SCALER

2D Matrices

TABLE OF CONTENTS

Notes

Search element "k" in row-wise & column-wise sorted 2D array

$$\begin{bmatrix} -5 & -2 & 1 & 13 \\ -4 & 0 & 3 & 14 \\ -3 & 2 & 5 & 18 \\ 2 & 6 & 10 & 20 \end{bmatrix}_{N * M}$$

Search (6) → True

Search (15) → False

Search (0) → True

Quiz : 1

BF Idea

Iterate over the matrices & check if element = K or n't.

T.C : O(N*M)

S.C : O(1)

Optimisation Approach

Observation : Matrix is sorted.

$$
\begin{array}{c|cccc}
 & 0 & 1 & 2 & 3 \\
\hline
0 & -5 & -2 & 1 & 13 \\
1 & -4 & 0 & 3 & 14 \\
2 & -3 & 2 & 5 & 18 \\
3 & 2 & 6 & 10 & 20
\end{array}
$$

Search (6)

Case 1 : Start from TOP LEFT ✗

L, L

Ambiguous situation b'coz k can be in Row & well as col. so not sure where to move.

Case 2 : Start from BOTTOM RIGHT ✗

S, S

Ambiguous situation b'coz k can be in Row & well as col. so not sure where to move.

Case 3 : Start from TOP RIGHT

S, L

mat[i][j]
- < k : Discard Row & search in col. Move down
- == k : Return TRUE
- > k : Discard col & search in Row. Move in left direction.

Case 4 : Start from BOTTOM LEFT

S, L

mat[i][j]
- < k : Discard col & search in Row. Move in right direction.
- == k : Return TRUE
- > k : Discard Row & search in col. Move in upward direction.

$$
\begin{array}{c|cccc}
 & 0 & 1 & 2 & 3 \\
\hline
0 & -5 & -2 & 1 & 13 \\
1 & -4 & 0 & 3 & 14 \\
2 & -3 & 2 & 5 & 18 \\
3 & 2 & 6 & 10 & 20
\end{array}
$$

Search (6)

STOP

$$
\begin{array}{c|cccc}
 & 0 & 1 & 2 & 3 \\
\hline
0 & -5 & -2 & 1 & 13 \\
1 & -4 & 0 & 3 & 14 \\
2 & -3 & 2 & 5 & 18 \\
3 & 2 & 6 & 10 & 20
\end{array}
$$

$-1 > -3$

Search (-1)

< / > Code

```
i=0 , j = m-1;        // TOP RIGHT as starting pt.
while ( i < n  && j >= 0) {
    if (arr[i][j] = = k)        return True;
    else if (arr[i][j] < k)
            i++;                // Move down.
    else if (arr[i][j] > k)
            j--;                // Move left
}
return False;
```

T.C: O(n+m)

S.C : O(1)

Row with maximum number of 1's

Given a row-wise sorted matrix A of size N*N. Find the row with the maximum number of 1's.

Note:

If 2 rows have same no of 1’s then return the row which has lower index.

Count of 1's

$$A = \left[\begin{array}{cc} & \begin{array}{ccc} 0 & 1 & 2 \end{array} \\ 0 & [\,0, 1, 1\,] \\ 1 & [\,0, 0, 1\,] \\ 2 & [\,0, 1, 1\,] \end{array}\right] \quad \begin{array}{c} \\ 2 \longrightarrow \text{Ans} = 0^{th}\text{ row} \\ 1 \\ 2 \end{array}$$

Count of 1's

$$A = \left[\begin{array}{cc} & \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ 0 & [\,0, 0, 0, 0\,] \\ 1 & [\,0, 0, 0, 1\,] \\ 2 & [\,0, 0, 1, 1\,] \\ 3 & [\,0, 1, 1, 1\,] \end{array}\right] \quad \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \longrightarrow \text{Ans} = 3^{rd}\text{ row} \end{array}$$

Quiz : 2

	0	1	2	3	Count of 1's
0	0	1	1	1	3
1	0	0	0	1	1
2	1	1	1	1	4 $\longrightarrow$ Ans = 2^{nd} row
3	1	1	1	1	4

BF Idea

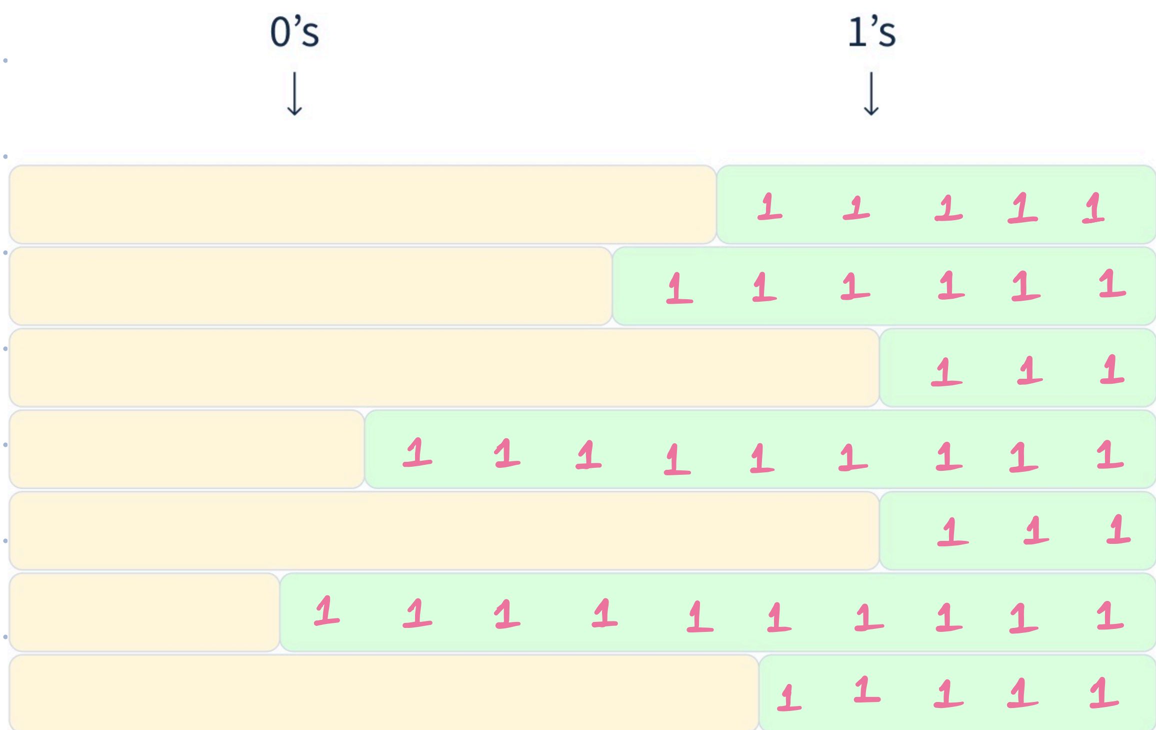
Iterate on each & every row & count no of 1's.

T.C : $O(N^2)$

S.C : $O(1)$

Optimisation Approach

Observation



Algorithm start from TOP RIGHT & check arr[i][j]

arr[i][j]
- $==1$ Move towards left & check for more 1's in same row.
- $==0$ Move in downward direction

	0	1	2	3	4	5
0	0	0	0	0	1	1
1	0	0	1	1	1	1
2	0	0	0	0	0	1
3	0	0	0	0	1	1
4	0	1	1	1	1	1
5	0	0	0	1	1	1

STOP

ans = $\cancel{0}$ $\cancel{2}$ 4

	0	1	2	3
0	0	0	1	1
1	1	1	1	1
2	0	1	1	1
3	0	1	1	1
4	0	0	0	1

STOP

ans = $\cancel{0}$ 1

</> Code

```
i = 0 , j = N-1;
while(i < n && j >= 0) {
    // Keep moving left till we get 0.
    while( j >= 0 && arr[i][j] == 1) {
        j--;
        ans = i ;
    }
    // Move down
        i++;
}
```

T.C : O(N+M)

S.C: O(1)

Print Boundary Elements

Given a matrix of N * N i.e. Mat [N][N], print boundary elements in clockwise direction.

mat[N][N]

mat[5][5]

	0	1	2	3	4
0	1	2	3	4	5
1	6	7	8	9	10
2	11	12	13	14	15
3	16	17	18	19	20
4	21	22	23	24	25

N = 5

4 ele →

4 ele ↓

4 ele ←

4 ele ↑

o/p → [1 , 2 , 3 , 4 , 5 , 10 , $\overset{15}{\wedge}$ 20 , 25 , 24 , 23 , 22 , 21 , 16 , 11 , 6]

Quiz : 3

1	2	3
4	5	6
7	8	9

o/p : 1 , 2 , 3 , 6 , 9 , 8 , 7 , 4

N = 3

2 ele →

2 ele ↓

2 ele ←

2 ele ↑

Approach

(n - 1) ele →

(n-1) ele ↓

(n-1) ele ←

(n-1) ele ↑

N * M

(m-1) →

(n-1) ↓

(m-1) ←

(n-1) ↑

</> Code

```
Void print Boundary( arr[N][N]) {
    i=0 , j=0;
    // (n-1) ele →
    for ( k=1 ; k<n ; k++){
        print arr[i][j];
        j++;
    }

    // i=0 , j = (n-1) → print (n-1)ele ↓
    for ( k=1 ; k<n ; k++){
        print arr[i][j];
        i++;
    }

    // i=(N-1) , j=(N-1) → print (n-1) ele ←
    for ( k=1 ; k<n ; k++){
        print arr[i][j];
        j--;
    }

    // i=(N-1) , j=0 → print (n-1) ele ↑
    for ( k=1 ; k<n ; k++){
        print arr[i][j];
        i--;
    }
}
```

T.C : $O(4N) = O(N)$
S.C : $O(1)$

Break till 8:45

Spiral Matrix

mat[N][N]

	0	1	2	3	4	5
0	1	2	3	4	5	6
1	7	8	9	10	11	12
2	13	14	15	16	17	18
3	19	20	21	22	23	24
4	25	26	27	28	29	30
5	31	32	33	34	35	36

i	j	N	Print (N-1)
0	0	6	5
1	1	4	3
2	2	2	1
3	3	0 STOP	

o/p → [1 , 2 , 3 , 4 , 5 , 6 , 12 , 18 , 24 , 30 , 36 , 35 , 34 , 33 , 32 , 31 , 25 , 19 ,
13 , 7 , 8 , 9 , 10 , 11 , 17 , 23 , 29 , 28 , 27 , 26 , 20 , 14 , 15 , 16 , 22 , 21]

Quiz : 4

	0	1	2	3
0	13	14	12	8
1	9	1	2	7
2	0	4	3	0
3	10	5	6	11

i	j	N	(N-1)
0	0	4	3
1	1	2	1
2	2	0 ↓ STOP	

o/p : 13, 14, 12, 8, 7, 0, 11, 6, 5, 10, 0, 9, 1, 2, 3, 4

Idea

Break this problem into BOUNDARY PRINTING PROBLEM.

$$
\begin{array}{c|ccccc}
 & 0 & 1 & 2 & 3 & 4 \\
\hline
0 & 1 & 2 & 3 & 4 & 5 \\
1 & 6 & 7 & 8 & 9 & 10 \\
2 & 11 & 12 & 13 & 14 & 15 \\
3 & 16 & 17 & 18 & 19 & 20 \\
4 & 21 & 22 & 23 & 24 & 25
\end{array}
$$

i	j	N	N - 1
0	0	5	4
1	1	3	2
2	2	1	0

$\downarrow$

if (N == 1)
print A[i][j]

</> Code

```
while (N>1) {
    i=0, j=0;

    // (n-1) ele →
    for (k=1; k<N; k++) {
        print (arr[i][j];
        j++;
    }

    // i=0, j=N-1 → print (n-1) ele ↓
    for (k=1; k<N; k++) {
        print (arr[i][j];
        i++;
    }

    // i=N-1, j=N-1 → print (n-1) ele ←
    for (k=1; k<N; k++) {
        print (arr[i][j];
        j--;
    }

    // i=N-1, j=0 → print (n-1) ele ↑
    for (k=1; k<N; k++) {
        print (arr[i][j];
        i--;
    }

    // i=0, j=0;
    i++, j++, N=N-2;
}
if (N==1)
    print arr[i][j];
```

T.C: $O(N^2)$
S.C: $O(1)$

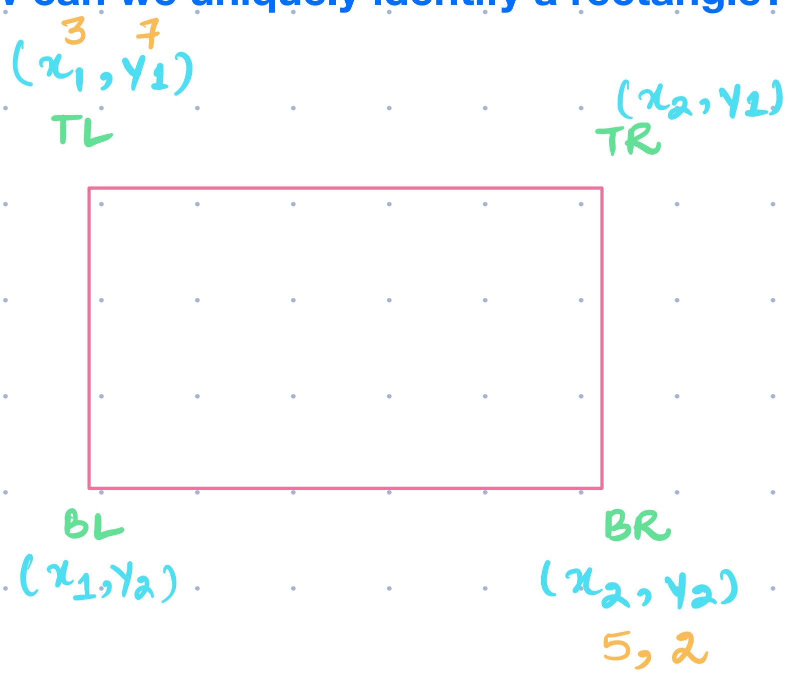
Sub - Matrix

Contiguous part of a matrix.

- Single cell is a sub matrix.
- Entire matrix is also a s sub matrix.

	0	1	2	3
0	1	2	3	4
1	5	6	7	8
2	9	10	11	12
3	13	14	15	16

How can we uniquely identify a rectangle?



TL & TR ✗

BL & BR ✗

TL & BR ✓

TR & BL ✓

Sum of all Sub-matrices Sum

Given mat[N] [M]. Find sum of all sub-matrix sums.

	0	1	2
0	4	9	6
1	5	-1	2

$\overset{4}{[4]}$ $\overset{13}{[4, 9]}$ $\overset{19}{[4, 9, 6]}$ $\overset{9}{\begin{bmatrix}4\\5\end{bmatrix}}$ $\overset{17}{\begin{bmatrix}4 & 9\\5 & -1\end{bmatrix}}$ $\overset{25}{\begin{bmatrix}4 & 9 & 6\\5 & -1 & 2\end{bmatrix}}$

$\overset{9}{[9]}$ $\overset{15}{[9, 6]}$ $\overset{8}{\begin{bmatrix}9\\-1\end{bmatrix}}$ $\overset{16}{\begin{bmatrix}9 & 6\\-1 & 2\end{bmatrix}}$

$\overset{6}{[6]}$ $\overset{8}{\begin{bmatrix}6\\2\end{bmatrix}}$

$\overset{5}{[5]}$ $\overset{4}{[5 \quad -1]}$ $\overset{6}{[5 \quad -1 \quad 2]}$

$\overset{-1}{[-1]}$ $\overset{1}{[-1 \quad 2]}$

$\overset{2}{[2]}$

Sum = 166

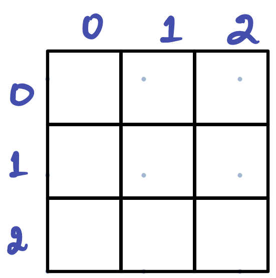
BF Idea

Generate all submatrices & find sum.

Total no of submatrices : $\frac{N(N+1).M(M+1)}{4}$

T.C : $O(N^3M^3)$
S.C : $O(1)$

Total no of Submatrices



$N = M = 3$

r_s	r_e	c_s	c_e
0	0	0	0
	1		1
	2		2
1	1	1	1
	2		2
2	2	2	2

Total no of ways to choose r_s and r_e : $\frac{N(N+1)}{2}$

Total no of ways to choose c_s and c_e : $\frac{M(M+1)}{2}$

Total no of submatrices : $\frac{N(N+1)}{2} \cdot \frac{M(M+1)}{2}$

Idea   Contribution Technique

Quiz : 5

	0	1	2	3
0	TL	TL	TL	
1	TL	TL	TL	
2	TL	TL	TL BR	BR
3			BR	BR
4			BR	BR

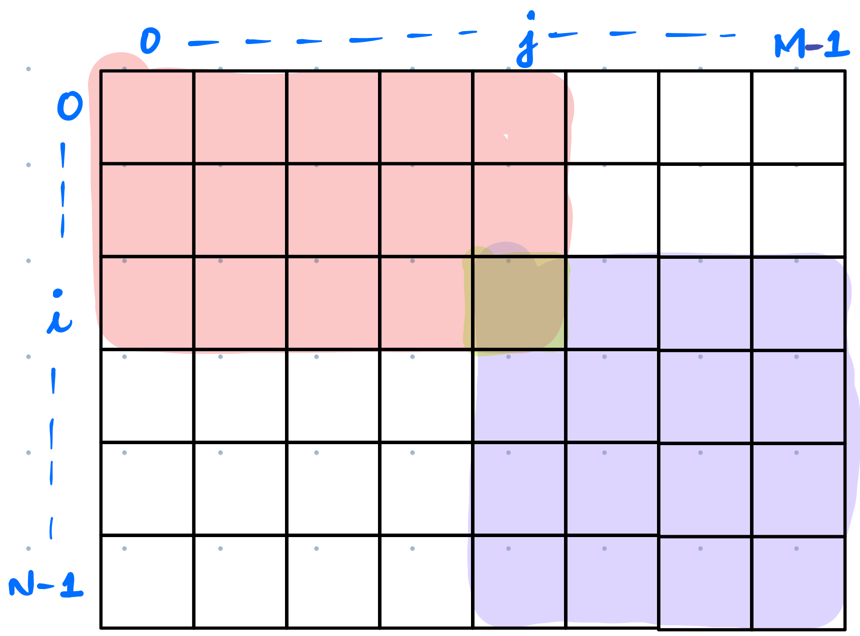
- In how many sub - matrices (2 , 2) will be present?

Options for TL = 9

Options for BR = 6

Options for TL * Options for BR = 9*6 = 54

- In how many sub - matrices (i , j) will be $(i+1)(j+1) * (N-i)(M-j)$



Options for TL :

$[0 \ldots i]$ $[0 \ldots j]$

$\boxed{(i+1)(j+1)}$

Options for BR :

$[i \ldots N-1]$ $[j \ldots M-1]$

$(N-\cancel{1}-i+\cancel{1})(M-\cancel{1}-j+\cancel{1})$

$\boxed{(N-i)(M-j)}$

< / > Code

```
total = 0;

for (i = 0; i < n; i++) {
    for (j = 0; j < m; j++) {
        top_left = (i+1) * (j+1);
        bottom_right = (n-i) * (m-j);
        Contribution = a[i][j] * top_left * bottom_right;
        total += contribution;
    }
}
return total;
```

$\underbrace{\text{top\_left} * \text{bottom\_right}}_{\text{no of submatrices in which (i,j) will occur.}}$

no of submatrices in which (i,j) will occur.

T.C : O(N * M)

S.C : O(1)