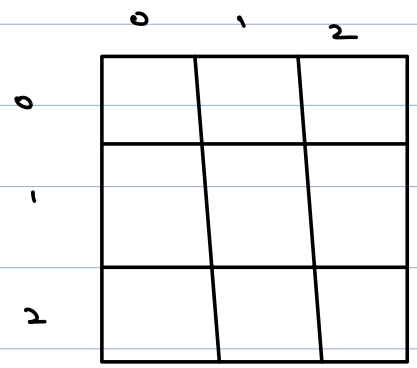
Definition :- 2d array, rectangular grid, where each number is an element.

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

int → Type

mat → variable name.

N → rows, M → cols

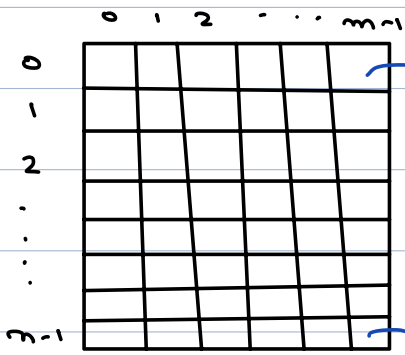


mat[3][3]

↓

9 elements.

int mat[N][M]



Ques 2d matrix [n][m], print row wise sum.

mat[3][4] =

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

o/p → 10
26
42

T.C → O(nm)
S.C → O(1)

```
for (i=0; i<n; i++) {
    sum = 0;
    for (j=0; j<m; j++) {
        sum += mat[i][j];
    }
    print (sum);
}
```

Ques 2d matrix [n][m], Print col wise sum.

mat[3][4] =

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

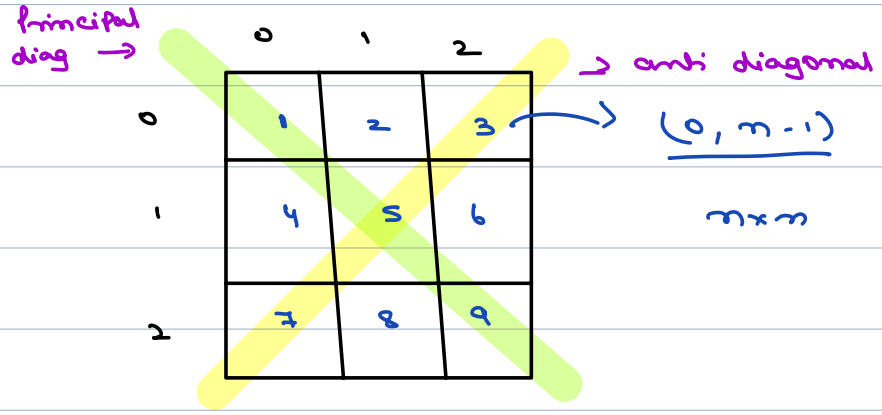
o/p → 15
18
21
24

```
for (col = 0; col < m; col++) {
    sum = 0;
    for (row = 0; row < n; row++) {
        sum += mat[row][col];
    }
    print(sum);
}
```

T.C → O(n * m)
S.C → O(1).

Ques Given 2d mat[n][n], Print 1) Principal diag. 2) Anti diagonal

square matrix.



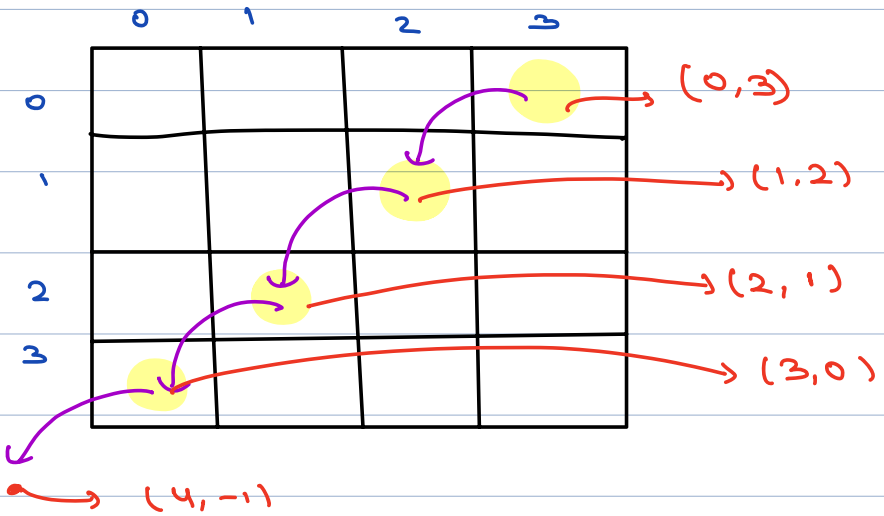
1) Print Principal diagonal

```
i = 0;                                   T.C → O(n)
                                         S.C → O(1)
while (i<n) {
    Print (mat[i][i]);
    i++
}
```

2) Print anti diagonal element,

```
i = 0, j = n-1                  → only one condn is enough.
while ( i<n && j>=0 ) {
    Print (mat[i][j]);
    i++;
    j--;
}
```

$n = 4$

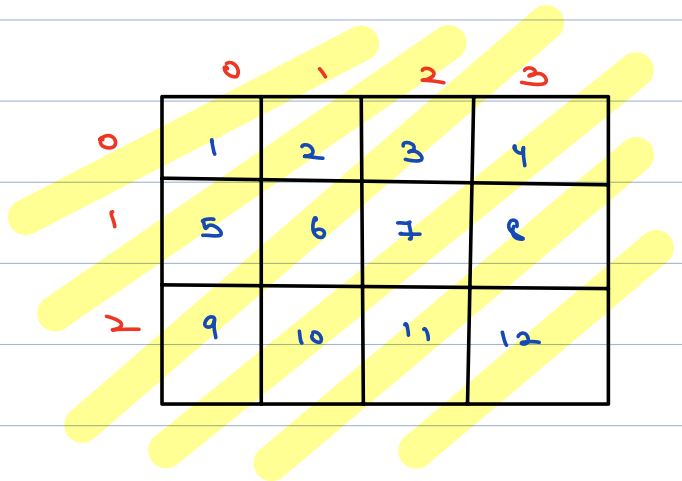


T.C → O(n)

S.C → O(1)

Ques given 2d matrix $n \times m$, print all the diagonals from right to left.

mat[3][4] =



O/P :-

1

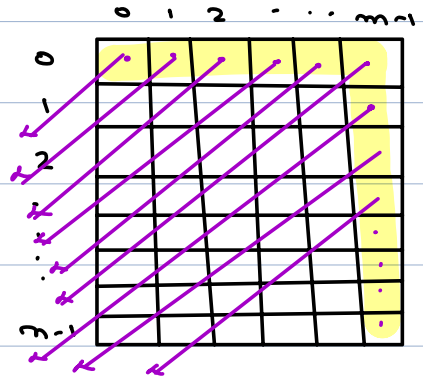
2 5

3 6 9

4 7 10

8 11

12



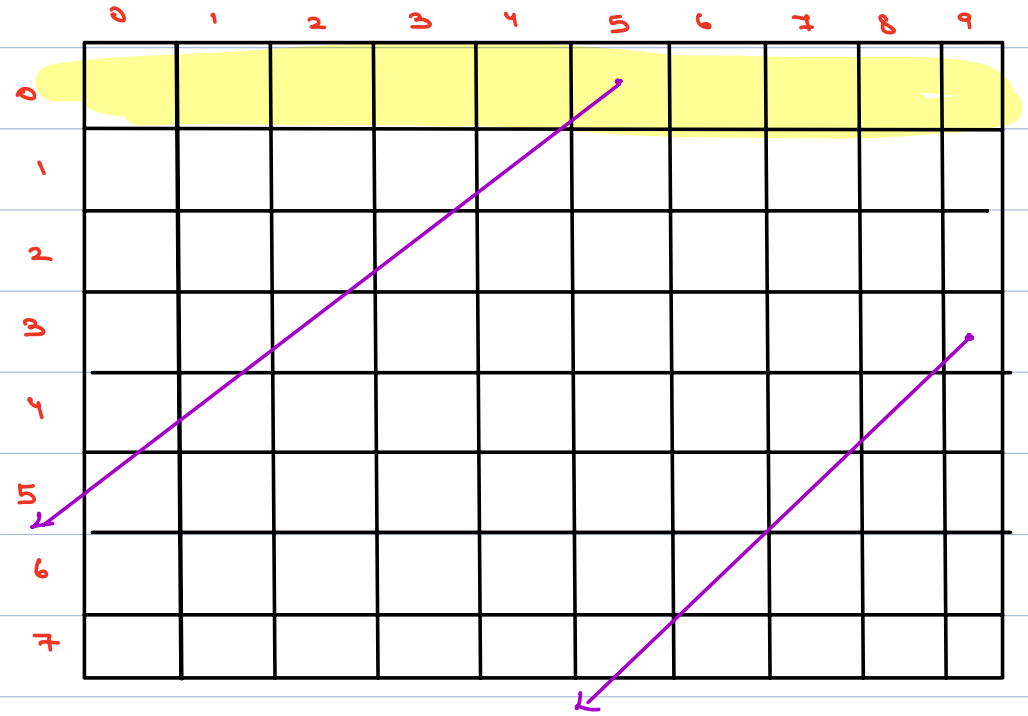
int mat[N][M]

M+N-1

↓

R to ↓ diagonals

int mat[4][5] = 4+5-1 => 8 R to l diag.



(0,5) → (1,4) → (2,3) → (3,2) → (4,1) → (5,0) → (6,-1)

(3,9) → (4,8) → (5,7) → (6,6) → (7,5) → (8,4)

```
for ( col = 0; col < m; col++ ) {
    i = 0; j = col;
    while (i < n && j >= 0) {
        print (mat[i][j]);
        i++;
        j--;
    }
}
```

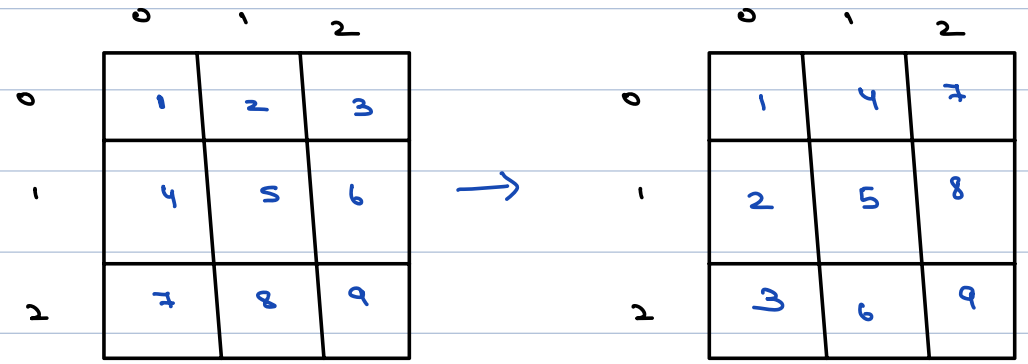
```
for (row = 1; row < n; row++) {
    i = row
    j = m - 1;
    while (i < n && j >= 0) {
        print (mat[i][j]);
        i++;
        j--;
    }
}
```

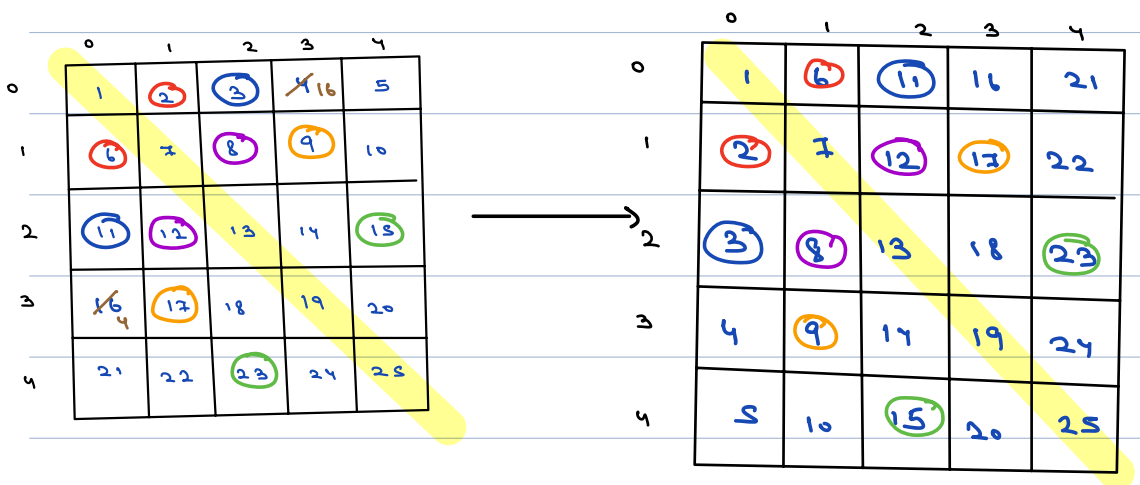
T.C → O(n × m)

S.C → O(1)

Ques Given a 2d matrix [N][N], find transpose.

↳ Square matrix





mat[i][j] ⇌ mat[j][i]

```
for (i=0; i<n; i++) {
    for (j=i+1; j<n; j++) {
        swap(mat[i][j], mat[j][i]);
```

3 (outer loop)

3 (inner loop)

i	j	
0	(1 n-1)	n-1
1	(2 n-1)	n-2
2	(3 n-1)	n-3
3	(4 n-1)	n-4
⋮		⋮
n-1	—	0

$1+2+3+\dots+(n-1)$

$1+2+3 \dots n \Rightarrow \frac{n(n+1)}{2}$

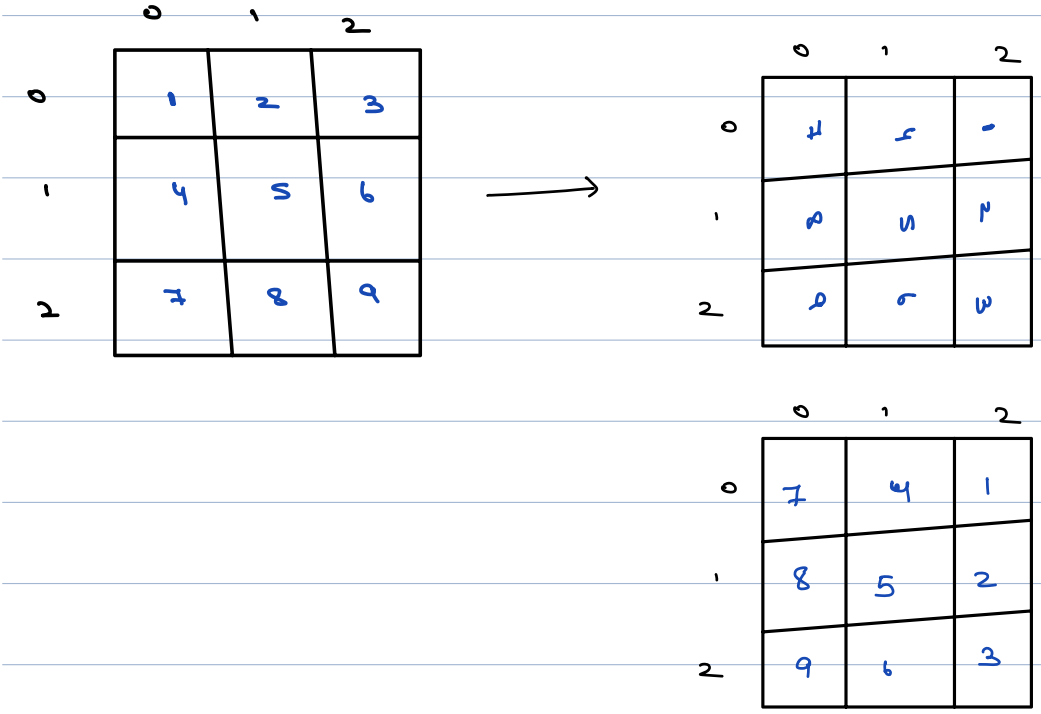
$1+2+3+\dots+(n-1)$

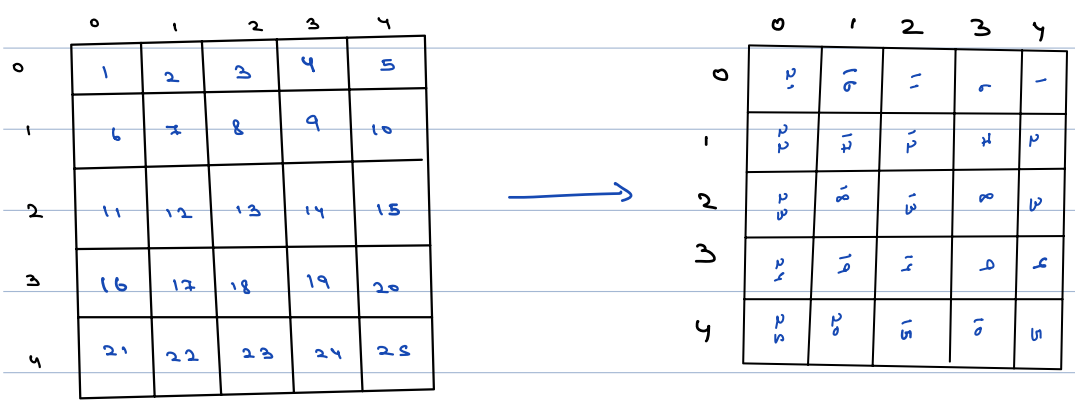
$\frac{(n-1)n}{2} \Rightarrow O(n^2)$

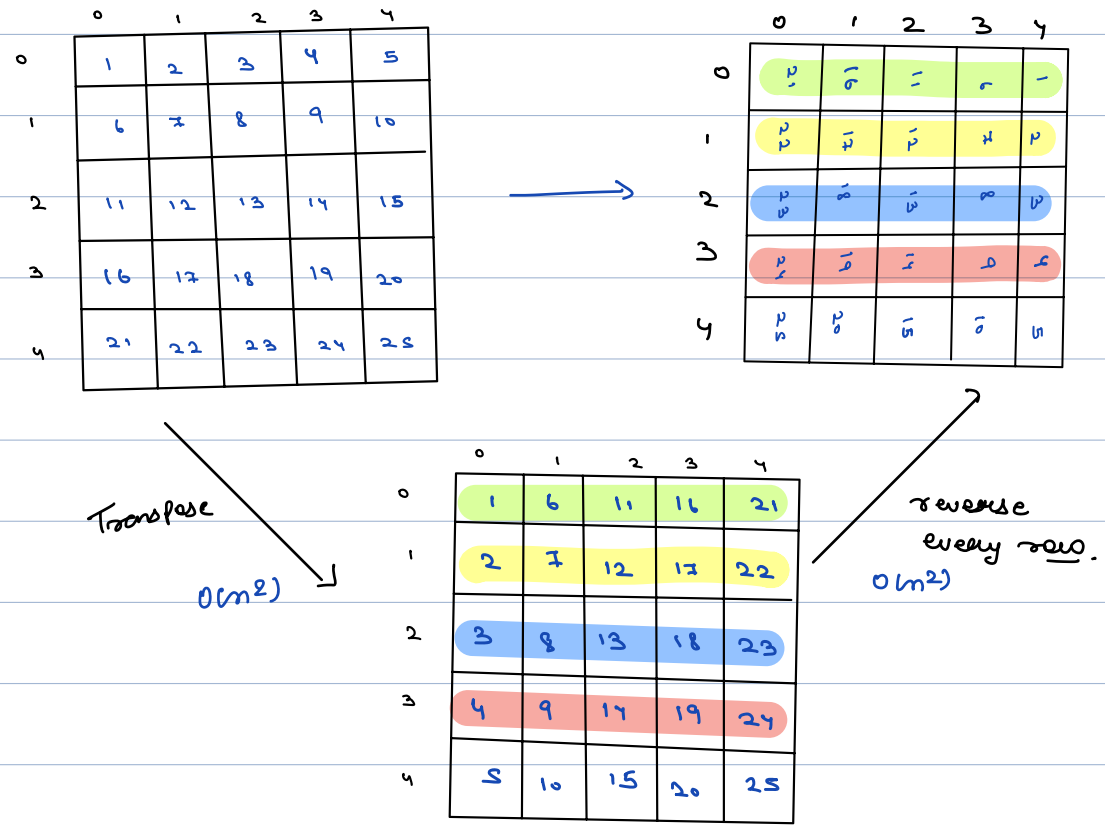
T.C

S.C $\rightarrow O(1)$

Ques Rotate 90° clockwise, mat[N][N]







T.C → O(n²), S.C → O(1).

Transpose + Reverse Every Row = Rotate 90°.

n/2 × n

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{bmatrix}_{3\times4} \xrightarrow{T} \begin{bmatrix} 1 & 5 & 9 \\ 2 & 6 & 10 \\ 3 & 7 & 11 \\ 4 & 8 & 12 \end{bmatrix}_{4\times3}$$

	10,	20,	30,	40,	50
pf[] =	10	30	60	100	150
sf[] =	150	140	120	90	50

pf[i] = sum (0 to i)

sf[i] = sum (i to n-1);

n = 10, m = 5

diff b/w n+m, n*m

```
for (i=0; i<n; i++) {
    print (—)        // 10 times
}
for (i=0; i<m; i++) {
    print (—)        // 5 times
}
```

n+m

⇒ 15 times

```
for (i=0; i<n; i++) {
    for (i=0; i<m; i++) {
        print (—)
    }
}
```

n*m

⇒ 50 times

$1 + 2 + 3 + \cdots n = \frac{n(n+1)}{2}$

5 → $\frac{5(5+1)}{2}$

10 → $\frac{10(10+1)}{2}$.

$(n-1)$ → $\frac{(n-1)(n-1+1)}{2} \Rightarrow \frac{(n-1)(n)}{2}$

$\Rightarrow \frac{n^2 - n}{2}$

$\frac{n^2}{2} - \frac{n}{2} \Rightarrow O(n^2)$.