Q ≡ given a matrix of N*M Q queries top left, bottom right

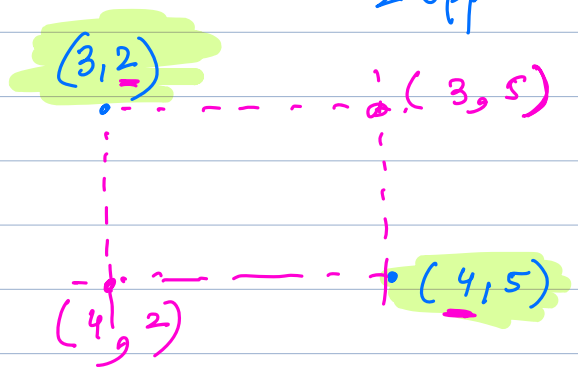
Find submatrix sum

↓ represent

⇓ 4 coordinates = 2 opposite corner ⇒ {Top left, Bottom right

	0	1	2	3	4
0	7	1	-6	3	12
1	10	5	-1	0	9
2	6	4	-3	8	11
3	13	-8	-5	12	4
4	3	2	1	9	8
5	4	3	-2	6	3

(2,1) (4,3)

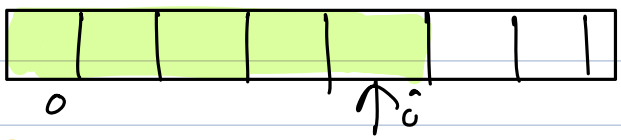


pf[i][j] = 0-i

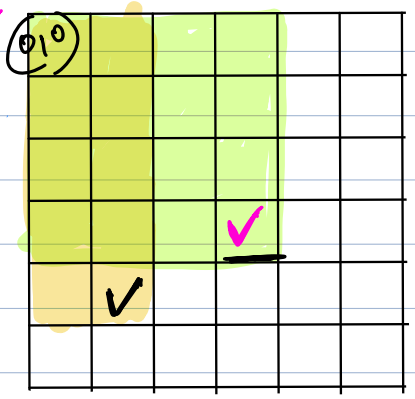
Brute force :- ↓ iterate over the submatrix

O(q * N*M)

input



pfsum



pf[i][j] = sum (0,0) to (i,j)

TL BR

0-i

(0,0) - (i,j)

a_0	b_0	c_0
a_1	b_1	c_1
a_2	b_2	c_2

Result

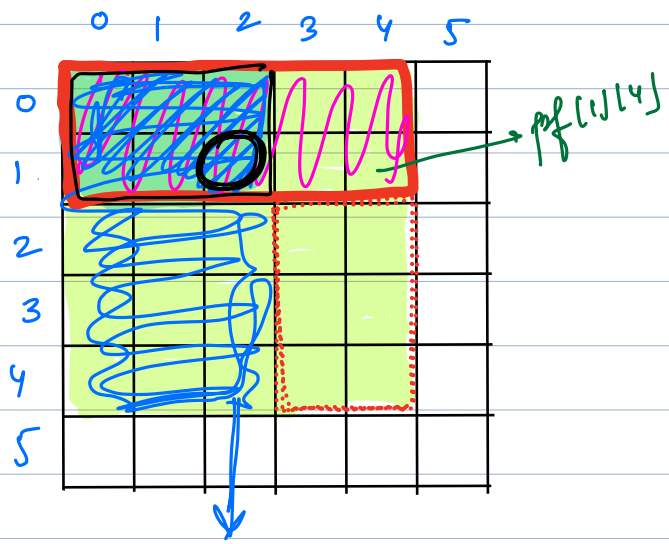
a_0	a_0+b_0	$a_0+b_0+c_0$
a_0+a_1	$a_0+b_0+a_1+b_1$	$a_0+b_0+c_0$ $a_1+b_1+c_1$
$a_0+a_1+a_2$	$a_0+a_1+a_2$ $b_0+b_1+b_2$	$a_0+a_1a_2$ $b_0+b_1+b_2$ $c_0+c_1+c_2$

1	3	2
6	5	9
-2	7	8

⟹

1	4	6
7 ~~6~~	15 ~~11~~	26 ~~20~~
5 ~~-2~~	20 ~~7~~	39 ~~15~~

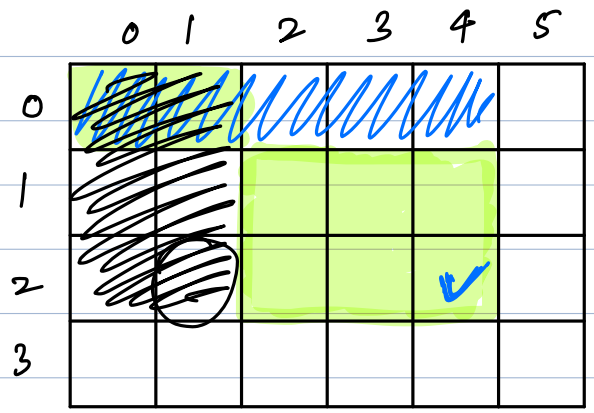
row wise
col wise



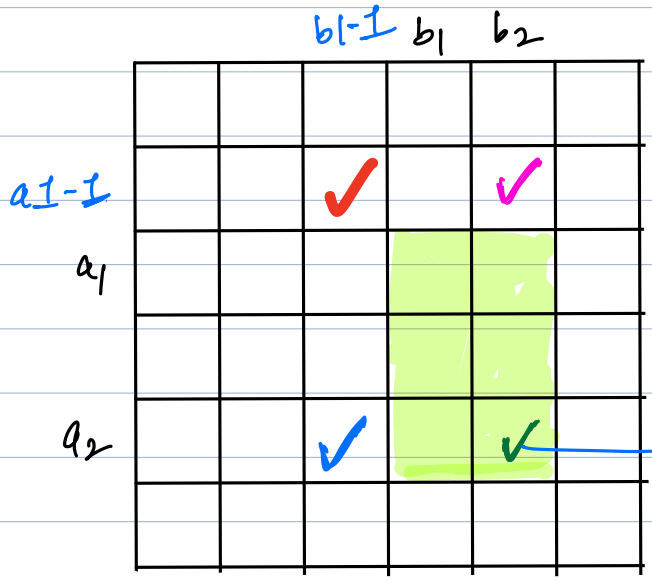
(2,3), (4,4)

pf[4][4] → O(1)

- pf[1][4]
- pf[4][2]
+ pf[1][2]




```
pf[2,4]
 - pf[0][4]
 - pf[2][1]
 + pf[0][1]
```



(a_1, b_1) (a_2, b_2)

```
pf[a2][b2]
 - pf[a1-1][b2]
 - pf[a2][b1-1]
 + pf[a1-1][b1-1]
```

$(a1, b1)$ $(a2, b2)$

TL BR

```
ans = pf[a2][b2]
if (a1 > 0)
    ans -= pf[a1-1][b2]
if (b1 > 0)
    ans -= pf[a2][b1-1]
if (a1 > 0 && b1 > 0)
    ans += pf[a1-1][b1-1]
```

N*M

Step 1: create pfsum

pf[i][j] = pf[i][j-1] + arr[i][j]

1) Take pfsum row wise
2) Take col wise on pf matrix

pf[i][j] += pf[i-1][j]

q

Step 2 for every query — O(1)

→ .

T.C: O(q + N*M)

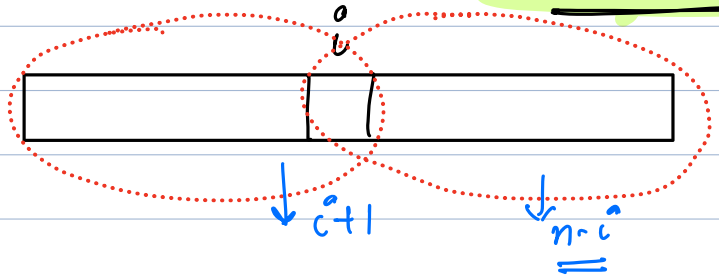
S.C: O(N*M)

Q

sum of all subarray sums

contribution

$(i+1)*(n-i)$



matrix → N*M. Find sum of all submatrices sum.

5	10	15	20
6	12	18	24
7	14	21	28
8	16	24	34

top left

Brute force :- consider all possible submatrices

T.L → a_1, b_1

B.R → a_2, b_2

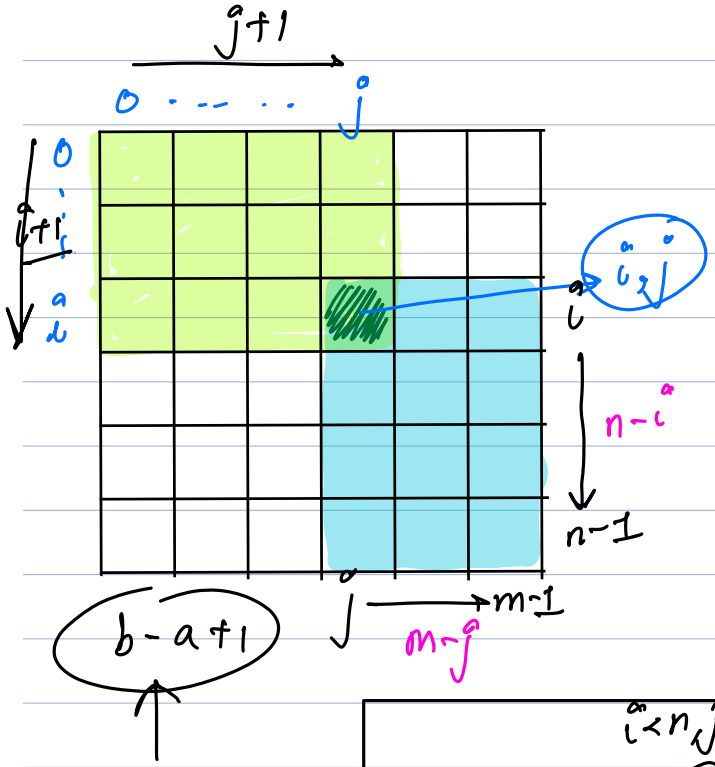
Fix TL

```
for(a1=0; a1<n; a1++)
{
    for(b1=0; b1<m; b1++)
    {
        a1, b1
        for(a2=a1; a2<n; a2++)
        {
            for(b2=b1; b2<m; b2++)
            {
                (a1, b1) (a2, b2)
                pfsum
            }
        }
    }
}
```

$O(n^2 m^2)$

In how many submatrices a particular cell $\Rightarrow$.

$$cont(i,j) = \#count * au[i][j]$$



count = valid combinat TL & BR

T.L = x, B.R = y, x * y

$(i+1) * (j+1)$ $\quad$ $(n-i) * (m-j)$

$$count = (i+1)(j+1) * (n-i) * (m-j)$$

$$ans = \sum_{i<n, j<m}^{i=0, j=0} (i+1) * (j+1) * (n-i) * (m-j) * au[i][j]$$

T.C: $O(N*M)$ $\quad$ $N^2 M^2$

Takeaway : If stuck in 2D quest
↳ Try to think what would if it 1-D you do

row wise, col-wise sorted matrix (asc order)

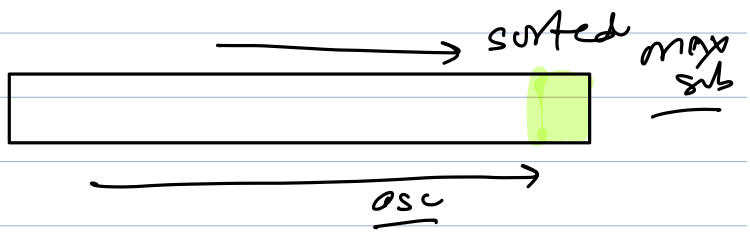
Find max submatrix sum

	0	1	2	3
0	-20	-16	-4	8
1	-10	-8	12	14
2	-1	6	21	30
3	5	7	28	42

max elmt

definitely include this

Basic idea :- consider all submat (TL & BR) $n^2 m^2$



T.C : O(n×m)

consider all options of TL

```
for ( a1=0→n)
  for ( b1=0→m)
```

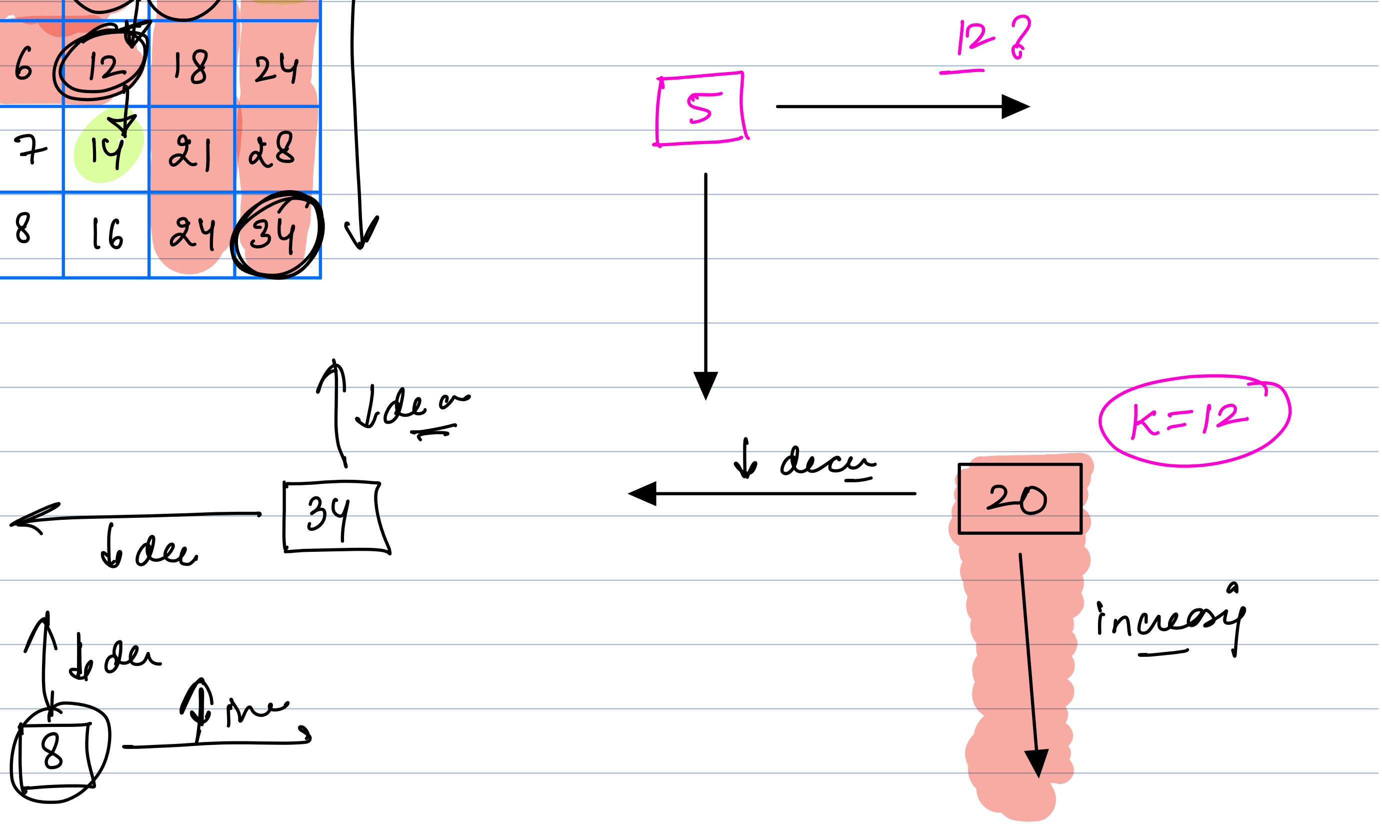
Q. rowwise, column-wise sorted matrix!
Search if element k is present or not!

k=14

5	10	15	20
6	12	18	24
7	14	21	28
8	16	24	34

k=14? ✓ k=22 ✗

Basic:- $O(N*M)$



life from 1 row 1 col

TR
BL

n*m

T.C: $O(m+n)$

n+m

```
int i=0, j=m-1;

while( i<n && j>=0)
{
    if( arr[i][j] == k)
    else if (arr[i][j] < k)
        i++;
    else    j--;
}
```

< k
12
14