Todays Content

→ Basics

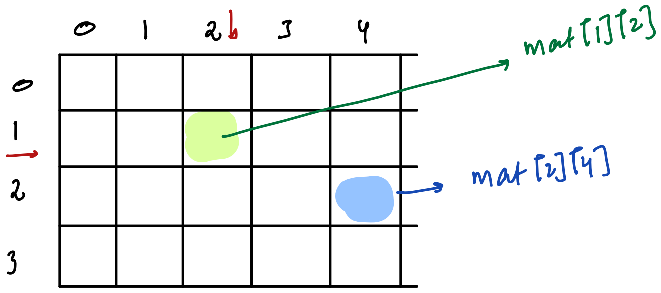
→ Problems

Declare:

int mat[4][5]

rows, horizontal lines

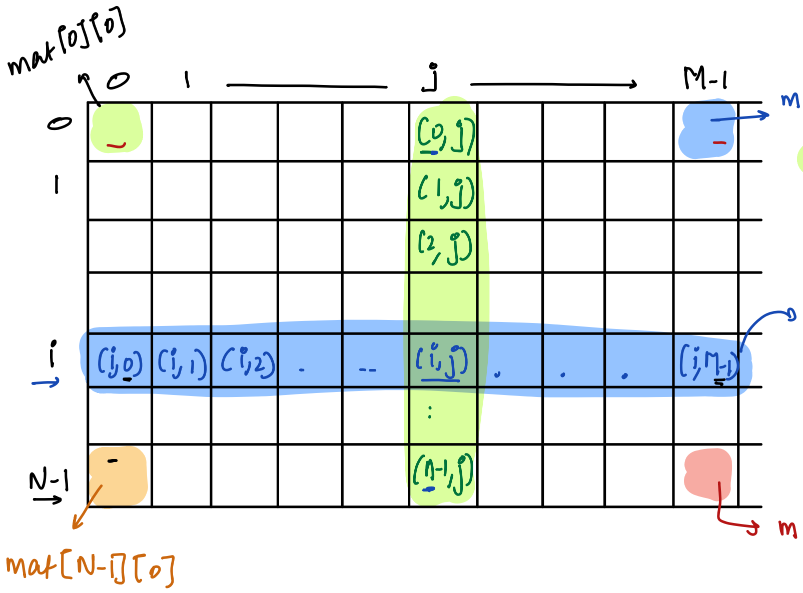
cols, vertical lines



int mat[N][M]

N rows

M Columns



Obs:

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]

1Q) Given mat[N][M], print raw wise sum

Ex: mat[3][4]

	0	1	2	3	output
0	4	3	1	7	15
1	6	2	3	4	15
2	5	3	2	7	17

```
void row_wise(int mat[][]){
    int N = mat.length
    int M = mat[0].length
    for(int i = 0; i < N; i++){
        // We need i^th rows sum
        int sum = 0
        for( int j = 0; j < M; j++){
            sum = sum + mat[i][j]
        }
        print(sum)
    }
    TC: O(N*M)   SC: O(1)
}
```

1Q) Given mat[N][M], print col wise sum {TODO}

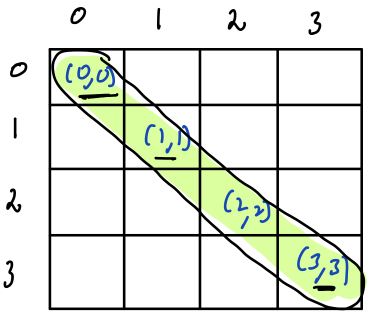
→ In Doubts Session Code :

	0	1	2	3
0	4	3	1	7
1	6	2	3	4
2	5	3	2	7

output: 15 8 6 18

28) Given Square mat[N][N] print diagnols → left → Right
→ Right → left

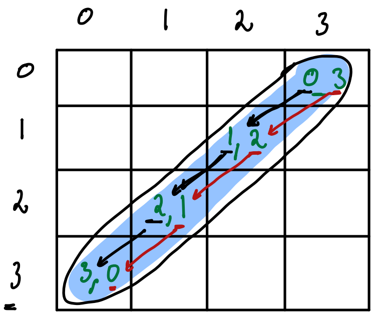
Ex: mat[4][4]



Pseudocode: Try it for loop

```
int i=0, j=0
while(i<n && j<n){
    print(mat[i][j])
    i++, j++
}
```

TC: O(N)
SC: O(1)



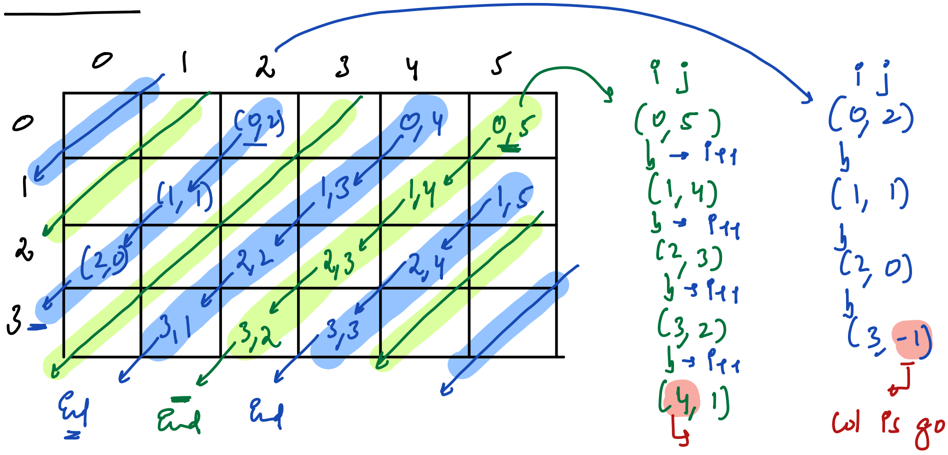
```
int i=0, j=N-1
while(i<N && j>=0){
    print(mat[i][j])
    i++ j--
}
```

TC: O(N)
SC: O(1)

3Q) Given a mat[N][M] print all diagnols going from R→L

starting from 0th row & M-1th column

mat[4][6]



row is going out of bounds

col is going out of bounds

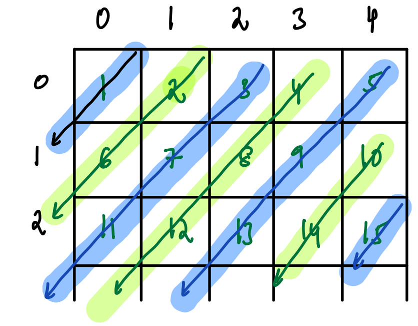
obs1: i is continously increasing, i<N

obs2: j is continuously decreasing, j>=0

// even if 1 condition fails we stop

i<N && j>=0

mat[3][5]



output:

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

idea

1) First print all diagnols starting from 0th row

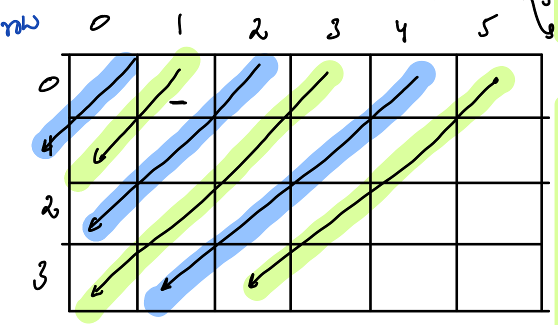
2) Print all diagnols starting from last column

```
void printdiagnols(int mat[][]){   mat[4][6]
    int N = mat.length
    int M = mat[0].length           diagnols from 0th row
    for( int c = 0, c < M; c++){
        int i=0, j=c
        while( i<N && j>=0 ){
            print( mat[i][j])
            i++, j--
        }
        print(new line)
    }
```

c = 0; c < 6; c++) { [0][c]}

start diagnol

- [0,0]
- [0, 1]
- [0, 2]
- [0, 3]
- [0,4]
- [0,5]



r=1 → {r==0, repeation same diagnol 2 times}

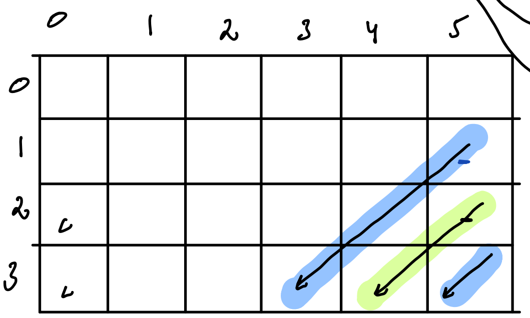
for(int r=~~0~~ ; r < N; r++){ → diagnols from last column

```
        int i=r, j= M-1
        while( i<N && j>=0){
            print(mat[i][j])
            i++, j--;
        }
        print(new line)
    }
}
```

r= 1 ; r < 4; r++ { [r, 5]

Start diagnol

- [0,5]
- [1,5]
- [2,5]
- [3,5]



TC: O(N*M) SC: O(1) 8:53 → 9:02 am

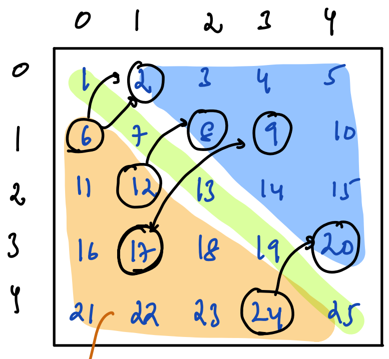
```
int i=0, j=c
while( i<N && j>=0 ){
    print( mat[i][j])
    i++, j--
}
print(new line)
```

for(int i=0, j=c; i<N && j>=0 ; i++, j--){

4Q) Given a mat[N][N], Calculate transpose of mat[], with SC : O(1)

Note: Get transpose in given mat itself

mat[5][5] :



Transpose

0th row → 0th col
1st row → 1st col
2nd row → 2nd col
3rd row → 3rd col
4th row → 4th col

→ transpose

	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

Sol:

1) Iterate in lower triangle & swap] TODO

2) Iterate in upper triangle & swap] TODO

swap mat[1,0] & mat[0,1]

swap mat[3,1] & mat[1,3]

swap mat[2,1] & mat[1,2]

swap mat[4,3] & mat[3,4]

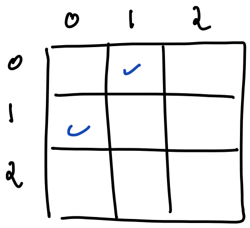
swap mat[i,j] & mat[j,i]

swap mat[3,3] & mat[3,3] → no impact

```
int[][] Transpose(int mat[][]){ TC: O(N^2) O(1) : Wrong it won't work
    int n = mat.length
    int m = mat[0].length;
    for(int i=0; i<n; i++){
        for(int j=0; j<m; j++){
            //swap mat[i][j] ⇄ mat[j][i]
            int temp = mat[i][j]
            mat[i][j] = mat[j][i]
            mat[j][i] = temp
        }
    }
}
```

mat[i][j] = mat[j][i]

Ex: mat[3][3]



mat[0,1] → mat[1,0]

mat[1,0] → mat[0,1]

//data won't get changed

a	b
10	20

swap a, b   a = 20   b = 10

swap a, b   a = 10   b = 20

// If Rectangle : {We need Extra Space to Solve Problem}

mat[2][5]   Transpose :   mat[5][2]

$$\begin{array}{c|ccccc} & 0 & 1 & 2 & 3 & 4 \\ \hline 0 & 1 & 2 & 3 & 4 & 5 \\ 1 & 6 & 7 & 8 & 9 & 10 \end{array} \qquad \begin{array}{cc} 1 & 6 \\ 2 & 7 \\ 3 & 8 \\ 4 & 9 \\ 5 & 10 \end{array}$$

// Given a mat[N][N], Rotate 90° clockwise from Top-Right

$$\begin{array}{c|ccccc} & 0 & 1 & 2 & 3 & 4 \\ \hline 0 & 1 & 2 & 3 & 4 & 5 \\ 1 & 6 & 7 & 8 & 9 & 10 \\ 2 & 11 & 12 & 13 & 14 & 15 \\ 3 & 16 & 17 & 18 & 19 & 20 \\ 4 & 21 & 22 & 23 & 24 & 25 \end{array} \xrightarrow{\text{Rotate } 90^\circ} \begin{array}{c|ccccc} & 0 & 1 & 2 & 3 & 4 \\ \hline 0 & 21 & 16 & 11 & 6 & 1 \\ 1 & 22 & 17 & 12 & 7 & 2 \\ 2 & 23 & 18 & 13 & 8 & 3 \\ 3 & 24 & 19 & 14 & 9 & 4 \\ 4 & 25 & 20 & 15 & 10 & 5 \end{array}$$

Calculate transpose + Reverse Every Row = $\boxed{\text{Rotate mat by } 90^\circ}$

$O(N^2)$ + $O(N*N)$ = TC: $O(N^2)$ SC: $O(1)$

$$\begin{array}{c|ccccc} & 0 & 1 & 2 & 3 & 4 \\ \hline 0 & 1 & 6 & 11 & 16 & 21 \\ 1 & 2 & 7 & 12 & 17 & 22 \\ 2 & 3 & 8 & 13 & 18 & 23 \\ 3 & 4 & 9 & 14 & 19 & 24 \\ 4 & 5 & 10 & 15 & 20 & 25 \end{array} \quad \begin{array}{c} \text{reverse} \\ \text{reverse} \\ \text{reverse} \\ \text{reverse} \\ \text{reverse} \end{array} \quad \begin{array}{c|ccccc} & 0 & 1 & 2 & 3 & 4 \\ \hline 0 & 21 & 16 & 11 & 6 & 1 \\ 1 & 22 & 17 & 12 & 7 & 2 \\ 2 & 23 & 18 & 13 & 8 & 3 \\ 3 & 24 & 19 & 14 & 9 & 4 \\ 4 & 25 & 20 & 15 & 10 & 5 \end{array}$$

// Rotate Rectangular matrix : We need Extra Space

$mat[2][5] \xrightarrow{\text{rotate}} mat[5][2]$

$$\begin{array}{c|ccccc} & 0 & 1 & 2 & 3 & 4 \\ \hline 0 & 1 & 2 & 3 & 4 & 5 \\ 1 & 6 & 7 & 8 & 9 & 10 \end{array}$$

$$\begin{bmatrix} 6 & 1 \\ 7 & 2 \\ 8 & 3 \\ 9 & 4 \\ 10 & 5 \end{bmatrix}$$