2D Matrices

Vinay Neekhra

Senior Instructor & Mentor

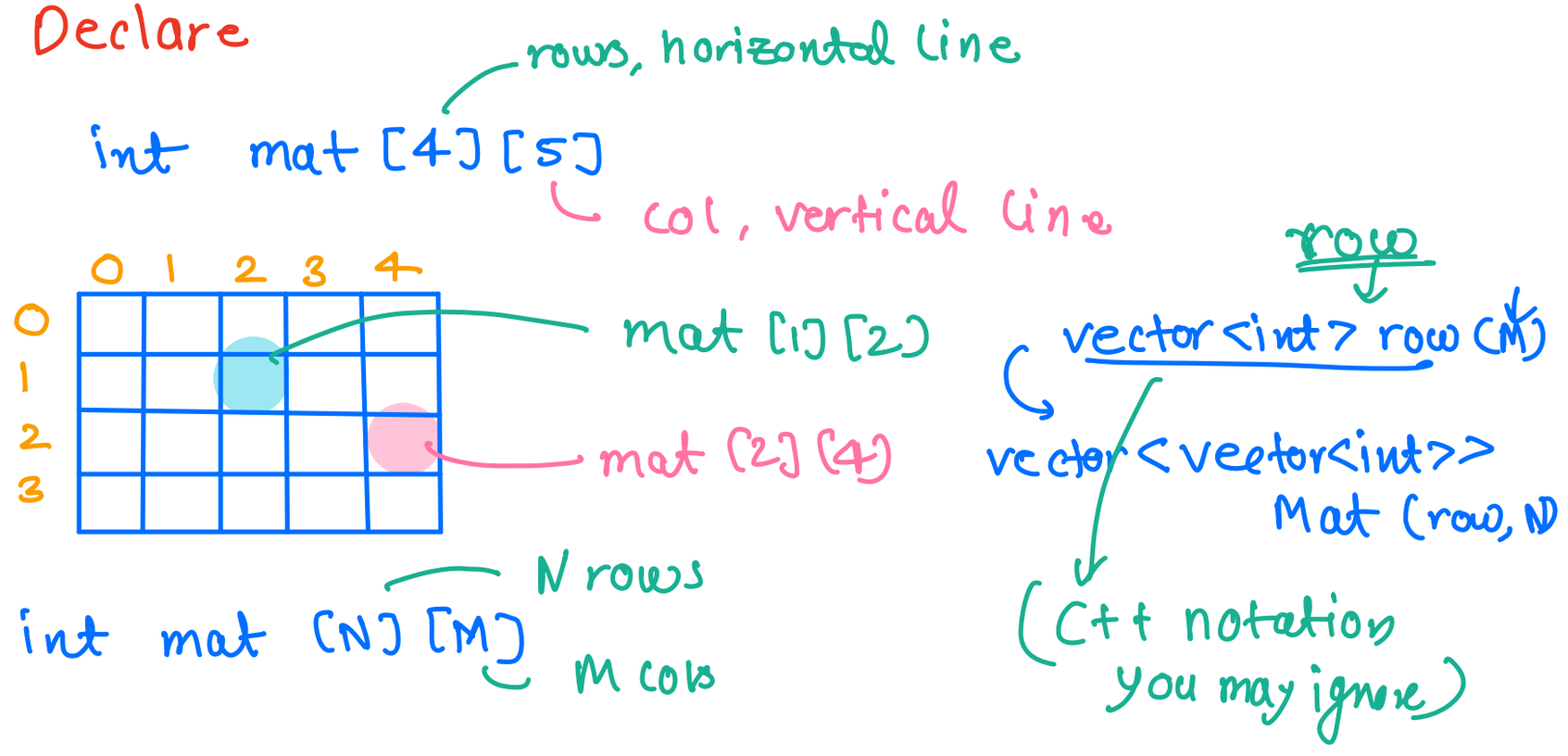
Reachable in Scaler Lounge

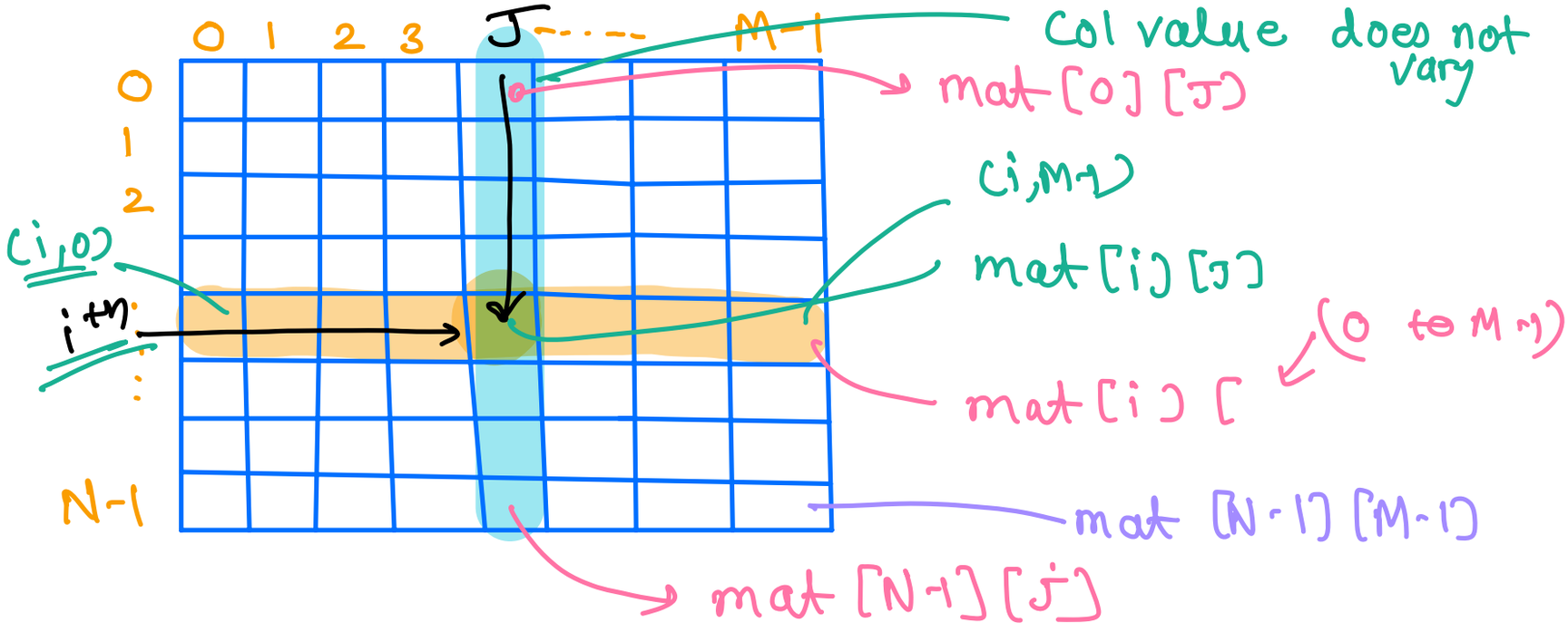
"Computer Science is like a matrix;
it's all about how you manipulate the numbers"

Today's content

1) Basics
2) problems.

§. Declare





Obsn:

① if we iterate on a row, col no. changes from [0, M-1]

② if we iterate on a col, row no. changes from [0, N-1]

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

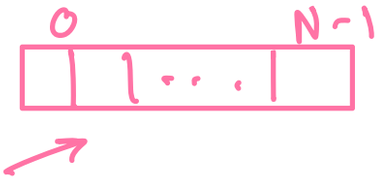
ex.

	0	1	2	3	4	Prints
0	2	-1	4	7	8	20
1	1	10	20	5	7	43
2	8	7	5	3	2	25
3	1	3	2	4	10	20

```
void rowWisePrint (int mat[][]){
    int N = mat.length
    int M = mat[0].length
    for(i=0; i<N; i++){
        sum=0
        // we need ith row's sum
        for (j=0; j<M; j++){
            sum += mat[i][j]
        }
        print (sum)
    }
```

T.C. = O(N*M)
S.C = O(1) ↑ most optimal

Quiz



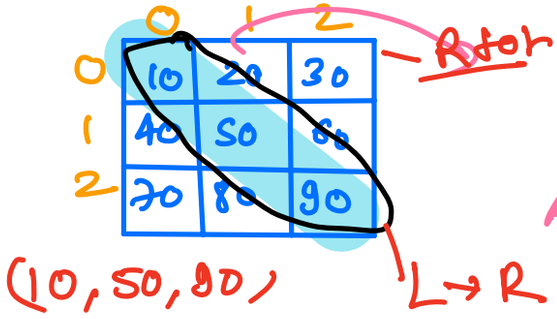
find max number
T.C. O(N)
↑ most optimal T.C.

Q ex: Given mat[N][M], print col^m wise sum. **TODO**
(can discuss during doubt session)

	0	1	2	3	4
0	2	-1	4	7	8
1	1	11	20	5	7
2	8	4	5	3	2
3	1	3	2	4	10

12, 17, 31, 19, 27

Q. Given a square matrix A[N][N], print the diagonal



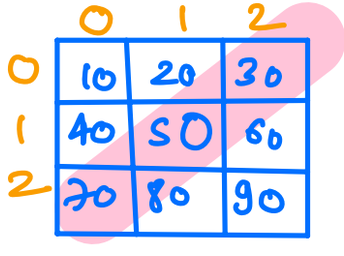
A[0][0]
A[1][1]
A[2][2]

L→R (R-L)

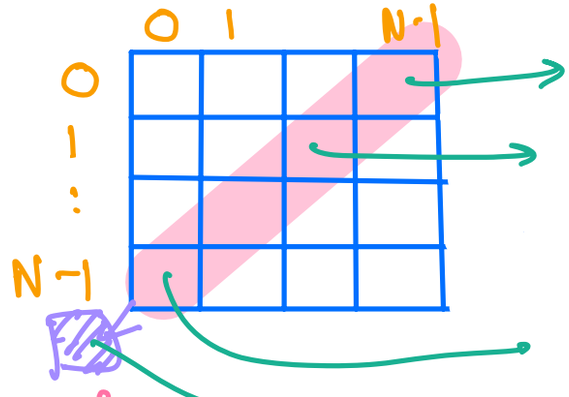
```
for(i=0; i<N; i++){
    print A[i][i]
}
```

T.C. = O(N)
S.C. = O(1)

Q. R to L Print diagonal elements.



Print (30, 50, 70)



A[0][N-1]
A[1][N-2]
A[N-1][0]
A[N][-1] ✗

obsn: ① row starts from 0 → N-1 ↑

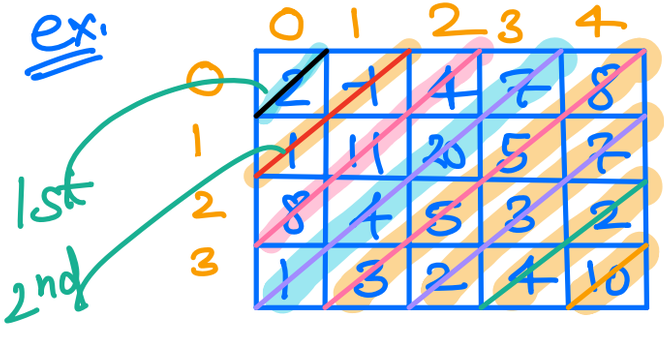
(2) col starts from N-1 → 0 ↓

```
i = 0
J = N-1
while ( i<N and J≥0 ){
    print A[i][J]
     i++;
    J--;
}
```

T.C. = O(N)
S.C. = O(1)

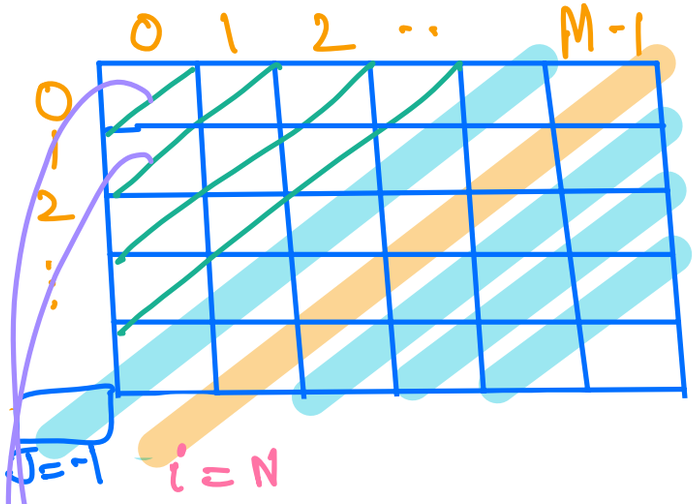
Q. Given a A[N][M], print all the diagonals from R to L

ex.



ans.
print

2
-1 1
4 11 8
7 20 4 1
8 5 5 3
7 3 2
2 4
1 0



Obsⁿ $i >= 0$ $J < M$

(I) $i < N$ and $J >= 0$

(II) when we are printing a diagonal J varies from (0, to M-1)
starting i was fixed at 0

(III) to print rest of the elemnts
starting (J) is fixed M-1
starting i will change

Solⁿ: (I) First print all the diagonal starting from 0th row

(II) print all the diagonals starting from last colm.

starting index (i) (J) for [0 M-1]


```
void printAllDiagonals(int A[][]) {
    int N = A.length
    int M = A[0].length

    for(c=0; c<M; c++){
        int i=0, J = c      // starting index of the diagonal
        while(i<N and J>=0){
            print(A[i][J])
            i++;
            J--;
        }
    }

    for(~~r=0~~ r=1; r<N; r++){
        int J=M-1, i=r      // starting index of the diagonal
        while(i<N and J>=0){
```

11:05

```
        print ( A[i][j] )
          i++;
        j--;
    }
}
```

TC. = O(N*M)
S.C = O(1)

	0	1	2	3	4
0	2	1	4	7	8
1	1	11	20	5	7
2	8	4	5	3	2
3	1	3	2	4	10

N = 4, M = 5

c = 0 1 2

i = 0 1 0 1 2 0 1 2 3 ...

j = 0 1 1 0 1 2 1 0 -1 ...

upper triangle

2
1 1
4 11 8
7 20 4 1
8 5 5 3

N = 4, M = 5

r = 1 2 3

j = 4 3 2 1 4

i = 1 2 3 4 2

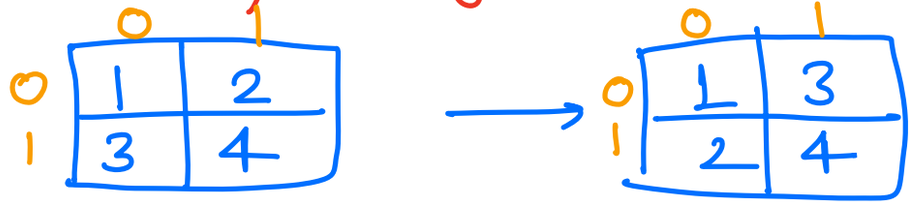
10:54 → 11:05

7 3 2

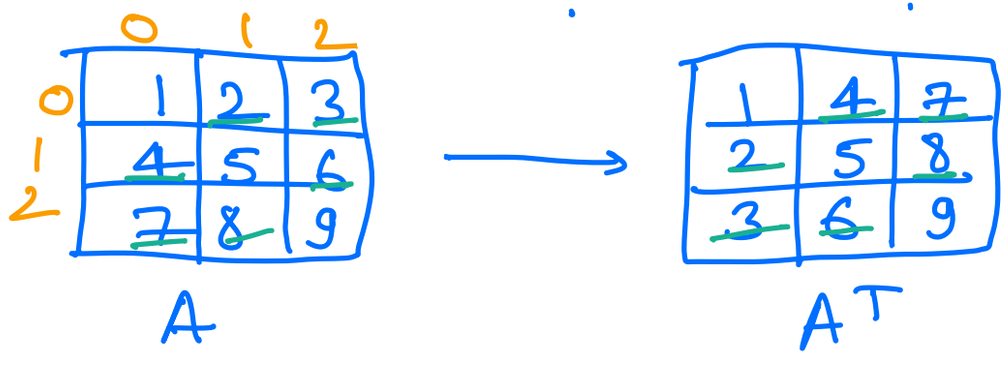
2 4

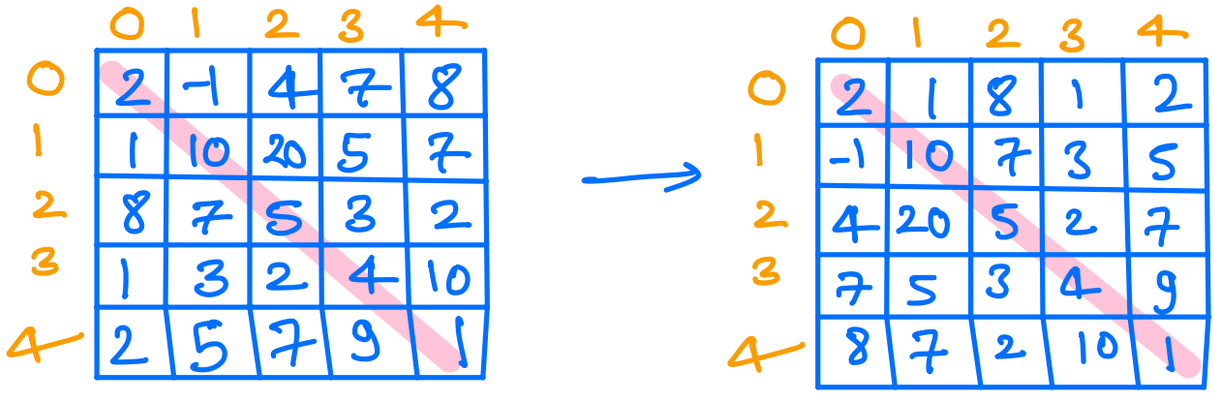
10

Q. Given a A[N][N], calculate transpose of A.
with S.C. = O(1)
modify the given matrix to get the answer.



transpose → 0^{th} row ⇒ 0^{th} col
1^{st} row ⇒ 1^{st} col
⋮ ⋮





soln: swap every A[i][j] with every A[j][i]

```
void transpose(int&A[][]){
```

ignore this if you are not comfortable with calling by address

```
int[][] transpose(int A[][]) {
```

✗ won't work

```
// N, M
for(i=0; i<N; i++){
    for(j=0; j<M; j++){
        int temp = A[i][j]
        A[i][j] = A[j][i]
```

swapping

A[0][0] → A[0][0]
A[3][3] → A[3][3]

1st soln ⇒ visit the lower triangle and swap

2nd soln → swap the upper triangle

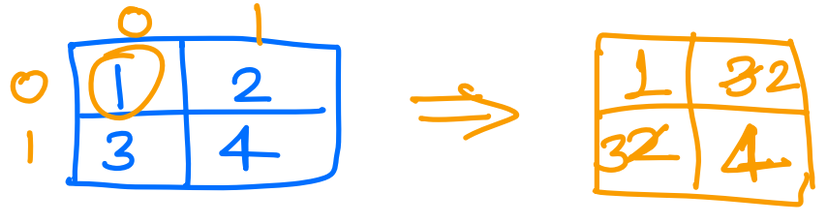
Hint

```
}
    }
        }
```

```
A[j][i] = temp
```

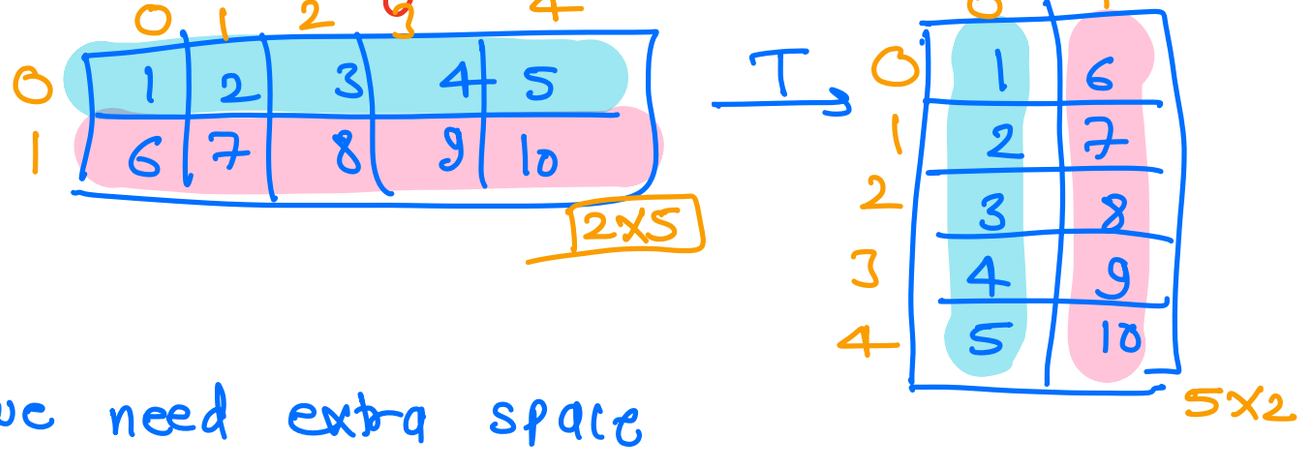
find the bug & solve it

i<j OR i>j



T.C. = $O(N^2)$

Q. transpose of a Matrix A[N][M] (Rectangle)

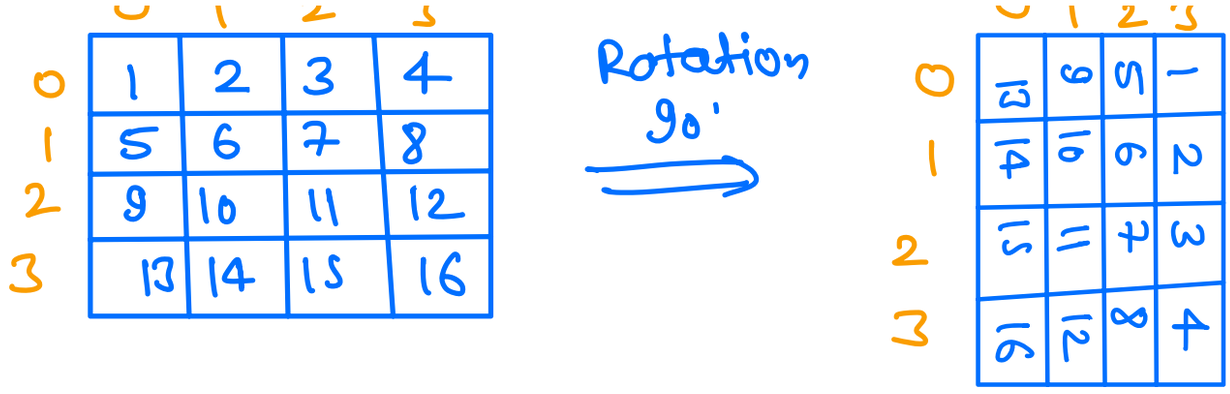


T.C = $O(N*M)$

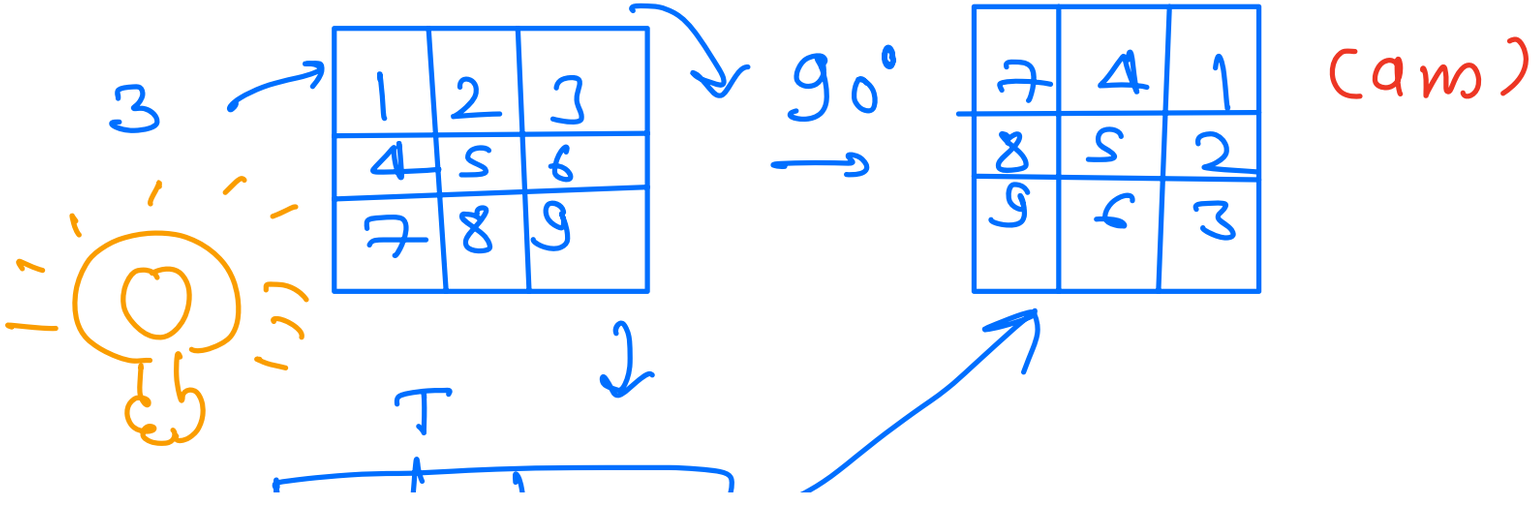
S.C = $O(N*M)$

we need extra space

Q. Given a matrix A[N][N], Rotate 90° clockwise



N	Matrix	ans
0	[ ]	[ ]
1	[1]	[1]
2	$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$	3 1 4 2



(ans)

1	4	7
2	5	8
3	6	9

reversing each row

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

90°

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

↓T

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

?

Reverse.

Soln ⇒ transpose ⟶ reverse each row

Doubt

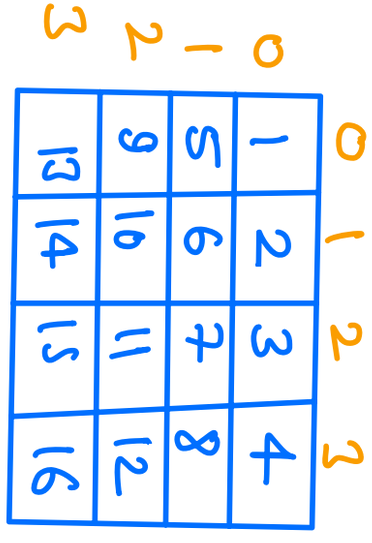
Q. (1) 30 min pen & paper Brute force Soln?

(examples & observations)

(2) accen the 1st hint

(2) > 1 hr. 2nd hint

(3) Raise the TA request



RGB

(0,255)

(255,0,0)

(0,255,0)