
    sum += A[i]
    ans = max (ans, sum)
}

return ans
```

TC: O(N)

SC: O(1)

DOUBT | Recursion 1 Q5

```
int bar (int x, int y) {
    if (y==0) return 0
    return (x + bar (x, y - 1))
}

int foo (x, y) {
    if (y==0) return 1
    return bar (x, foo (x, y-1))
}
```

① $bar(x,y) = x + bar(x, y-1)$ — ①

$bar(x, y-1) = x + bar(x, y-2)$

② $bar(x,y) = 2x + bar(x, y-2)$ — ②

$bar(x, y-2) = x + bar(x, y-3)$

After k steps

$bar(x,y) = kx + bar(x, y-k)$

$y - k = 0$ $\Rightarrow k = y$

$\boxed{bar(x,y) = yx}$ After y steps

$$foo(g,n) \Rightarrow bar(g, foo(g, n-1))$$

$$\downarrow$$

① $\rightarrow$ $foo(g,n) = g \times \underline{foo(g, n-1)}$

$$foo(g, n-1) = g \times foo(g, n-2)$$

② $foo(g,n) = g^2 \times foo(g, n-2)$

$$foo(g,n) = g^k \times foo(g, n-k)$$

$$n-k = 0 \quad \Rightarrow \quad k = n$$

$$foo(g,n) = g^n$$

$$foo(3,5) = 3^5 = 243$$