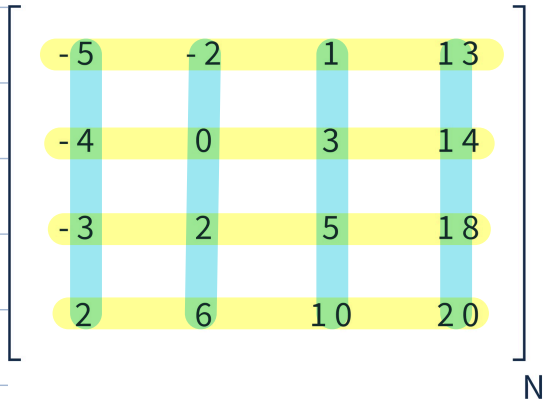
SCALER

2D Matrices

TABLE OF CONTENTS

Search in row-wise & column-wise sorted 2D array

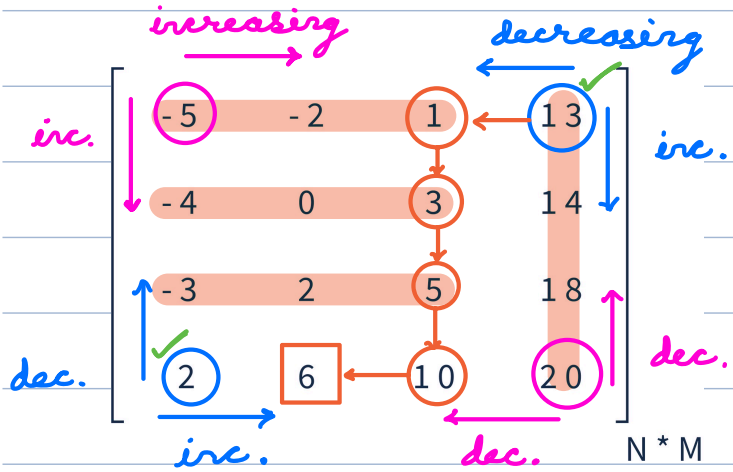


Search (6) → ✓ true

Search (15) → ✗ false

Bruteforce → Travel all cells.

TC = O(N * M) SC = O(1)



travel from corners.

Search (6) → true

Start → Top Right or Bottom Left

```
r = 0    c = M-1
while ( r < N && c >= 0 ) {
    if ( A[r][c] == K )   return true
    else if ( A[r][c] < K )   r++
    else      c--
}
return false
```

TC = O(N+M) SC = O(1)

Row with maximum number of 1's

Given a binary sorted matrix A of size N*N. Find the row with the maximum number of 1's [Only rows are sorted]

0 0 . . . 0 1 1 . . . 1

```
        0   1   2
A = [ 0 [ 0 , 1 , 1 ]
    1 [ 0 , 0 , 1 ]
    2 [ 0 , 1 , 1 ]  ]
```

Ans = 0

If multiple rows are present, return the first row.

```
        0   1   2   3
A = [ 0 [ 0 , 0 , 0 , 0 ]
    1 [ 0 , 0 , 0 , 1 ]
    2 [ 0 , 0 , 1 , 1 ]
    3 [ 0 , 1 , 1 , 1 ]  ]
```

Ans = 3

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

Ans = 2

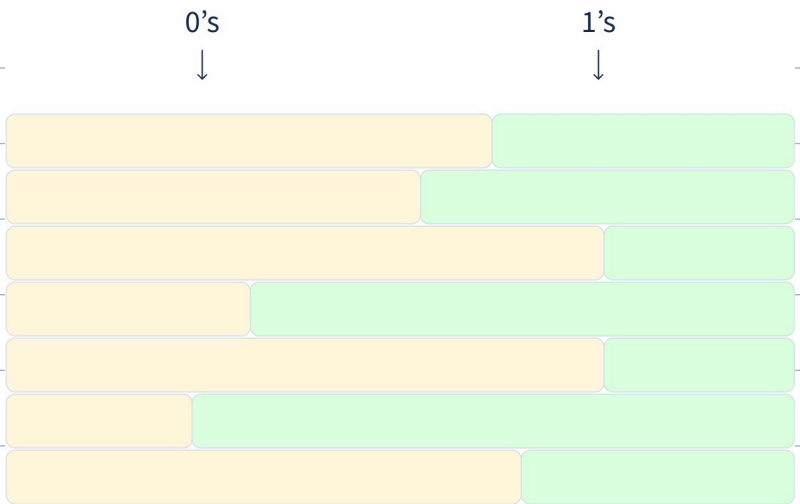
Bruteforce → ∀ rows, count #1's.

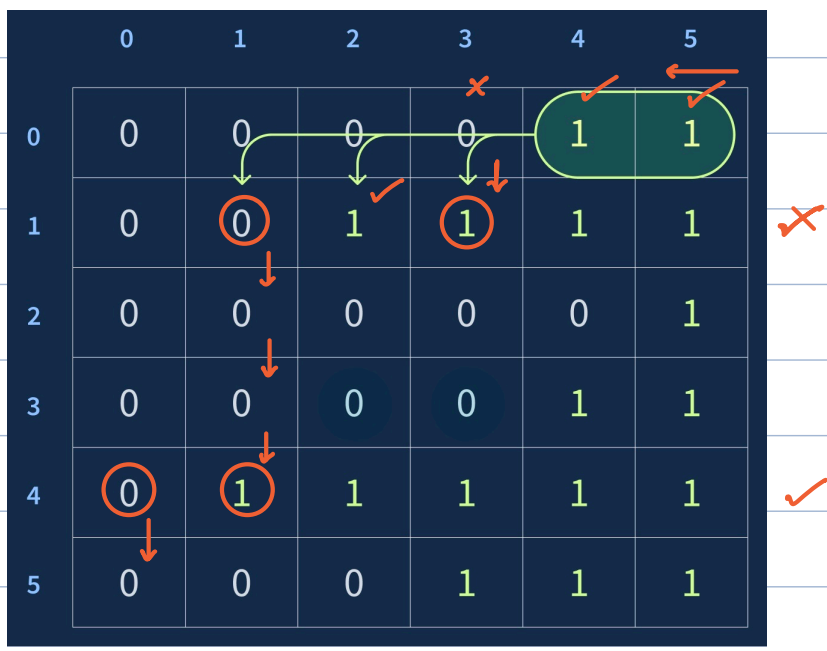
TC = $O(N^2)$ SC = $O(1)$

Observation

1) Row with min 0's ✓

2) First 1 can give #1's ✓





Ans = 4

```
ans = 0
r = 0      c = M-1
while (r < N && c >= 0) {
    while (c >= 0 && A[r][c] == 1) {
        c--
        ans = r
    }
    r++
}
return ans
```

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

ans = ~~0~~ 2

TC = O(N + N) = O(N) SC = O(1)

Print Boundary Elements

mat[N][N] ⟶ Square Matrix

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

o/p → [1 , 2 , 3 , 4 , 5 , 10 , 15 , 20 , 25 , 24 , 23 , 22 , 21 , 16 , 11 , 6]

A = $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$

o/p → 1 2 3 6 9 8 7 4

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

```
r = 0    c = 0
while ( c < (N-1)) {                    →
    print ( A[r][c])
    c++
}
while ( r < (N-1)) {                    ↓
    print ( A[r][c])
    r++
}
while ( c > 0) {                        ←
    print ( A[r][c])
    c--
}
while ( r > 0) {                        ↑
    print ( A[r][c])
    r--
}
```

$TC = O(4 * (N-1)) \rightarrow \underline{O(N)}$

$SC = \underline{O(1)}$

Spiral Matrix

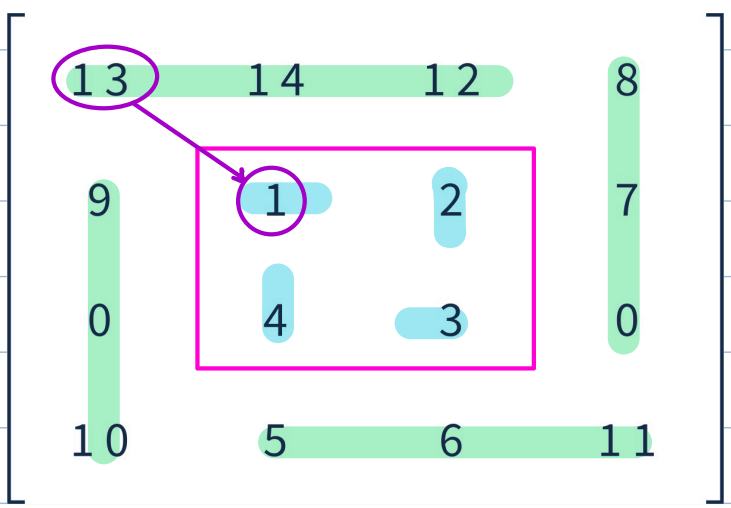
mat[N][N]

	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

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 :



o/p → 13 14 12 8 7 0

11 6 5 10 0 9

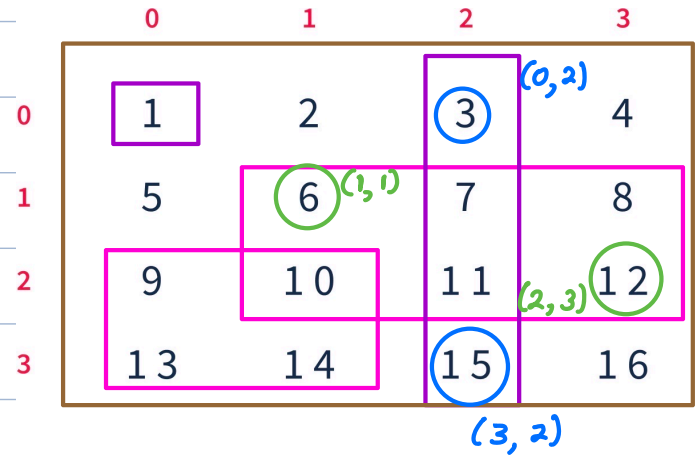
1 2 3 4

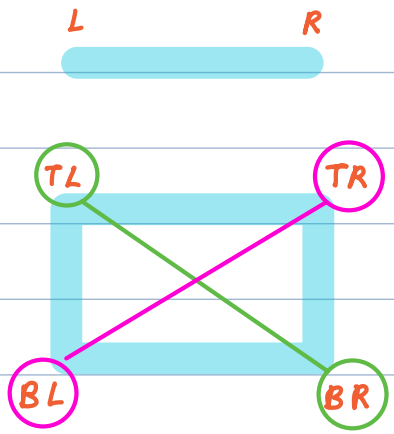
```
r = 0    c = 0
while ( N > 1 ) {
    for (i → 1 to (N-1)) {          →
        print ( A[r][c] )
        c++
    }
    for (i → 1 to (N-1)) {          ↓
        print ( A[r][c] )
        r++
    }
    for (i → 1 to (N-1)) {          ←
        print ( A[r][c] )
        c--
    }
    for (i → 1 to (N-1)) {          ↑
        print ( A[r][c] )
        r--
    }
    r++
    c++
    N -= 2
}
if (N == 1)  print ( A[r][c] )
```

TC = $O(N^2)$ SC = $O(1)$

Sub - Matrices

- Contiguous part of a matrix





< *Question* > : Given mat[N] [M]. Find sum of all sub-matrix sums.

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

2 * 3

[4] 4 [4 9] 13 [4 / 5] 9

[5] 5 [9 6] 15

[9] 9 [5 -1] 4 [9 / -1] 8

[-1] -1 [-1 2] 1

[6] 6 [4 9 6] 19 [6 / 2] 8

[2] 2 [5 -1 2] 6

subarray = $\frac{N*(N+1)}{2}$

[4 9 / 5 -1] 17 [9 6 / -1 2] 16 [4 9 6 / 5 -1 2] 25

submatrix = $\frac{N*(N+1)}{2} * \frac{M*(M+1)}{2}$

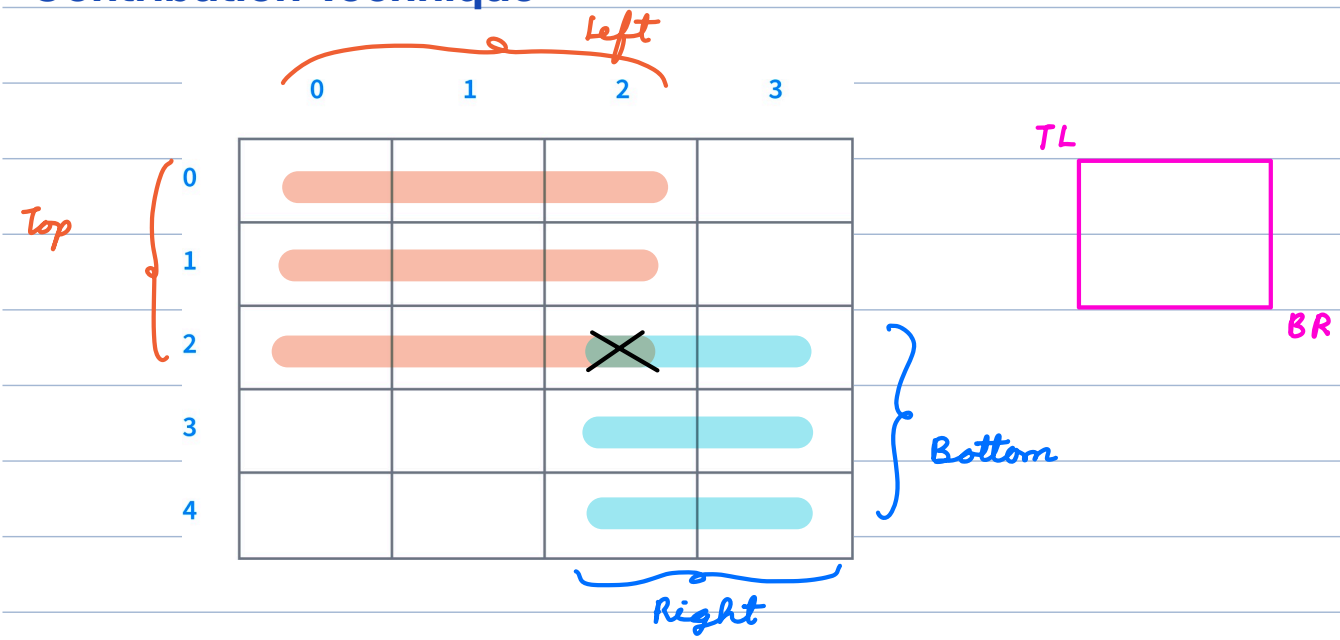
Ans = 166

2 * 3 ⟶ 3 * 6 = 18

Contribution Technique → Ans = $\sum_{i,j}$ contribution of A[i][j]

A[i][j] * (# submatrix A[i][j] is part of)

Contribution Technique



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

9 * 6 = 54

i , j

Top → [0 i] → (i + 1)

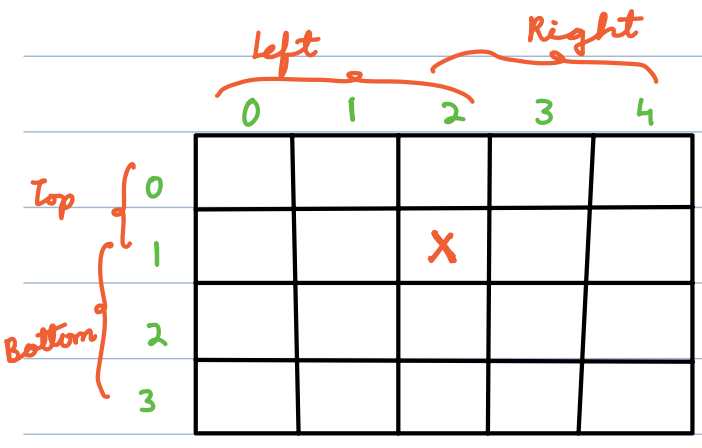
Left → [0 j] → (j + 1)

Bottom → [i N-1] → (N - i)

Right → [j M-1] → (M - j)

```
ans = 0
for i → 0 to (N-1) {
    for j → 0 to (M-1) {
        ans += A[i][j] * (i+1) * (j+1) * (N-i) * (M-j)
    }
}
return ans
```

TC = O(N * M) SC = O(1)



Top → 2

Left → 3

Bottom → 3

Right → 3

2 * 3 * 3 * 3

= 6 * 9

= 54