2D array

Matrix in Maths

	0	1	2	3
0	00	01	02	03
1	10	11	12	13
2	20	21	22	23
3	30	31	32	33

3×4

4×4

[ij] i → row
j → column

```
int[][] arr = new int[3][4];
```

4×3×4 = 48 bytes

No. of elements * size of 1 Element = Memory allocated

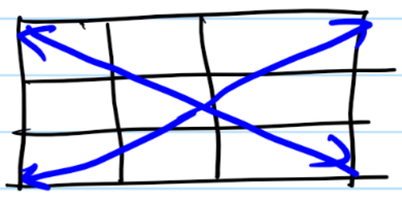
arr[2][3]

Square Matrix

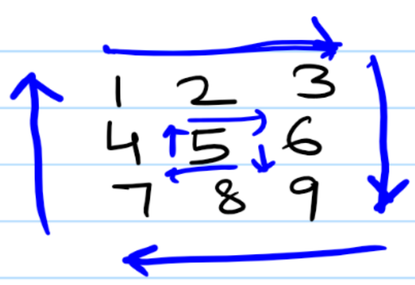
→ |rows| == |columns|

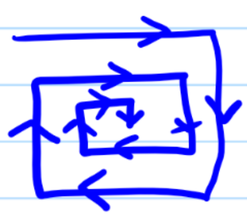
4×4 3×3 . . . n×m

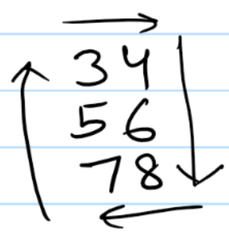
Rectangular Matrix n×m n ≠ m



Q) Print matrix in spiral order

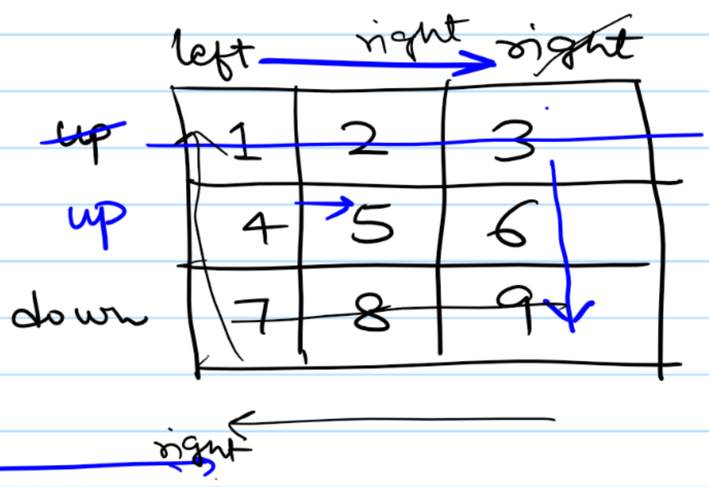






346875

1 2 3 6 9 8 7 4 5



left → right

up ~~top~~ 1 2 3 4 = 1 2 3 4 4 3 2 1

down

up = 1
down = 0

right = 3
left = 0

if (left <= right) {

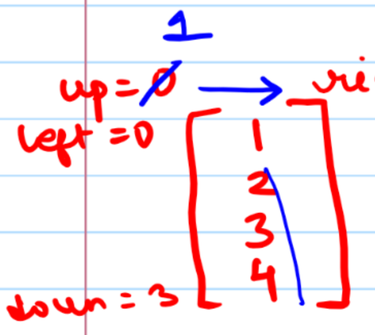
1 2 3 6 9 8 7 4 5

L2R
```
for (int col = left; col <= right; col++)
{
    print arr[up][col];
}
up++;
```

T2B
```
for (int row = up; row <= down; row++)
{
    print (arr[row][right]);
```

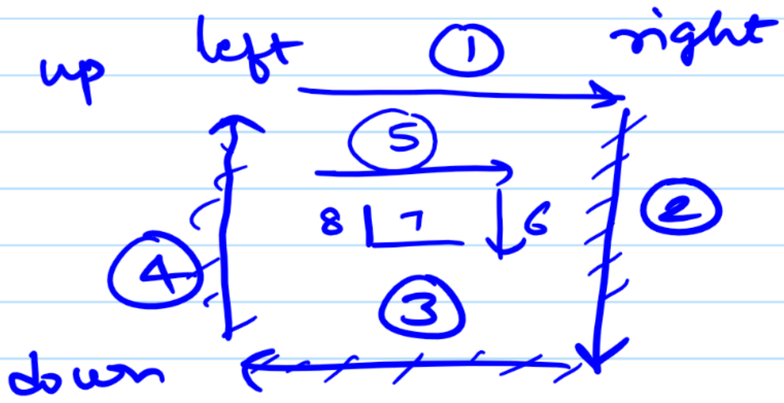
```
for( int row = down; row>=up; row--)
{ print ( arr[row][left]);
}
left++;
}
```

```
right--;
if(up<=down)
{
for( int col = right ; col>=left; col--)
{
print ( arr[down][col];
}
down--;
}
```



1234

1234

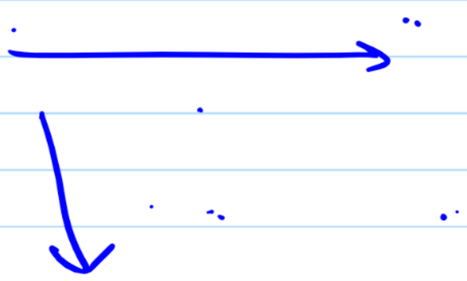


left to right
up++

top to bottom
right--;

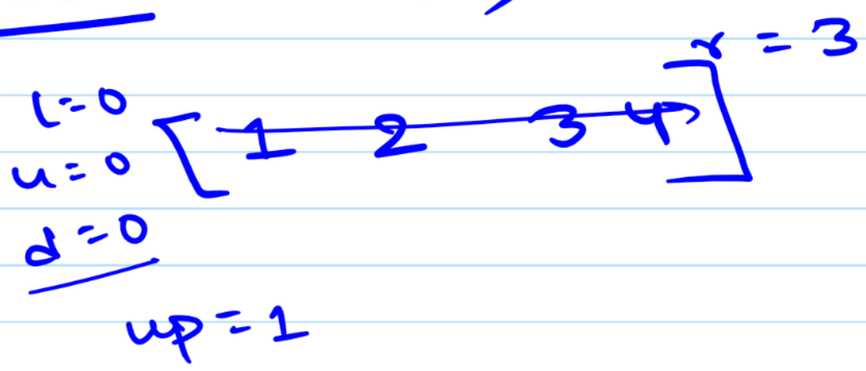
if( )
right to left;
down--;

if( )
down to up
left++;



TC: M rows N col O(MXN)

SC: O(1)

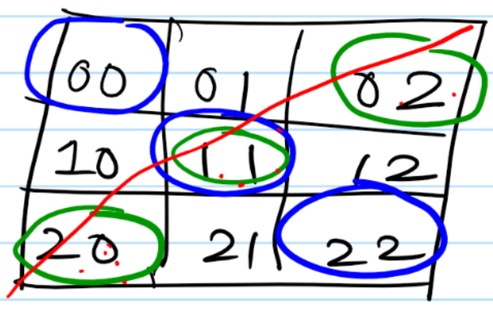


1234
4321 if(u<=d)

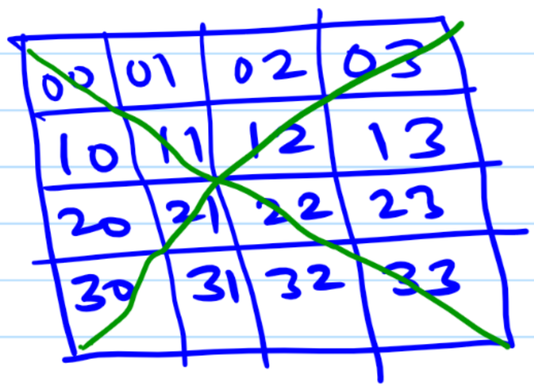
Q2 Matrix diagonal sum

= 2

3x3



nxn



i = j

i+j = n-1

j = n-1-i

mat[3/2][3/2]
mat[1][1]

```
{
ans = 0;
for( int i=0; i<n; i++)
{
ans += mat[i][i]; // adding primary diagonal elements
ans += mat[i][n-1-i]; // adding 2ndary diagonal elements
}
if(n%2==1) // odd order matrix
```

Duplicate element

```
{   ans -= mat[n/2][n/2];
    }
return ans;
}
```

$3/2 = 1$

TC: O(N)
SC: O(1)

Q3: count negative numbers

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

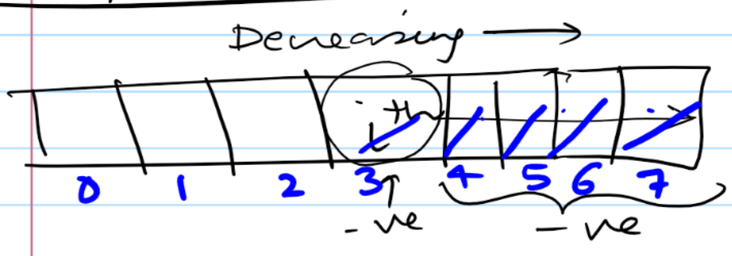
rows & column are in decreasing order

Brute force:

```
count = 0;
for( i=0 ... m-1)
{
  for( j=0 ... n-1)
  {
    if mat[i][j]<0
      count++;
  }
return count
```

TC: O(m×n)
SC: O(1)

Approach 2: Hint: **Binary Search**



$(n - i) = 8 - 3 = 5$

left = 0	1	2	3	4	5	~~right = 6~~
10	4	4	0	-1	-1	100
				↑ left		
				↑ right		

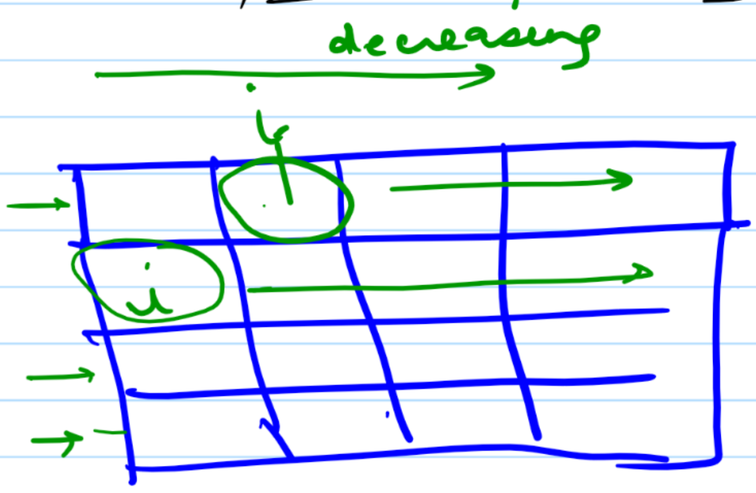
mid = left + right/2
= 0+6/2 = 3

left = mid + 1 = 4

mid = 4+6/2 = 5

right = 4

n n/2 . . . . 1 ($\log n$)



Algo:
1) Initialise count = 0 ← Total -ve elements
2) Let n be no. of columns, m = rows
Iterate on each row of the matrix &

find the index of 1st -ve element (left).

3) all element from left to n will be -ve : row is in decreasing order.

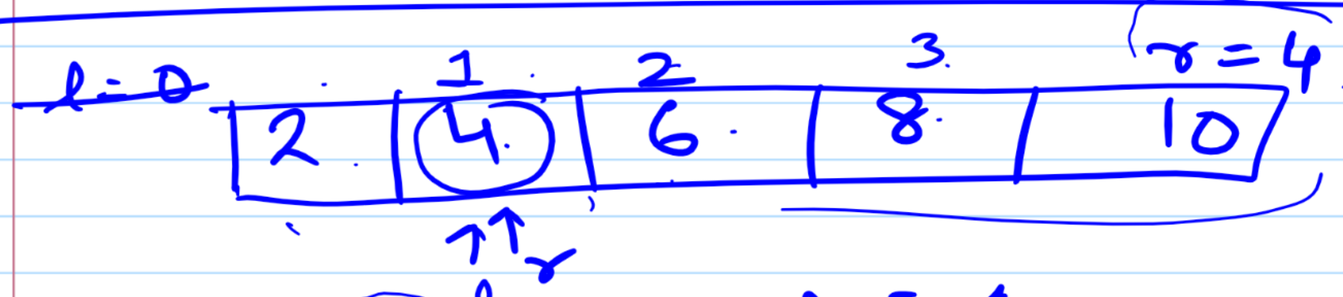
Increment count by (n - left)

TC: n elements in each row

BS: O(log n)

m * log n = O(m log n)

SC: O(1)



4 is in array?

mid = $\frac{0+4}{2}$ = 2 l r = 1

True

mid = $\frac{0+1}{2} = \frac{1}{2} = 0$

m = $\frac{1+1}{2} = 1$

Q4: Richest customer wealth (M customers N accounts)

[1 2 8
4 1 2]

1st = 1+2+8 = 11 1st

2nd = 4+1+2 = 7

Algorithm: 1) Calculate row sum for each row

2) Return row with max. sum

TC: O(M * N)

SC: O(1