Content

⟹ Submatrix (2D) sum queries

⟹ Sum of all submatrices

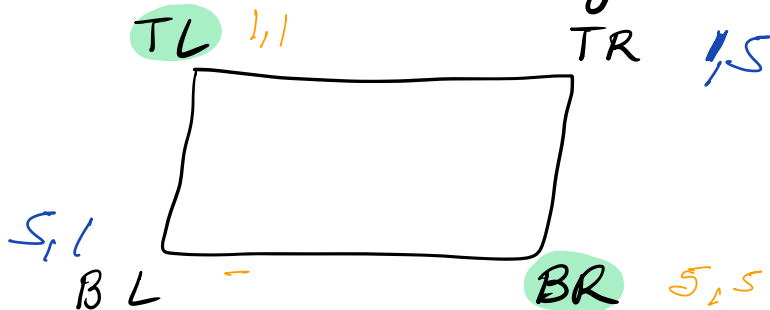⟹ Max submatrix sum (sorted)

⟹ Search in sorted matrix

Q1 Given int mat[N][M], for each query q,
CRED find sum of given submatrix.
                              ↳ continuous part of matrix.

● How is submatrix given in query?

TL 1,1   TR 1,5

S,1
  BL   —   BR 5,5

Eg —   0   1   2   3

     0   2   -1   3   2

     1   3   2   6   2

     2   10   9   8   2

     3   4   -1   2   3

     4   3   2   6   9

                              TL   BR
q ⟹ 2,1 , 4,2

Brute: Iterate on the whole submatrix

$$O(n*m)$$

q queries $\Rightarrow$ $O(qnm)$

pf i, j

- **Idea:** Prefix sum

submatrix
0, 0   i, j

	0	1	2	3	4
0					
1					
2					
3					
4					

$\Rightarrow$ pf [1][3]

pf $\Rightarrow$ sum of submatrix

TL $\Rightarrow$ 0, 0

BR $\Rightarrow$ i, j

- Assume pf is calculated. Answer Query.

TL   BR

a_1, b_1 a_2, b_2

2,1 to 3,3

$$= pf[3][3]$$
$$- pf[3][0]$$
$$- pf[1][3]$$
$$+ pf[1][0]$$

2,2 to 4,3

$$pf[4][3]$$
$$- pf[4][1]$$
$$- pf[1][3]$$
$$+ pf[1][1]$$

Generalize   TL   BR

a_1, b_1 a_2, b_2

$$pf[a_2][b_2] - pf[a_2][b_1-1]$$
$$- pf[a_1-1][b_2]$$
$$+ pf[a_1-1][b_1-1]$$

$pf[e] - pf[s-1]$

```
int sum ( int a₁, int b₁, a₂, b₂) {
    sum = pf[a₂][b₂]
    if ( b₁-1 ≥ 0)
        sum - = pf[a₂][b₁-1]
    if ( a₁-1 ≥ 0)
        sum - = pf[a₁-1][b₂]
    if ( a₁-1 ≥ 0  &&  b₁-1 ≥ 0)
        sum + = pf[a₁-1][b₁-1]
}
```

Total TC: $O(n*m + q)$

● We can now answer queries using pf.

● How to build pf ?

vvv simple

$\Rightarrow$

Step 1) Apply row-wise prefix sum

Step 2) Apply column-wise prefix sum

	0	1	2
0	a_0	b_0	c_0
1	a_1	b_1	c_1
2	a_2	b_2	c_2

a_0	a_0+b_0	$a_0+b_0+c_0$
a_0+a_1	$a_0+b_0+a_1+b_1$	$a_0+b_0+c_0 a_1+b_1+c_1$
$a_0+a_1+a_2$	$a_0+b_0 a_1+b_1 a_2+b_2$	all

row wise
prefix su

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

column wise
pref sum

	0	1	2
0	a_0	a_0+b_0	$a_0+b_0+c_0$
1	a_0+a_1	$a_0+b_0+a_1+b_1$	$a_0+b_0+c_0+a_1+b_1+c_1$
2	$a_0+a_1+a_2$	$a_0+b_0 a_1+b_1 a_2+b_2$	all

	0	1	2
0	1	2	3
1	4	5	6
2	7	8	9

row →

1	3	6
4	9	15
7	15	24

TC: $O(n*m)$

21 − 12 − 6
← 3

0 ↓1 2 col

	0	1	2
0	1	3	6
1	5	12	21
2	12	27	45

Q2 Given arr [N] [M], calc sum of all ==Submatrix sums==

E.x.

```
      0    1
  0 ⎡ 3    1 ⎤
  1 ⎣-1   -2 ⎦
```

$$\begin{bmatrix} 3 & 1 \\ -1 & -2 \end{bmatrix} = 1$$

$[3] = 3$	$[3,1] = 4$
$[1] = 1$	$[-1,-2] = -3$
$[-1] = -1$	$\begin{bmatrix} 3 \\ -1 \end{bmatrix} = 2$
$[-2] = -2$	$\begin{bmatrix} 1 \\ -2 \end{bmatrix} = -1$

$tot = 4$

● Idea from intermediate for 1-D array

Sum of all subarray sums ⇒

Contribution Technique

Thus we will use the same technique here. Calculate number of submatrices which contain cell (i,j)

```
⎡  |  | i |  /  /  / ⎤
⎣  |  |   |  /  /  / ⎦
```

$$TL = 3 \times 4 = 12$$
$$(2+1)(3+1)$$

$$BR = 3 \times 2 = 6$$
$$(5-2) \times (5-3)$$

$$3 \times 4 \times 3 \times 2 = 72$$
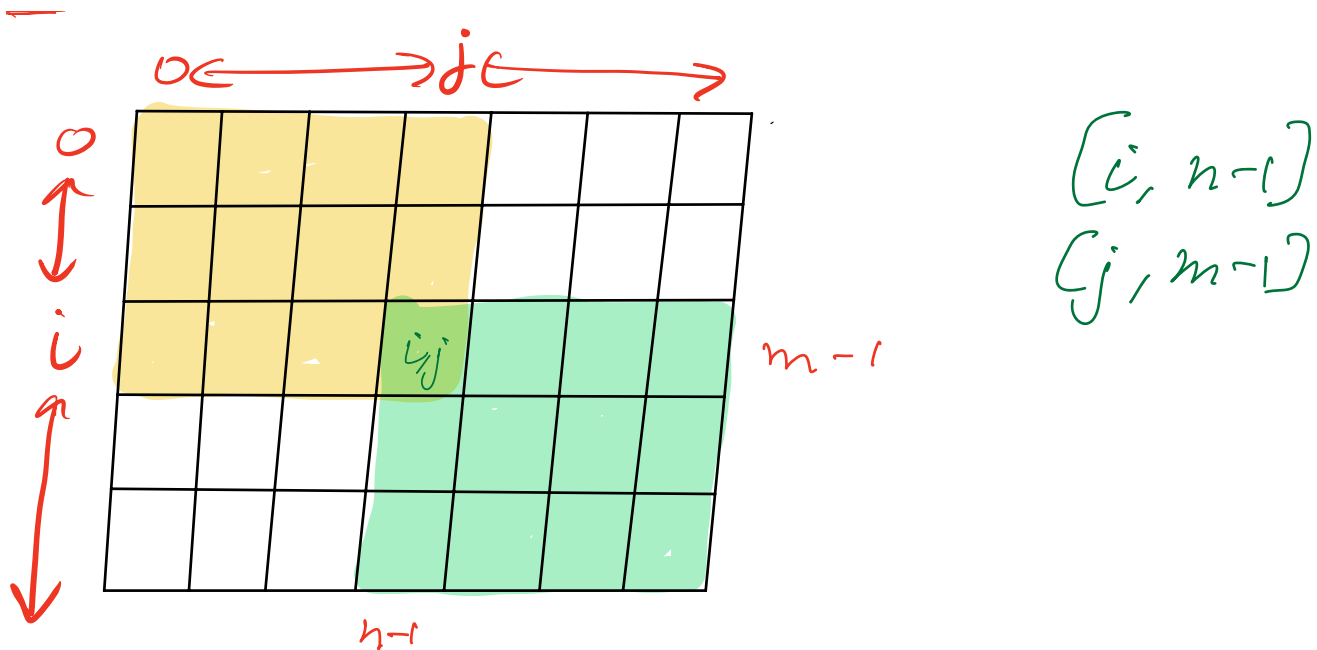
$$TL = (i+1)(j+1)$$
$$BR = (n-i)(m-j)$$

total no of sub matrices
$$= TL \times BR$$
$$(i+1)(j+1)(n-i)(m-j)$$

For every TL & BR, we get unique submatrix

$(i, n-1)$

$(j, m-1)$

Total number =
$$(i+1)(j+1)(n-i)(m-j)$$

Code sum = 0
for (i = 0 ; i < n ; i++) {
  for (j = 0 ; j < m ; j++) {
    sum += $a[i][j]$ *
      $(i+1)(j+1)(n-i)(m-j)$
  }
}

TC: $O(nm)$ SC: $O(1)$

Q3 Given arr[N][M] find max sub sum. Matrix is row-wise & col-wise ~~sorted~~ ascending

Eg

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

	0	1	2
0	-20	-16	-4
1	-10	-8	12
2	-1	6	21

Idea 1 : Try all submatrices

Obs: Always BR = n-1, m-1

TL = no idea

Create pf sum matrix

TL   BR

Try all points   n-1, m-1
  as top left

Code

1) Create pf matrix

```
ans = Integer.Min_Value
for (i=0; i<n; i++) {
  for (j=0; j<m; j++) {
    TL =   i,j
    BR =  n-1, m-1
    //Get sum of submatrix
    ans = max (sum, ans)
  }
}
```

TC: O(nm)   SC: O(1)

Q4 Given row-wise and col-wise sorted matrix, find k. duplicates allowed

	0	1	2	3	4	5
0	-10	-5	-2	2	4	7
1	-7	-4	-1	3	6	9
2	-2	3	5	7	11	14
3	3	6	8	11	14	12
4	7	11	12	15	19	20
5	10	14	18	20	24	29

$k = 12$

Brute:

Iterate whole matrix

TC: $O(n * m)$

Idea: Start at Top right corner

If   arr[i][j] < k

         i++

arr[i][j] > k

         j--

Code

```
bool search (int a[][], int k) {
    i = 0          j = m-1
    while ( i < n  && j >= 0) {
        if (a[i][j] == k)
            return true
        else if ( a[i][j] > k)
            j--
        else        // a[i][j] < k
            i++
    }
    return false
}
```

y

TC: $O(n+m)$ SC: $O(1)$

{done}

Mon   Tue   ~~Wed~~   Fri

S.S