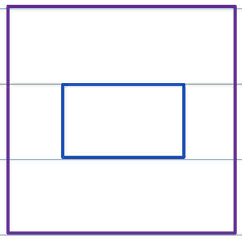
Todays Content

Intro to Submatrices

Submatrix sum query

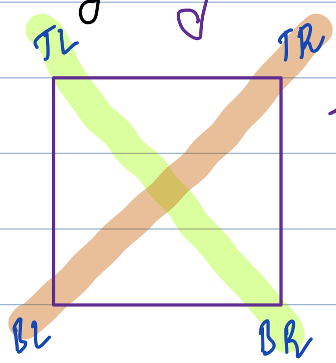
Max Submatrix sum .

Submatrix: Part of matrix is submatrix.



1. Single element is also submatrix
2. Complete matrix also submatrix.

Identify: Any submatrix has corners



To identify a submatrix we need 2 opposite points

a. If we have 2 opposite points, can identify submatrix

Case1: TL & BR or Case2: TR & BL

Example:

int mat[5][5]:

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

TL and BR		TR and BL	
[1, 1]	[3, 2]	[1 2]	[3 1]
[2, 2]	[4, 4]	[2 4]	[4 2]
[1, 0]	[3, 3]	[1 3]	[3 0]
[1, 1]	[3, 3]	[1 3]	[3 1]

Note: In questions on submatrix generally TL & BR is given.

1Q Find sum of all element in given submatrix

Constraints:

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

TL and BR

r_1 c_1 r_2 c_2

[1, 1] [3, 4] : 74

```
            TL                       BR
int sum(int mat[][], int r1, int c1, int r2, int c2){  TC: O(N*M)  SC: O(1)
    long ans = 0;
    for(int i = r1; i <= r2; i++){
        for(int j = c1; j <= c2; j++){
            ans = ans + mat[i][j];
        }
    }
    return ans;
}
```

2Q

Given mat[N][M] and Qmat[Q][4]

Each row in Qmat represents a query.

Each query contains 4 ele

i^{th} row in Qmat[i][4] represents:

Qmat[i][0] : r_1 Qmat[i][1] : c_1 Qmat[i][2] : r_2 Qmat[i][3] : c_2

$\boxed{r_1 \text{ \& } c_1 \text{ represents TL}}$ $\boxed{r_2 \text{ \& } c_2 \text{ represents BR.}}$

For every query calculate sum of all elements in submatrix & print

Ex: mat[5][5] =

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

Qmat[3][4]:

	0: r_1	1: c_1	2: r_2	3: c_2	Output:
→ 0	1	1	3	3	→ 57
1	1	2	3	4	→ 56
2	2	1	4	3	→ 45

Ideal: For every Query

Iterate on submatrix from (r_1 c_1) to (r_2 c_2) calculate & print sum

TC: O(Q * (N*M)) SC: O(1)

↳ # Iterating on submatrix.

↳ # No: of Queries

Optimization:

Similar question in 1D arrays:

pSum[i] = Sum of all elements from [0.. i]

Note: Prefix Sum:

Sum of all from start to that index

Extend 2D Matrices:

TL BR

pmat[i][j] = Sum of all elements from (0, 0) .... (i, j)

mat[3][5] ⟶ pmat[3][5]

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

	0	1	2	3	4
0	3	13	22	28	33
1	6	20	38	52	59
2	8	33	57	76	90

Step1: Create pfMat[][]

Given ar[N] create pf[N]

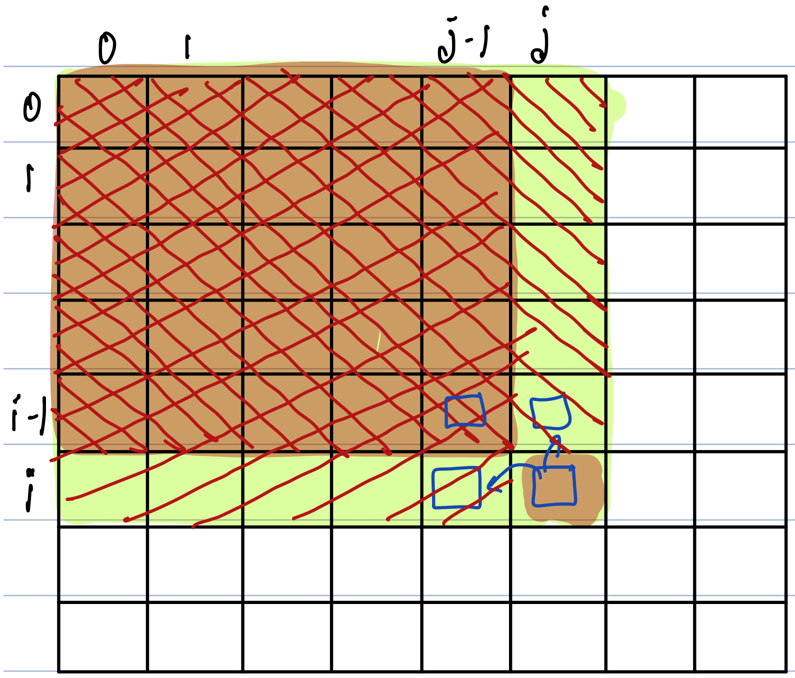
Pf[i] = sum of all elements from [0..i] # $\boxed{a_0 + a_1 + a_2 + \cdots a_{i-1}} + \boxed{a_i}$

Pf[i] = Pf[i-1] + a_i # Calculated using previous

Given mat[N][M] create Pfm[N][M]

Pfm[i,j] = sum of all elements from (0,0) to (i,j)

Pfm[i,j] = Pfm[i, j-1] + Pfm[i-1,j] - Pfm[i-1, j-1] + mat[i][j]



Given mat[N][M], create Pfm[N][M] TC: O(N*M) SC: O(1)

If we return: O(1)

```
int Pfm[N][M];
for(int i=0; i<N; i++){
    for(int j=0; j<M; j++){
        Pfm[i,j] = mat[i][j]
        if(j>0){ Pfm[i][j] += Pfm[i, j-1] }
        if(i>0){ Pfm[i][j] += Pfm[i-1,j] }
        if(i>0 && j>0){ Pfm[i][j] -= Pfm[i-1, j-1] }
    }
}
return Pfm;
```

Step2: Answer queries using pfm[][]

Given pf[N] get sum of all elements from {s .. e}

ans = pf[e] - pf[s-1]

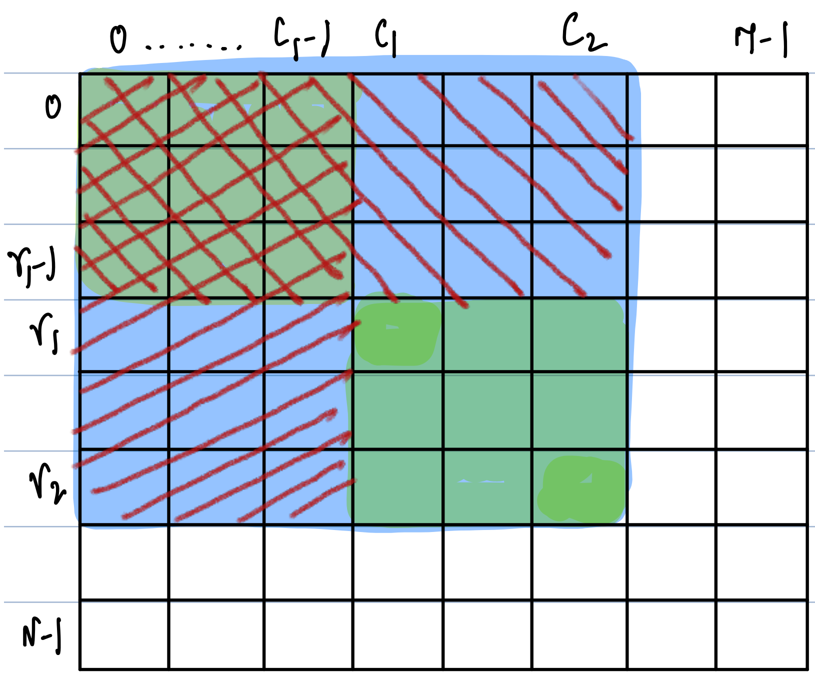
Given pfm[N][M] get sum of all elements from $\{\overset{s}{r_1\ c_1}\}$ $\{\overset{e}{r_2\ c_2}\}$

ans = pfm[r_2][c_2] - $\underset{\boxed{\text{if } c_1 > 0}}{\text{pfm}[r_2][c_1-1]}$ - $\underset{\boxed{\text{if } r_1 > 0}}{\text{pfm}[r_1-1][c_2]}$ + $\underset{\boxed{\text{if } r_1>0 \text{ \&\& } c_1>0}}{\text{pfm}[r_1-1][c_1-1]}$

We removed it Twice, hence we add it again

for mat[N][M]

Create pfmat[N][M]



TC: O(N*M + Q)   SC: O(N*M) # Because of Pfm[][]
↳ # Can store in mat[][], if datatype is same.

```
void printQueries(int mat[][], int N, int M, int Qmat[][], int Q){

    # Step1: Create Pfm[N][M]
        int Pfm[N][M];
        for(int i=0; i<N; i++){
            for(int j=0; j<M; j++){
                Pfm[i,j] = mat[i][j]
                if(j>0){ Pfm[i][j] += Pfm[i, j-1] }
                if(i>0){ Pfm[i][j] += Pfm[i-1, j] }
                if(i>0 && j>0){ Pfm[i][j] -= Pfm[i-1, j-1] }
            }
        }

    # Step2: For every query ans using Pfm[][]
        for(int i=0; i<Q; i++){
            int r1 = Qmat[i][0], c1 = Qmat[i][1];
            int r2 = Qmat[i][2], c2 = Qmat[i][3];
            long ans = Pfm[r2][c2]
            if(c1>0){ ans -= Pfm[r2][c1-1] }
            if(r1>0){ ans -= Pfm[r1-1][c2] }
            if(r1>0 && c1>0){ ans += Pfm[r1-1][c1-1] }
            print(ans);
        }
}
```

Given row-wise & col wise sorted matrix of mat[N][M]

Find Maximum submatrix sum :

Ex1:

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

#Idea:

Max submatrix sum will end at {N-1, M-1}

TL	BR
?	(N-1, M-1)

Hint1: Consider each cell as TL & BR (N-1, M-1)

Iterate & calculate submatrix sum

& get overall max.

$TC: O(\underbrace{N*M}_{\text{All possible TL}} * \underbrace{\{N*M\}}_{\text{To iterate on submatrix}})$

Ex2:

	0	1	2	3
0	-20	-16	-4	-1
1	-10	-8	-2	5
2	-4	2	4	8

Ex3:

	0	1	2
0	-50	-40	-30
1	-35	-20	-15
2	-19	-14	-3

Hint2: Consider each cell as TL & BR (N-1, M-1)

Calculate submatrix sum using pfm[][]

& get overall max.

$TC: O(\underbrace{N*M}_{\text{All possible TL}} * \underbrace{\{1\}}_{\text{To calculate submatrix sum using pfm[][]}} + N*M)$ SC: $O(N*M)$