

Rectangular grid \Rightarrow Element / cell

Matrix \Rightarrow Rows \times Columns \Rightarrow Size

Declare \Rightarrow `int mat[N][N]`
 \uparrow \uparrow
 Rows Column

Access element in i^{th} row & j^{th} column
`mat[i][j]`

3 \times 4 matrix
N M

	0	1	2	3	
0				X	$\rightarrow 0, M-1$ (Top Right)
1			1, 2		
2				X	$\rightarrow N-1, M-1$ (Bottom Right)

📌 Given a 2D matrix of size $N \times M$.
Print the row-wise sum of the matrix

3 \times 4 matrix

	0	1	2	3	
0	1	2	3	4	$\rightarrow 10$
1	5	6	7	8	$\rightarrow 26$
2	9	10	11	12	$\rightarrow 42$

Solⁿ

Iterate all the rows one by one.

Code

```
void sumRow (int mat[N][M]) {
```

```
    for (i=0; i<N; i++) {
```

```
        sum = 0;
```

```
        for (j=0; j<M; j++) {
```

```
            sum = sum + mat[i][j];
```

```
        }
```

```
        print(sum);
```

```
    }
```

```
}
```

T.C. = $O(N \times M)$

S.C. = $O(1)$

Q

Given a 2D matrix of size $N \times M$.
Print the column-wise sum of the matrix

3x4 matrix

	0	1	2	3
0	1	2	3	4
1	5	6	7	8
2	9	10	11	12
	15	18	21	24

```
void sumRow (int mat[N][M]) {
```

```
    for (j = 0; j < M; j++) {
```

```
        sum = 0;
```

```
        for (i = 0; i < N; i++) {
```

```
            sum = sum + mat[i][j];
```

```
        }
```

```
        print(sum);
```

```
    }
```

```
}
```

T.C. = $O(N \times M)$
S.C. = $O(1)$



Given a square matrix of size $N \times N$

Print all the diagonal elements from left to right.

	0	1	2
0	0,0		
1		1,1	
2			2,2

Principal diagonal
($i = j$)

	0	1	2
0			
1			
2			

Anti-diagonal

1) Principal diagonal

```
void printdiagonal (mat) {
```

```
    int i = 0;
```

```
    int j = 0;
```

```
    while (i < N && j < N) {
```

```
        print (mat[i][j]);
```

```
        i++;
```

```
        j++;
```

```
    }
```

```
}
```

```
void printdiagonal (mat) {
```

```
    int i = 0;
```

```
    while (i < N) {
```

```
        print mat[i][i];
```

```
        i++;
```

```
    }
```

```
}
```

T.C. = $O(N)$

2) Anti Diagonal

	0	1	2
0			0, 2
1		1, 1	
2	2, 0		

$i--$, $j++$
 \longrightarrow

Anti-diagonal

$(N-1, 0)$

$i = N-1, j = 0$

while ($i > 0$) &

print mat[i][j];

$i--;$

$j++;$

&

T.C. = $O(N)$

Rectangular matrix $\Rightarrow N \times M$

M

	0	1	2	3
0	1	2	3	4
1	5	6	7	8
2	9	10	11	12

N Right to Left

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

Diagonals = $N+M-1$

	0	1	2	3
0	1	2	3	4
1	5	6	7	8
2	9	10	11	12

```
for (i = 0; i < M; i++) {
```

```
    r = 0
```

```
    c = i
```

```
    while (r < N && c >= 0) {
        print (mat[r][c]);
```

```
        r++;
```

```
        c--;
```

```
    }
```

```
    print ("\n");
```

```
}
```

```
for (i = 1; i < N; i++) {
```

```
    c = M - 1
```

```
    r = i;
```

```
    while (r < N && c >= 0) {
        print (mat[r][c]);
```

```
        r++;
```

```
        c--;
```

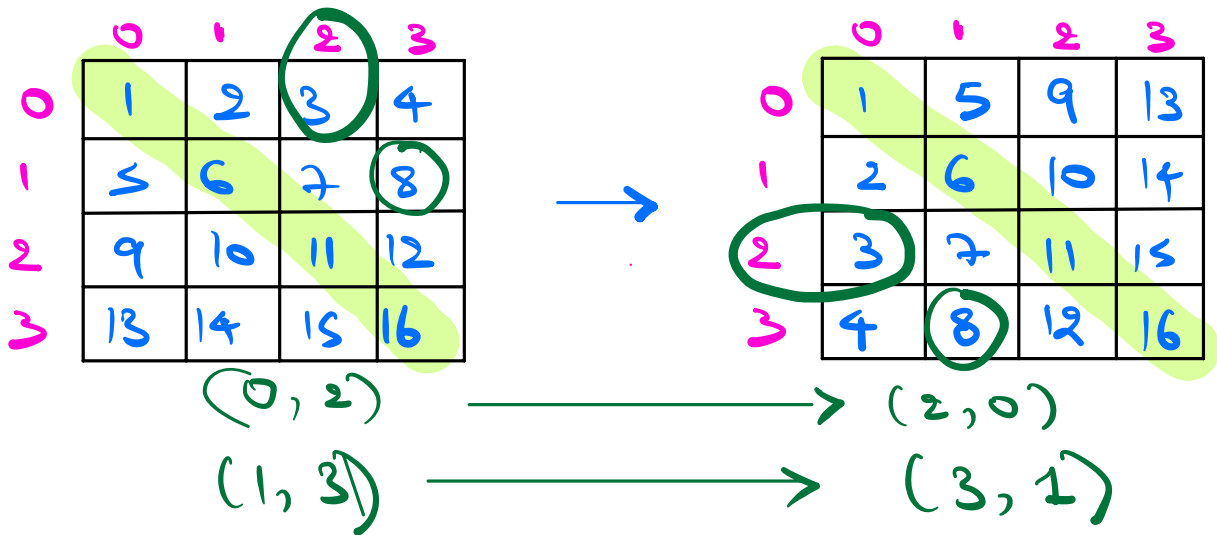
```
    }
```

print("\n")

↓

Q

Given a square matrix of $N \times N$.
Convert the matrix to its transpose.



for ($i=0$; $i < N$; $i++$)
 for ($j=0$; $j < N$; $j++$)
 transpose[j][i] = mat[i][j];

↓

↓

Code

for ($i=0$; $i < N$; $i++$)
 for ($j=i+1$; $j < N$; $j++$)

$temp = mat[i][j];$
 $mat[i][j] = mat[j][i];$
 $mat[j][i] = temp;$

$T.C. = O(N^2)$
 $S.C. = O(1)$

Rectangular matrix

3 × 4

⇒

4 × 3

Given a square matrix of $N \times N$
 Rotate the matrix 90° clockwise

90°

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

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

↓

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

Transpose

$0 \rightarrow N-1$
 $1 \rightarrow N-2$

0 \rightarrow 0
1 \rightarrow 1

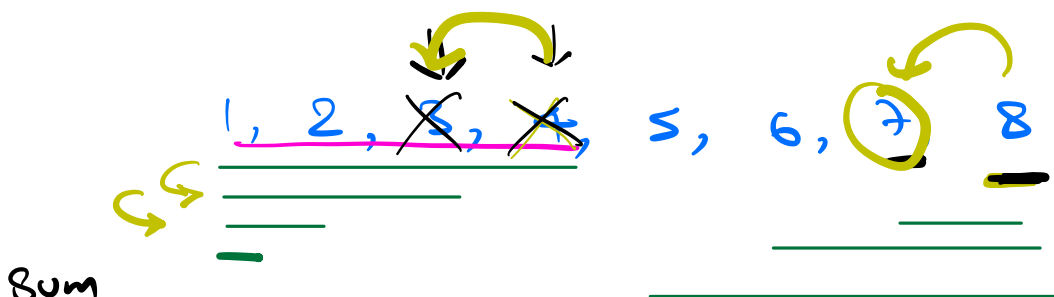
- 1) Take transpose of matrix N^2
- 2) Reverse each row. N^2

$$\begin{aligned} \text{T.C.} &= O(N^2) \\ \text{S.C.} &= O(1) \end{aligned}$$

Interviewkit
LeetCode
GeeksforGeeks

100 perp

100 hurr



$B = 4$

Sum

$$i = B - 1$$

$$j = N - 1$$

for ($k=0$; $k < B$; $k++$) {

$Sum = Sum - A[i] + A[j];$

arr \leftrightarrow

$i++$

$j--$

k