Arrays 2 : Two Dimensional

Question 1

Given a row-wise & column-wise sorted matrix, find out whether element K is present or not.

A =

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

K = 13 ans = true

K = 0 ans = true

K = -1 ans = false

Brute force → for i,j check a[i][j] TC: O(N×M)

Optional

If binary search

TC: O(N log M)

or

O(M log N)

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

K = 0

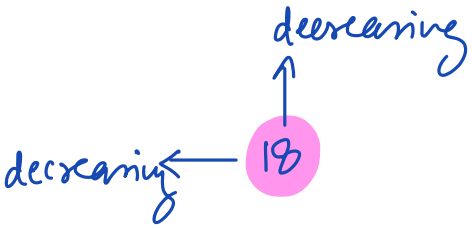
-5 < 0

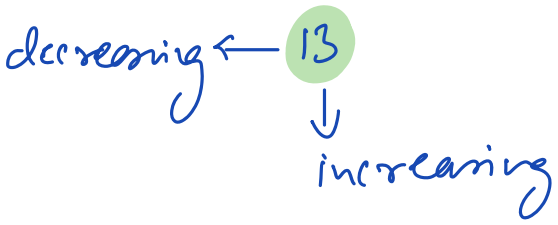
⇒ go to large values

-5 → increasing

↓

increasing





18 > 0

⇒ go to smaller element

	0	1	2	3
0	-5	-2	1	13
1	-4	0	3	14
2	-3	2	6	12

K = 0

13 > 0

⇒ go to left

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

1 > 0

⇒ go to left

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

-2 < 0

⇒ go to down

	0	1
1	-4	0
2	-3	2

FOUND !!

return true

Code

```
i=0, j=m-1      //top right (OR bottom left)

while( i<n && j>=0) {

    if (a[i][j] == K) return true

    else if (a[i][j] > K)   j--      // left

    else    i++     // down

}

return false
```

i → 0 to n-1
N times

j → m-1 to 0
M times

TC : O(N+M)

SC : O(1)

Question 2

Given a binary (0/1) matrix sorted row-wise. 000... 111...

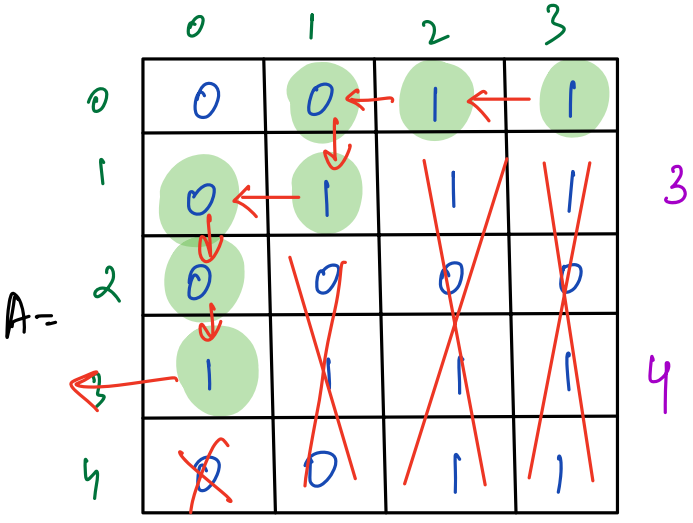
Find the smallest row index with max # 1's.

A =

	0	1	2
0	0	1	1
1	0	0	1
2	1	1	1

ans = 2 [answer is index]

Brute force - count # 1's in each row TC : O(N*M)
SC : O(1)



Code

```
i=0, j=m-1 , ans=0

while( i<n && j>=0) {

    while(j>=0 && a[i][j]==1) {
        j--  // left
        ans = i
    }

    i++ // down
}

return ans
```

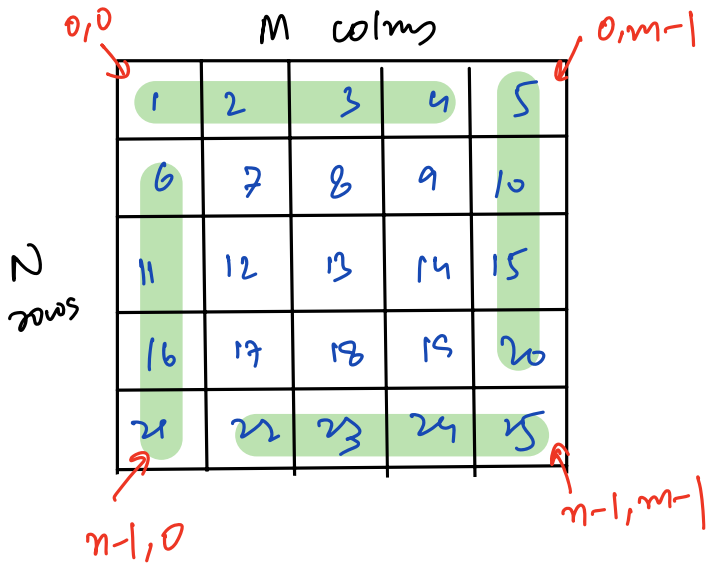
nested loop
⇒ O(N*M)

TC : O(N+M)

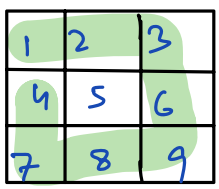
SC : O(1)

Question-3

Given the matrix, print boundary elements clockwise.



output : 1 2 3 4 5 10 15 20 25 24 23 22 21 16 11 6



output : 1 2 3 6 9 8 7 4

Code

```
i=0, j=0

for (k = 0 to m-2) {        → m-1 times      i=0, j=0
    print (a[i][j])                          first row L→R
    j++ //right
}

for (k = 0 to n-2) {        → n-1 times      i=0, j=m-1
    print (a[i][j])                          last column, top ↓ down
    i++ //down
}
```

```
for (k = 0 to m-2) {        → m-1 times
    print (a[i][j])
    j--  //left
}
for (k = 0 to n-2) {        → N-1 times
    print (a[i][j])
    i--  //up
}
```

i = n-1, j = m-1 → last row L ← R

i = n-1, j = 0 → first colum, top ↑ bottom

$= m-1 + N-1 + m-1 + N-1$

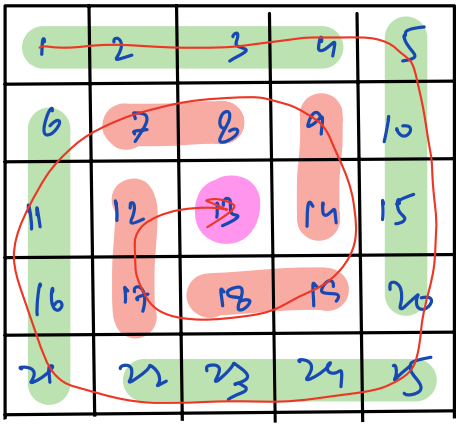
TC : O(N+M)

SC : O(1)

Question

Print elements in spiral order in clock wise direction. (square matrix) → N×N

A=



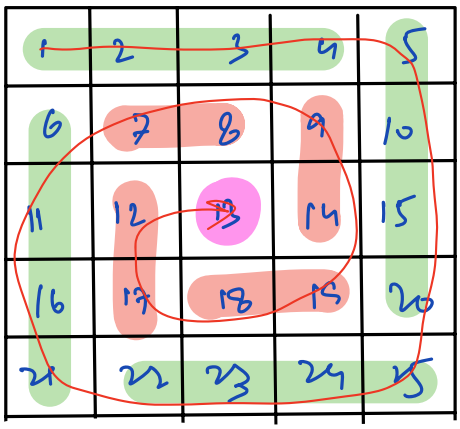
output : 1 2 3 4 5 10 15 20 25

24 23 22 21 16 11 6

7 8 9 14 19 18 17

12 13

N=5



r	c	N
0	0	5
1	1	3
2	2	1

+1, +1, -2, -2

r=0, c=0

```
while ( n > 1 ) {
    i or j = c

    for (k = 0 to n-2) {
        print (a[i][j])
        j++ //right
    }
    for (k = 0 to n-2) {
        print (a[i][j])
        i++ //down
    }

    for (k = 0 to n-2) {
        print (a[i][j])
        j-- //left
    }
```

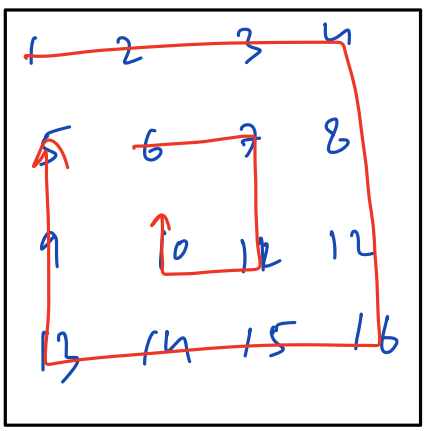
because square matrix, no M

TC: $O(N^2)$

SC: $O(1)$

```
for (k=0 to n-2) {
    print (a[i][j])
    i-- // up
}
r++, c++, n-=2
}
if (n==1)
    print (a[r][c])
```

→ executed if n is odd

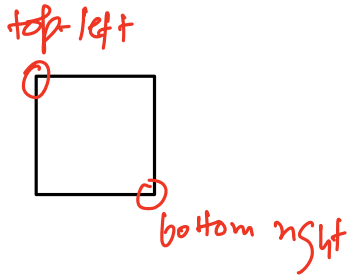


Submatrix

A:

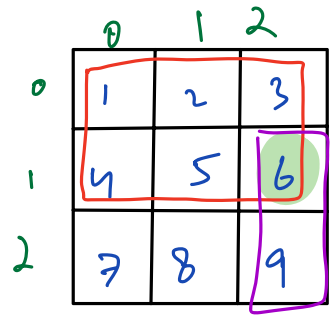
1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

→ submatrix



Question

find the # submatrix in which a[i][j] is present for given i,j.



(i,j) = (1,2)

ans = 12

$2 \times 3 \times (3-1) \times (3-2)$

$2 \times 3 \times 2 \times 1$

$= 12$

1 2 3 2 3 3

4 5 6 5 6 6

4 5 6 5 6 6

7 8 9 8 9 9

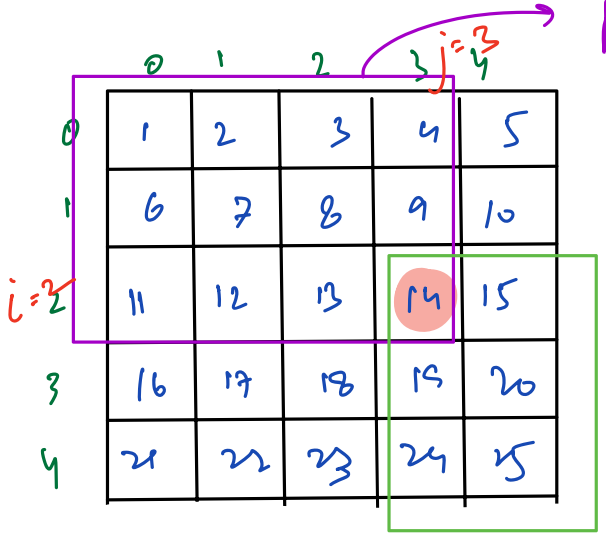
4 5 6 5 6 6

7 8 9 8 9 9

1 2 3 2 3 3

4 5 6 5 6 6

7 8 9 8 9 9



possible top-left cells = $(i+1) \times (j+1)$

bottom-right cells = $(n-i) \times (m-j)$

submatrix containing a[i][j] =

$= (i+1)*(j+1) \times (n-i)*(m-j)$

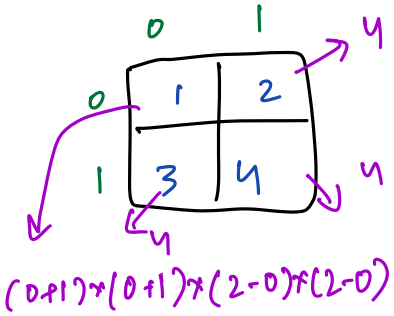
Question 5

find sum of all submatrix sum.

```
am=0
for(i=0 to N-1){
    for(j=0 to m-1){
        aus += (i+1)*(j+1)*(n-i)*(m-j)*a[i][j]
    }
}
return am
```

TC: O(N*m)

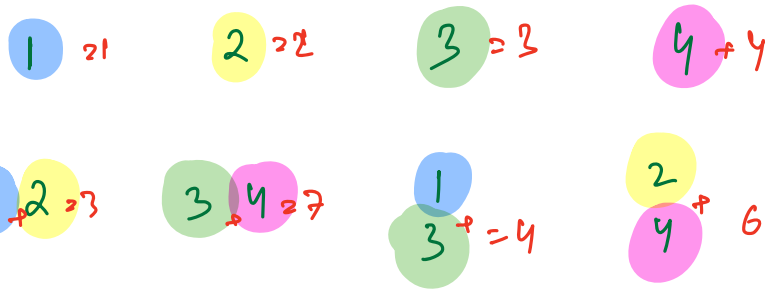
SC: O(1)

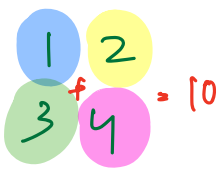


(0+1)*(0+1)*(2-0)*(2-0)

1*1*2*2 = 4

contribution





am = 40

$= 1x4 + 2x4 + 3x4 + 4x4 = 40$