Q: Matrix Multiplication :

$$\underset{\substack{2\times 3\\ m_1\times n_1}}{\begin{bmatrix}1&2&3\\4&5&6\end{bmatrix}} * \underset{\substack{3\times 2\\ m_2\times n_2}}{\begin{bmatrix}10&11\\20&21\\30&31\end{bmatrix}} = \underset{2\times 2}{\begin{bmatrix}1*10+2*20+3*30 & 1*11+2*21+3*31\\ 4*10+5*20+6*30 & 4*11+5*21+6*31\end{bmatrix}}$$

* you can only multiply $m_1 \times n_1$ matrix with $m_2 \times n_2$ matrix only if $n_1 == m_2$

  no. of cols in 1st mat = no. of rows in 2nd mat

* the resultant matrix will be $m_1 \times n_2$

$$\underset{m_1\ \ n_1}{2\times 3} * \underset{m_2\ \ n_2}{3\times 2} = 2\times 2$$

$$\begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & a_{00} & a_{01} & a_{02} \\ 1 & a_{10} & a_{11} & a_{12} \end{array} \quad * \quad \begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & b_{00} & b_{01} & b_{02} & b_{03} \\ 1 & b_{10} & b_{11} & b_{12} & b_{13} \\ 2 & b_{20} & b_{21} & b_{22} & b_{23} \end{array} \quad = \quad \begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & c_{00} & c_{01} & c_{02} & c_{03} \\ 1 & c_{10} & c_{11} & c_{12} & c_{13} \end{array}$$

2×3 $\quad$ 3×4 $\quad$ 2×4

r $\quad$ ← c →

$$\overset{rc}{c_{00}} = \overbrace{a_{00} \cdot b_{00}}^{0} + \overbrace{a_{01} \cdot b_{10}}^{1} + \overbrace{a_{02} \cdot b_{20}}^{2}$$

$$c_{01} = \overbrace{a_{00} \cdot b_{01}}^{0} + \overbrace{a_{01} \cdot b_{11}}^{1} + \overbrace{a_{02} \cdot b_{21}}^{2}$$

$$c_{02} = \overbrace{a_{00} \cdot b_{02}}^{0} + \overbrace{a_{01} \cdot b_{12}}^{1} + \overbrace{a_{02} \cdot b_{22}}^{2}$$

$$c_{10} = \overbrace{a_{10} \cdot b_{00}}^{0} + \overbrace{a_{11} \cdot b_{10}}^{1} + \overbrace{a_{12} \cdot b_{20}}^{2}$$

$$c_{12} = \overbrace{a_{10} \cdot b_{02}}^{0} + \overbrace{a_{11} \cdot b_{12}}^{1} + \overbrace{a_{12} \cdot b_{22}}^{2}$$

$c_{12} \Rightarrow$ 1st row of a * 2nd col of b

c[r][c] = a[r][?] * b[?][c]

* c[1][2] = a[1][?] * b[?][2]

$\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad$ k $\quad\quad$ k

(or) 0 → no-of rows (b)

0 → no. of cols (a)

Q: Transpose of a matrix :

	0	1	2	3	4
0	1	2	3	4	5
1	6	7	8	9	10
2	11	12	13	14	15
3	16	17	18	19	20
4	21	22	23	24	25

	0	1	2	3	4
0	1	6	11	16	21
1	2	7	12	17	22
2	3	8	13	18	23
3	4	9	14	19	24
4	5	10	15	20	25

$n \times m \Leftrightarrow m \times n$

rows ⟷ cols

mat → $n \times m$

1. initialize
   res ⇒ $m \times n$

2.
```
for(r=0; r<m; r++){
    for(c=0; c<n; c++){
        res[r][c] = mat[c][r]
    }
}
```

Eg: res[2][3] = mat[3][2]

res[0][2] = mat[2][0]

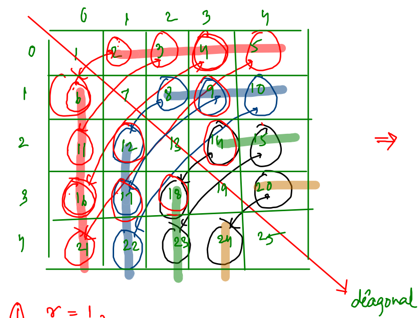
$O(m*n)$ → TC

$O(m*n)$ → SC

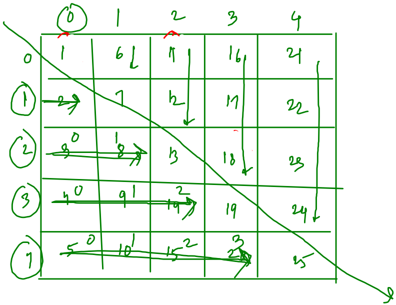
* Inplace

$O(1)$

Optimising SC, Inplace :



⇒



(1) r = 1,

c = 0 ⇒ mat[1][0] ↔ mat[0][1]

c = 1 ⇒ 1 < 1 ✗

(2) r = 2,

c = 0 ⇒ mat[2][0] ↔ mat[0][2]

c = 1 ⇒ mat[2][1] ↔ mat[1][2]

c = 2 ⇒ 2 < 2 ✗

(3) r = 3

c = 0 ⇒ mat[3][0] ↔ mat[0][3]

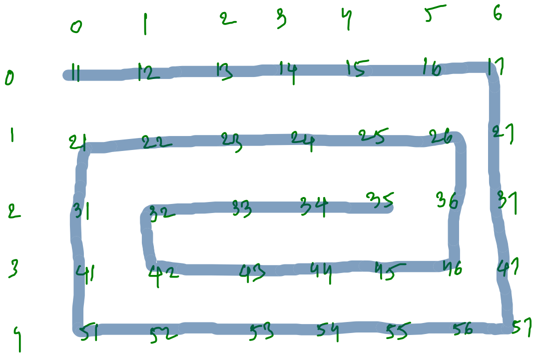
c = 1 ⇒ mat[3][1] ↔ mat[1][3]

c = 2 ⇒ mat[3][2] ↔ mat[2][3]

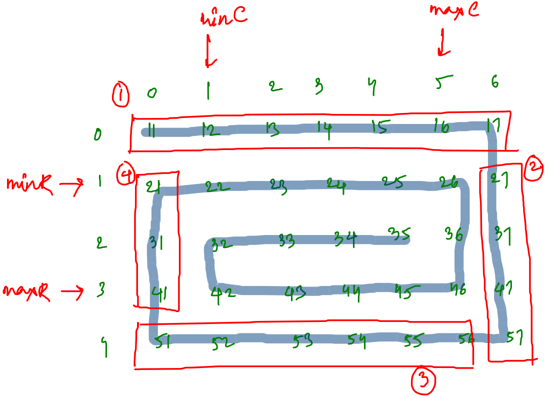
```
for(let r=1; r<n; r++) {
    for(let c=0; c<r; c++) {
        swap(mat[r][c], mat[c][r])
    }
}
```

$\boxed{\text{9:35 PM – 9:50 PM BREAK}}$

Q: Spiral Traversal :



op: 11 12 13 14 15 16 17 27 37 47 57
56 55 54 53 52 51 41 31 21 22
23 24 25 26 36 46 45 44 43
42 32 33 34 35



1. Topwall ✗
2. rightwall
3. bottom wall
4. Leftwall

minR (top wall)
minC (left wall)
maxR (bottom wall)
maxC (right wall)

①
```
for(let c = minC ; c <= maxC; c++){
    print(mat[minR][c] + ' ');
}
minR++ ;
```

②
```
for(let r = minR ; r <= maxR; r++){
    print (mat [r][maxC]);
}
maxC--;
```

③
```
for(let c = maxC; c >= minC; c--){
    print(mat[maxR][c]);
}
maxR--;
```

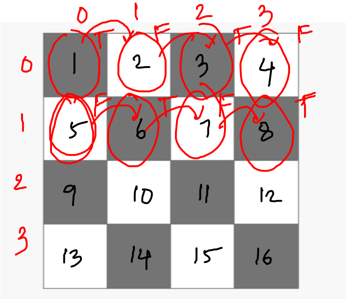
④
```
for(let r = maxR; r >= minR; r--){
    print(mat[r][minC]);
}
minC++
```

* you need to run these 4 loops until all boundaries are covered. (all elements are printed)

Q: Alt Matrix Sum:

op: $1+3+6+8+9+11+14+16 = X$

$2+4+5+7+10+12+13+15 = Y$



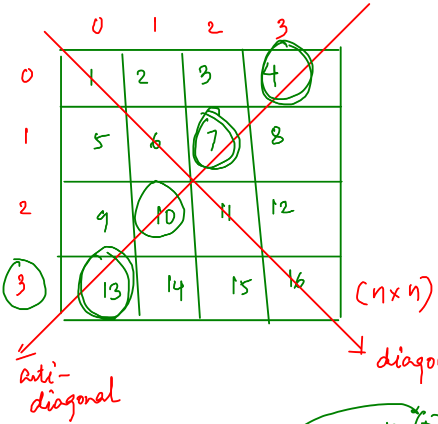
4 x 4 Chess Board

```
for (r = 0; r < rows; r++) {
    if (r % 2 == 0) isBlack = true;
    else
    for (c = 0; c < cols; c++) {
        if (isBlack) bSum += mat[r][c];
        else wSum += mat[r][c];
        isBlack = !isBlack
    }
}
```

Black	white
(0, 0) = 0	(0, 1) = 1
(0, 2) = 2	(0, 3) = 3
(1, 1) = 2	(1, 0) = 1
(1, 3) = 4	(1, 2) = 3
(2, 0) = 2	(2, 1) = 3
(2, 2) = 4	(2, 3) = 5
(3, 1) = 4	(3, 0) = 3
(3, 3) = 6	(3, 2) = 5
even	odd

```
for (r = 0; r < rows; r++) {
    for (c = 0; c < cols; c++) {
        if ((r + c) % 2 == 0) {
            bSum += mat[r][c]
        }
        else {
            wSum += mat[r][c]
        }
    }
}
```

Q: Diagonal difference :



diagonal

(0, 0)
(1, 1)
(2, 2)
(3, 3)

$r == c$

(r, r)
(c, c)

anti diagonal

$(0, 3) \Rightarrow (0, n-0-1) \Rightarrow (0, 4-0-1)$
$(1, 2) \rightarrow (1, n-1-1) \Rightarrow (0, 4-1-1)$
$(2, 1) \rightarrow (2, n-2-1) \Rightarrow (0, 4-2-1)$
$(3, 0) \Rightarrow (3, n-3-1) \Rightarrow (0, 4-3-1)$

$(r, n-r-1)$

cols

$n - (0) - 1$
$r = 0$, cols − 1
$n - (1) - 1$
$r = 1$, cols − 2
$n - (2) - 1$
$r = 2$, cols − 3
$n - (3) - 1$
$r = 3$, cols − 4

```
for( let r = 0; r < n; r++) {
    diag += mat[r][r];
    adiag += mat[r][n-r-1];
}
console.log( abs(diag-adiag));
```