Array: 2D Matrices

2D Matrix → 2D Array

rectangular grid of values

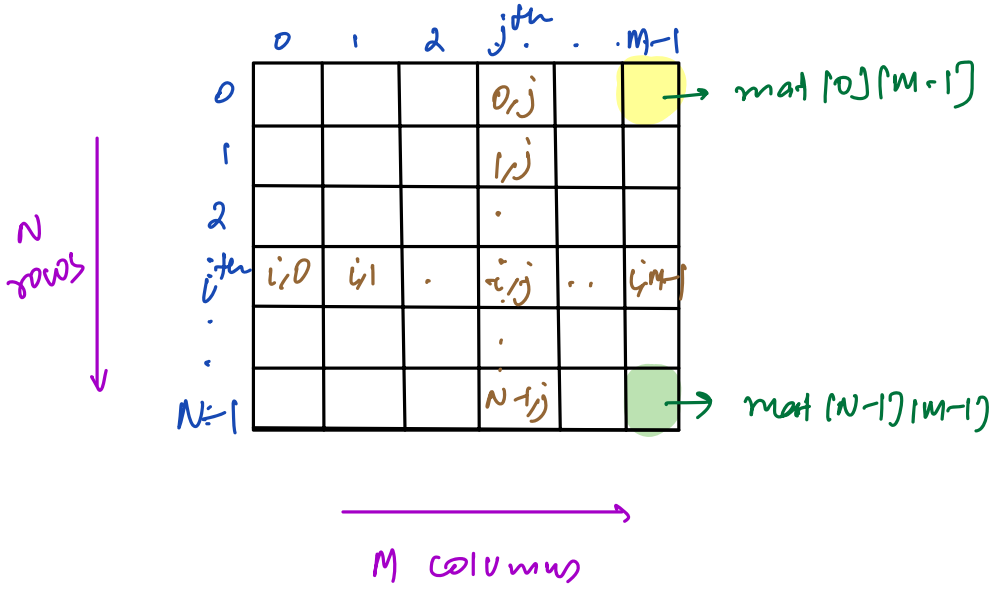
Declare

```
int mat[N][M]
```

M columns

N rows

```
int[][] mat = new int[N][M]
```



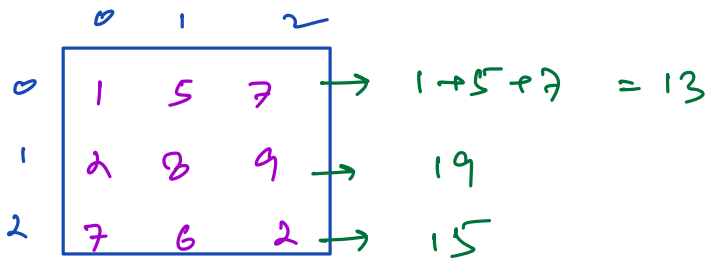
Observation

→ If we move in i^{th} row, row no. is constant but column will change → [0, M-1)

→ If we move in j^{th} column, column no. is constant but row will change → [0, N-1)

Question-1

Given mat[N][M], print row-wise sum.



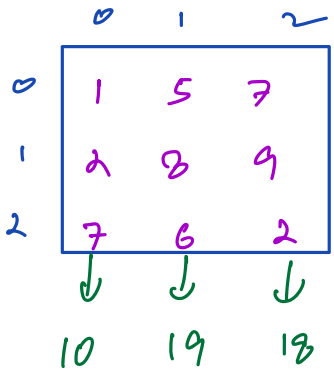

```
def sumRow (mat[N][M]) {

    for( i=0 to N-1) {
        // ith row
        sum=0
        for( j = 0 to M-1) {
            sum += mat[i][j]
        }
        print(sum)
    }
}
```

TC = O(N*M)

SC = O(1)

Question 2

Given mat(N)(M), print column wise sum.



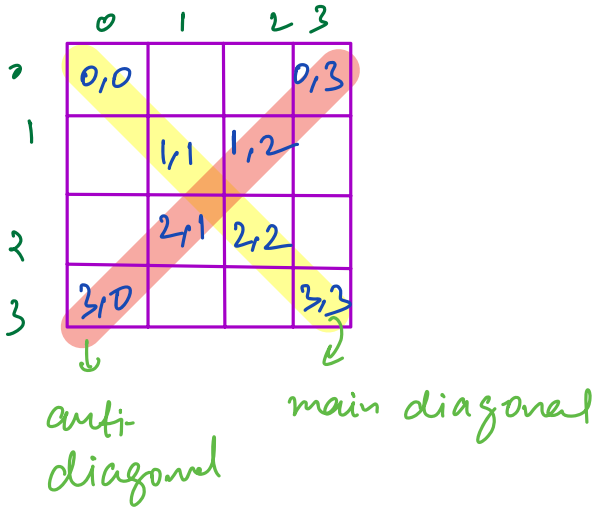

```
for(j=0 to M-1){
    sum=0
    for(i=0 to N-1){
        sum += mat[i][j]
    }
    print(sum)
}
```

TC = O(N*M)

SC = O(1)

Question 3

Given a square matrix mat[N][N], print diagonals.




```
for(i=0 to n-1){
    for(j=0 to n-1){
        if(i==j)
            print(mat[i][j])
    }
}
```

TC = O(N²)

Main Diagonal :

```
for(i=0 to N-1){
    print(mat[i][i])
}
```

TC = O(N)

SC = O(1)

Anti Diagonal

$i + j = n-1 \Rightarrow j = n-1-i$

```
for(i=0 to N-1){
    print(mat[i][n-1-i])
}
```

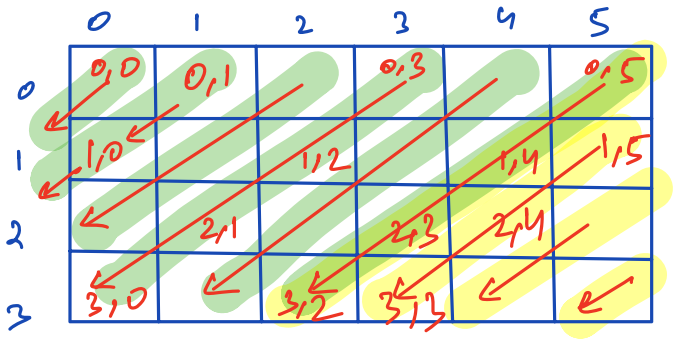
```
i=0, j=N-1
while(i<N){
    print(mat[i][j])
    i++, j--
}
```

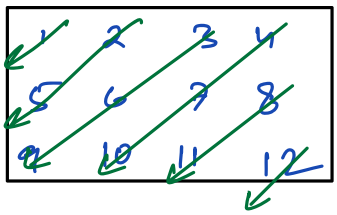
TC = O(N) SC = O(1)

Question

Given mat[N][M], print all diagonals going from right to left.

Diagonals should start from 0^{th} row OR $m-1^{th}$ col..





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

M diagonals from 0^{th} row

N diagonals from $M-1^{th}$ col.

1 overlap

⇒ M + N − 1

```
printDiagonals( mat[N][M] ) {

// print all diagonals from 0th row

for( j=0 to M-1 ) {

    r=0, c=j

    while( r<N && c>=0 ) {

        print (mat[r][c])

        r++, c--

    }

    print (newline)

}

// print all diagonals from m-1th col.

for( i=1 to N-1 ) {

    r=i, c= M-1

    while( r<N && c>=0 ) {

        print (mat[r][c])

        r++, c--

    }

    print (newline)

}
```

j	iterations
0	1
1	2
2	3
⋮	⋮
M-1	M

$\frac{M(M+1)}{2}$

i	iterations
1	N-1
2	N-2
⋮	
N-1	1

$\frac{N(N-1)}{2}$

$TC = O(N \times M)$

$SC = O(1)$

Question 5

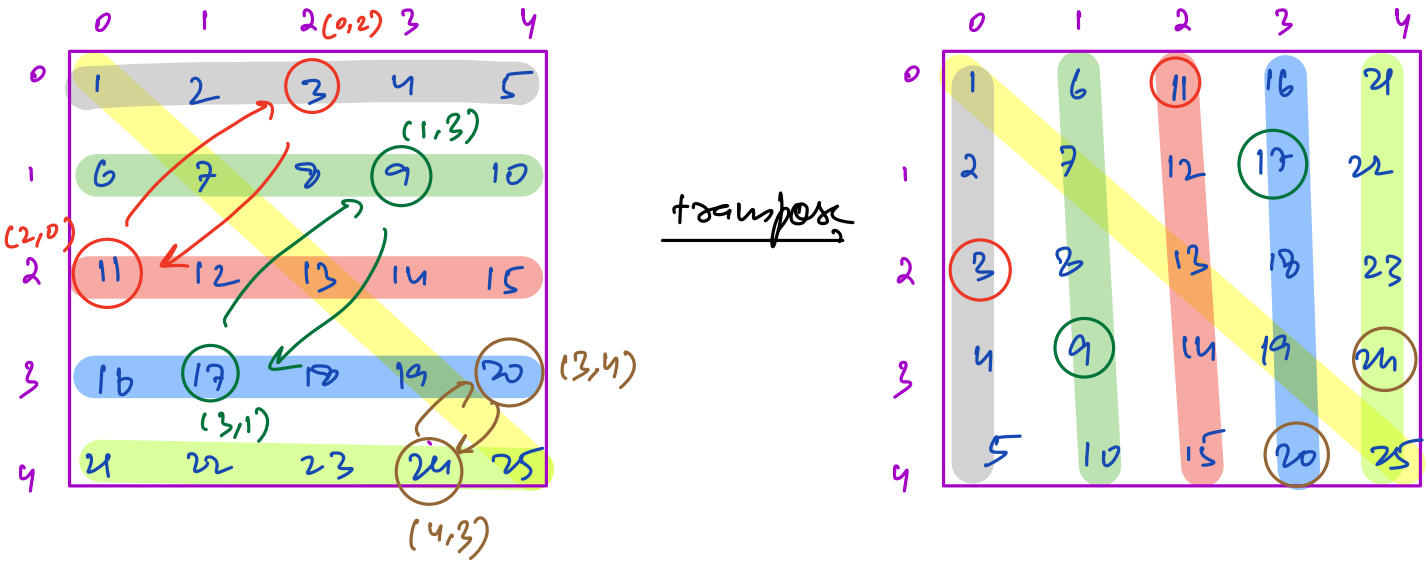
Given mat[N][N], calculate transpose of the matrix w/o extra space

Transpose:

0^{th} row ⟶ 0^{th} col

1^{st} row ⟶ 1^{st} col

⋮



[i,j] ⟷ [j,i]

[i,i] ⟷ [i,i]

```
for (i=0 to n-1) {
    for (j=0 to n-1) {
        swap(mat[i][j], mat[j][i])
    }
}
```

✗ will do double swap

i=0, j=1

[0,1] ⟷ [1,0]

i=1, j=0

[1,0] ⟷ [0,1]

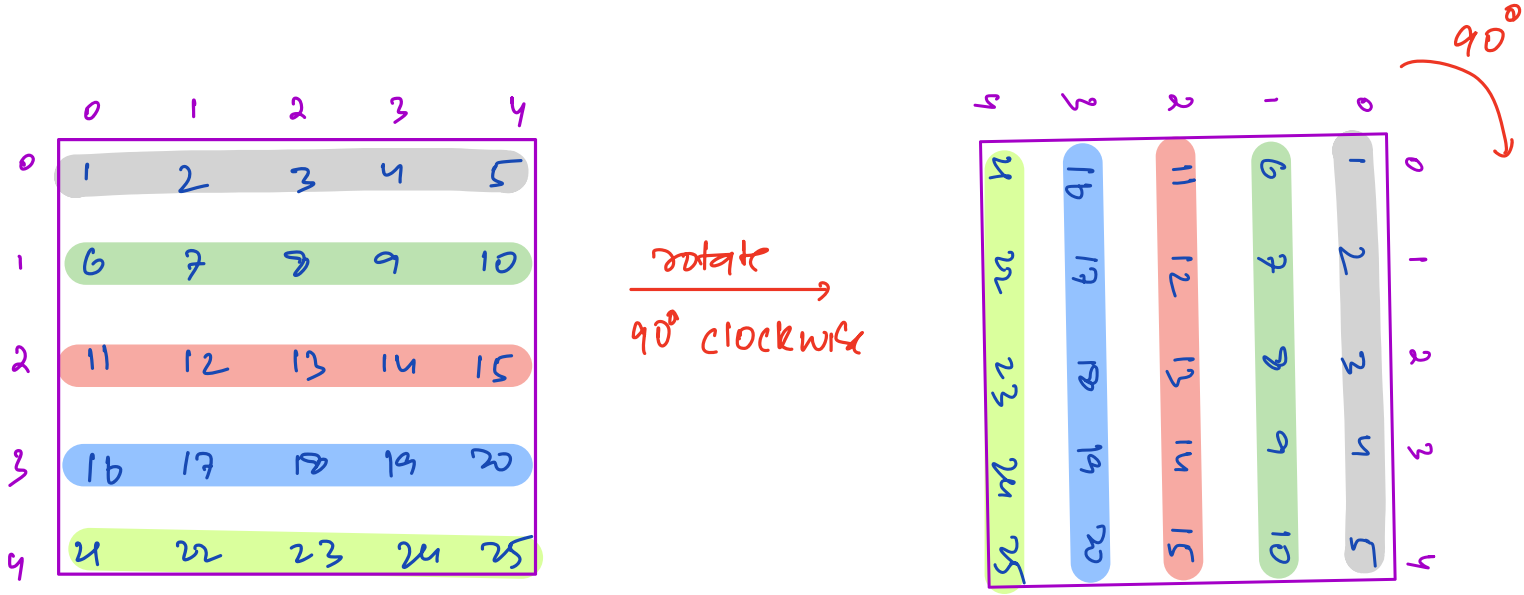
```
for (i=0 to n-1) {
    for (j=i+1 to n-1) {
        swap(mat[i][j], mat[j][i])
    }
}
```

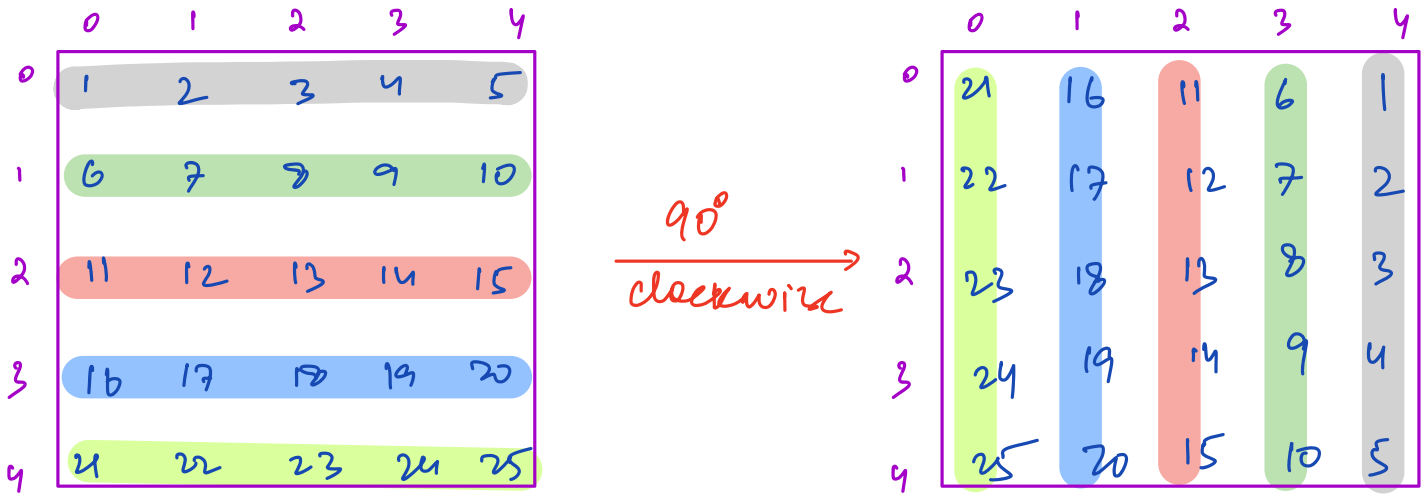
TC = O(N²)

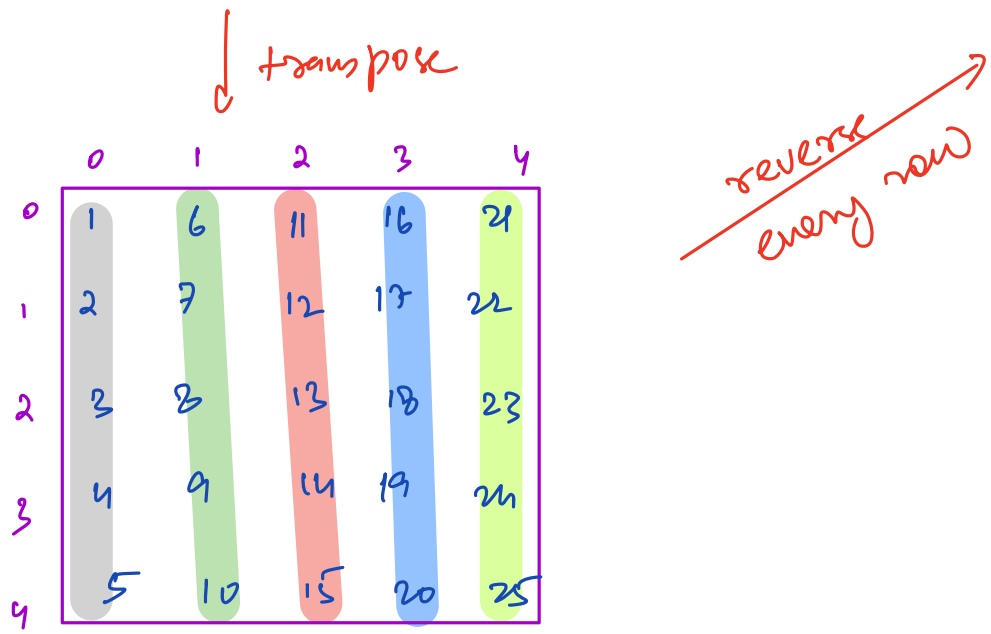
SC = O(1)

Question 6

Given mat[N][N], rotate 90° clockwise from top-right. SC = O(1)








```
rotate ( mat[N][N] ) {
    mat = transpose (mat)  ⟶ TODO → O(N²)
    for ( i=0 to n-1 ) {
        reverse (mat[i])  ⟶ TODO → O(N)
    }
}
```

like reversing an array

TC = O(N²)

SC = O(1)