Array : Two Dimensional

Today's Agenda:

1) Search in row wise and col wise sorted matrix.

2) Row with maximum ones.

3) Printing boundary of a matrix.

4) Spiral matrix.

5) Sum of all submatrix sum.

Q.1) Given a row wise and column wise sorted matrix. Check whether element K is present or not.

	0	1	2	3
0	-5	-2	1	13
1	-4	0	3	14
2	-3	2	6	18

K	Output
13	True
0	True
5	False

Brute Force: Iterate over every cell.

Quiz: $O(N*M)$

Optimisation:

Searching → $\overset{K}{0}$

	0	1	2	3
0	-5	-2	1	13
1	-4	0	3	14
2	-3	2	6	18

$13 > 0$

$1 > 0$

$-2 < 0$

True

$K = -3$

	0	1	2	3
0	-5	-2	1	13
1	-4	0	3	14
2	-3	2	6	18

$13 > -3$

$1 > -3$

$-2 > -3$

$-5 < -3$

$-4 < -3$

$\underline{\underline{3*4}}$ $\underline{\underline{7}}$

Approach:

1) Start at top right corner.

2) If A[row][col] < K, move ↓ (reject row)
   else move ← (reject col)

3) Repeat untill element is found or (row or col) goes out of bound.

Code:

```
row = 0 , Col = M-1
while( row<N && col >=0)
{
    if(A[row][col] == K)
    { return True;
    }
    else if ( A[row][col] < K)
    {   row++;
    }
    else
    {
        col--
    }
}
return False;
```

TC: $O(N+M)$
SC: $O(1)$

Q.2) Given a binary sorted matrix A of size N*N. Find the row with maximum number of ones.

Note: 1) If two rows have same no. of ones, return the one having lower index.

2) Every row is sorted.

Eg.

$$\begin{array}{c|ccc|l}
 & 0 & 1 & 2 & \\
\hline
0 & 0 & 1 & 1 & \rightarrow 2 \\
1 & 0 & 0 & 1 & \rightarrow 1 \\
2 & 0 & 1 & 1 & \rightarrow 2 \\
\end{array}$$

Ans = 0

Eg. 2

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

Ans = 3

Quiz

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

Ans = 2

Optimisation:

ansIndex = ~~0~~ ~~x~~ 4

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

0	0	1	1
0	1	1	1
0	1	1	1
1	1	1	1

Algorithm:

1) Start at row = 0, col = m−1

2) If one is present, move left by decrementing col and update ans to current row.

3) If zero is present, move down by incrementing row.

```
row=0 , col = M-1; ansIndex = -1;
while(row<n && col>=0)
{
    if(A[row][col] == 1)
    {
        ansIndex = row;
        col--;
    }
    else
    {
        row++;
    }
}
return ansIndex
```

TC : O(N+M)

SC : O(1)

Q.3) Given a matrix of N*N. Print boundary elements in clockwise direction.

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

Output: 1, 2, 3, 4, 8, 12, 16, 15, 14, 13, 9, 5

Approach:

Print N-1 elements in each:-

a) first row →
b) last column ↓
c) last row ←
d) first column ↑

```
void printBoundaryElements(int[][] A)
{
    row=0; col=0;

    1) N-1 in the first row →
    for(cnt=0; cnt<N-1; cnt++)
    {
        Print(A[row][col]);
        col++;
    }
    2) N-1 in the last column ↓
    for(cnt=0; cnt<N-1; cnt++)
    {
        Print(A[row][col]);
        row++;
    }
    3) N-1 in the last row ←
    for(cnt=0; cnt<N-1; cnt++)
    {
        Print(A[row][col]),
        col--;
    }
    4) N-1 in the first column ↑
    for(cnt=0; cnt<N-1; cnt++)
    {
        Print(A[row][col]),
        row--;
```

TC: O(N)

SC: O(1)

y y

Q.4) Given an integer A, generate a square matrix of size A*A filled with elements in spiral form 1 to A^2.

A→4

4*4 = 16

Eg. A=4

Output:

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

Eg. A=6

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

row	col	A
0	0	6
		−2
1	1	4
		−2
2	2	2
3	3	0

```
int[A][A];   N=A;
row=0; col=0; int count=0;
while(A>1 )
{   1) N-1 in the first row →
    for(cnt=0; cnt<N-1; cnt++)
    {       A[row][col]=++count;
            col++;
    }
    2) N-1 in the last column ↓
     for(cnt=0; cnt<N-1; cnt++]
     {    A[row][col] = ++count;
            row++;
     }
    3) N-1 in the last row ←
     for(cnt=0; cnt<N-1; cnt++]
     {      A[row][col] = ++count;
            col--;
     }
    4) N-1 in the first column ↑
      for(cnt=0; cnt<N-1; cnt++]
      {     A[row][col]=++count;
            row--;
      }
```

```
}
row++; col++;
A = A-2;
}
if(A==1)
{
A[row][col] = ++count;
}
}
```

$A = 5$
3
1

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

row = ~~0~~ ~~1~~ 2
col = ~~0~~ ~~1~~ 2

++count = count+1;
Preincrement

TC: $O(A^2)$
SC: $O(1)$

Break: 10.35 PM

Submatrices

→ Same concept as a subarray.
→ A continuous part of 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

Uniquely identify a rectangle

Q.5) Given a matrix, find the sum of all submatrix sum.

$$\begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & 4 & 9 & 6 \\ 1 & 5 & -1 & 2 \end{array}$$

All Submatrices

$$\begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & 4 & 9 & 6 \\ 1 & 5 & -1 & 2 \end{array}$$

1*1

4 → 4
9 → 9
6 → 6
5 → 5
-1 → -1
2 → 2

1*2

4, 9 → 13
9, 6 → 15
5, -1 → 4
-1, 2 → 1

1*3

4, 9, 6 → 19
5, -1, 2 → 6

2*1

$\left.\begin{matrix}4\\5\end{matrix}\right\} \rightarrow 9$

$\left.\begin{matrix}9\\-1\end{matrix}\right\} \rightarrow 8$

$\left.\begin{matrix}6\\2\end{matrix}\right\} \rightarrow 8$

2*2

$\left.\begin{matrix}4 & 9\\5 & -1\end{matrix}\right\} \rightarrow 17$

$\left.\begin{matrix}9 & 6\\ \ldots\end{matrix}\right\} \rightarrow 16$

2*3

25

166

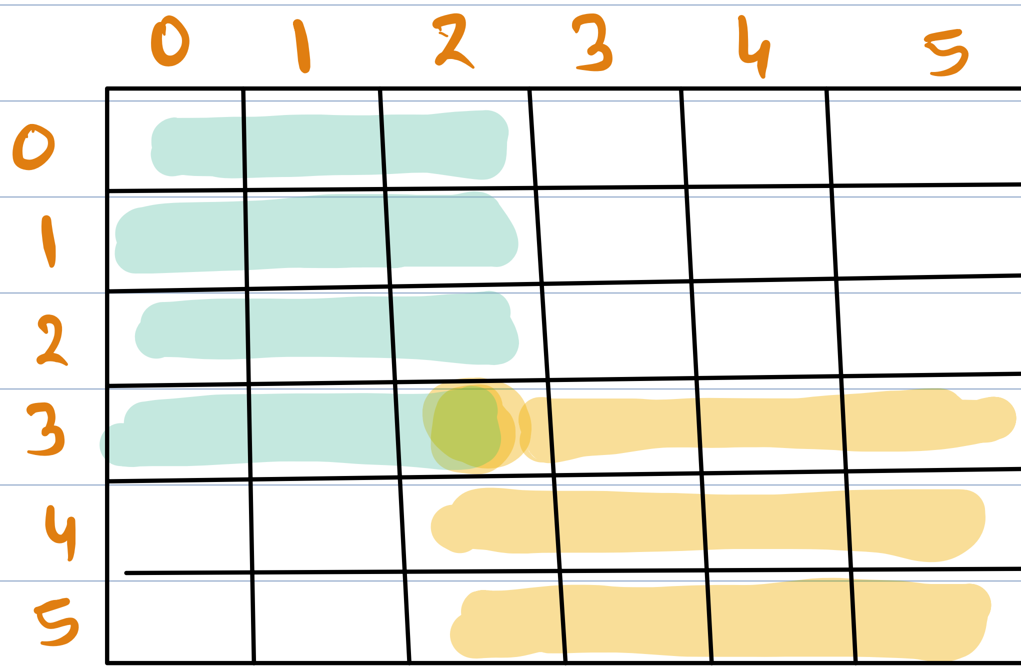
-1 2

Total submatrix = $O(n^4)$

Iterate over submatrix = $O(n^2)$

$TC = O(n^6)$

Approach:



(3, 2)

$12 * 12 = \underline{144}$

Quiz:

(1,2)

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

$6 * 12 = 72$

$[i \quad N-1]$
$= N - \cancel{1} - i + \cancel{1}$
$= N - i$

$[j \quad m-1]$
$= m - \cancel{1} - j + \cancel{1}$
$= m - j$

$$[0 \quad j] = j - 0 + 1 = j+1$$
$$[0 \quad i] = i - 0 + 1 = i+1$$

$$TL \rightarrow (i+1) * (j+1)$$
$$BR \rightarrow (m-j) * (N-i)$$

Contribution of element =
$$(i+1) * (j+1) * (m-j) * (N-i) * A[i][j]$$

```
ans = 0

for(i=0; i< N; i++)
{
    for(j=0; j< N; j++)
    {
        ans = ans + ((i+1) * (j+1) *
              (M-j) * (N-i) * A[i][j]);
    }
}

return ans;
```

no. of cols → $M = 3$
no. of rows → $N = 2$

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

ans = ~~0~~ ~~24~~ ~~96~~ ~~132~~ ~~162~~ ~~154~~ 166

i	j
0	~~0~~ ~~1~~ ~~2~~ 3
1	~~0~~ ~~1~~ 2

TC: $O(N*M)$