SCALER

2D Arrays

Starts 9:05

TABLE OF CONTENTS

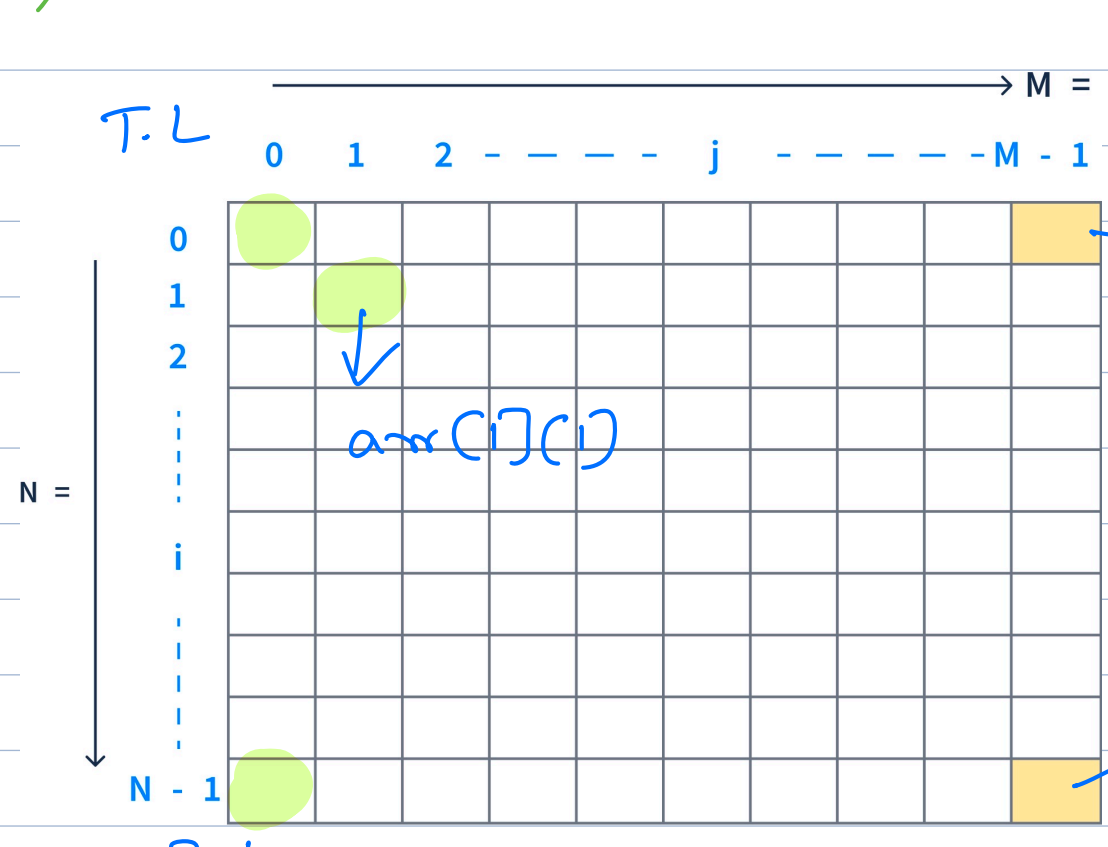
Declaration

name of the 2D array.

C++; `int arr[N][M]`

- `int` → datatype.
- `N` → rows
- `M` → columns

Java; `int[][] arr = new int[N][M];`



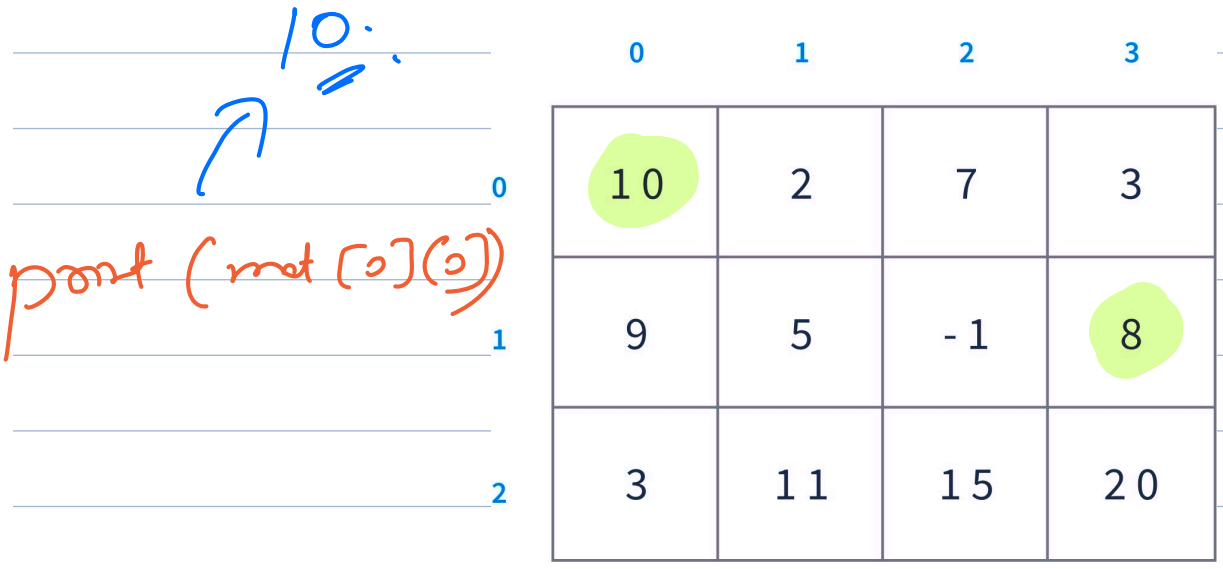
Q:- 1

(0, N-1)

Q-2.

(N-1, m-1).

ex:- int mat[3][4];



10:
print(mat[0][0])

8:
print(mat[1][3])

arr[5].
*(arr+5*4)

O(1).

Q:- 1. Given a matrix of size N * m. Print row wise sum.

Ex:-

	0	1	2	3
0	10	2	7	3
1	9	5	-1	8
2	3	11	15	20

O/P:-

22
21
49

Code

```
void printRowWiseSum(int mat[][], N, M) {
    int sum;
    for (rows = 0; rows < N; rows++) {
        int sum = 0;
        for (col = 0; col < M; col++) {
            sum += mat[row][col];
        }
        print(sum);
        print("\n");
    }
}
```

Quiz 3

T.C → O(N x m)
S.C → O(1).

Q:- 2. Given a matrix of size N * m. Print column wise sum.

ex:-

	0	1	2	3
0	10	2	7	3
1	9	5	-1	8
2	3	11	15	20

o/p:-

22
18
21
31

Code

```
void printColWiseSum(int mat[][], N, m) {
    for(col = 0; col < m; col++) {
        int sum = 0;
        for(row = 0; row < N; row++){
            sum += mat[row][col];
        }
        print(sum); print("\n");
    }
}
```

T.C → O(N * m)
S.C → O(1).

N = M

Q:-3. Given a square matrix. Print its principal diagonal & anti diagonal.

row++ ; col++ row++; col--;

T.L

	0	1	2	3
0	1 (0,0)	5	8	7
1	2	11 (1,1)	3	9
2	15	20	-3 (2,2)	18
3	30	40	50	60 (3,3)

4*4

T.R.

	0	1	2	3
0	1	5	8	7 (0,3)
1	2	11	3 (1,2)	9
2	15	20 (2,1)	-3	18
3	30 (3,0)	40	50	60

B.L.

4*4

B.R.

Principal Diagonal

Principal Antidiagonal

Code

```
void printDiagonals(int[][] mat, N){

    row = 0; col = 0;
    while(row < N){                      // N times, col < N
        print(mat[row][col]);
        row++; col++;
    }

    row = 0; col = N-1;
    while(row < N){                      // N times, col >= 0
        print(mat[row][col]);
        row++; col--;
    }
}
```

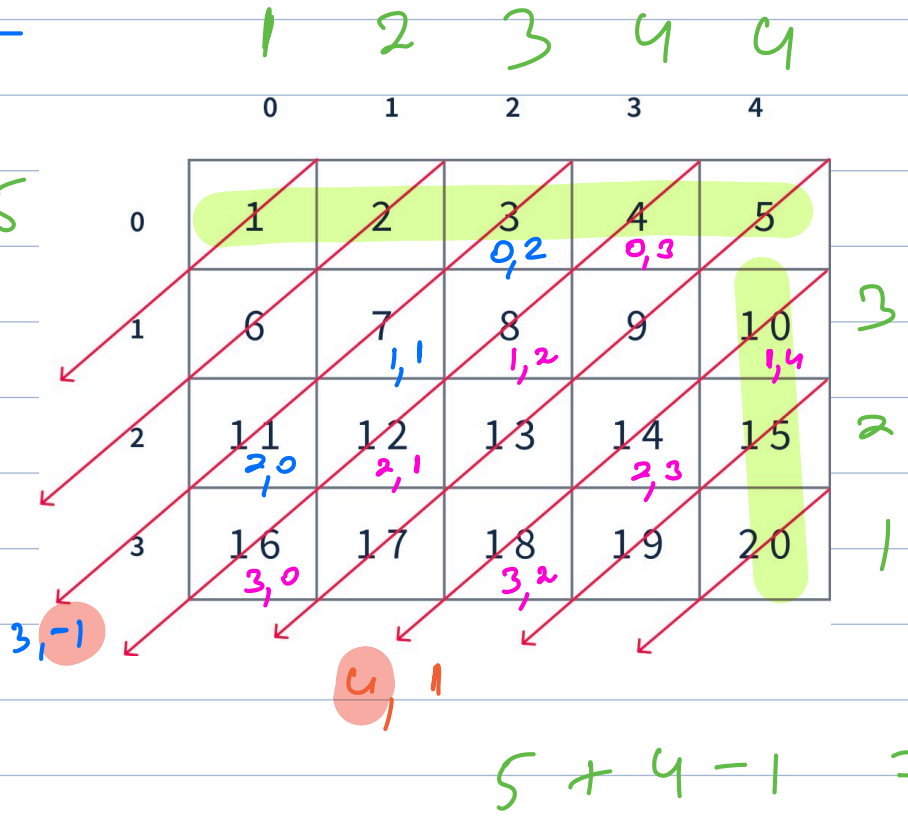
T.C → O(N)

S.C → O(1)

Quiz 4

Q:- 4. Print all the anti diagonals of a matrix.

Ex:-



o/p:-

1
2 6
3 7 11
4 8 12 16
5 9 13 17
10 14 18
15 19
20

8:

Ques 5

m + N - 1

Approach:-

Iterate on the first row & fix the S.P. of the m antidiagonals.

Iterate on that A.D from that S.P. until an invalid index is found.

Do the same for the last column.

N, m;

$0 \le row < N$

$0 \le col < m$

Code

i, j → S.P of an A.D.

```
void printAllDiagonals(int[][] mat, N, m){
    i = 0;
    for(j = 0; j < m; j++){
        row = i; col = j;
        while(row < N && col >= 0){
            print(mat[row][col]);
            row++; col--;
        }
        print("\n");
    }
    j = m-1;
    for(i = 1; i < N; i++){
        row = i; col = j;
        while(row < N && col >= 0){
            print(mat[row][col]);
            row++; col--;
        }
        print("\n");
    }
}
```

m A.D. are printed.

N-1 A.D.

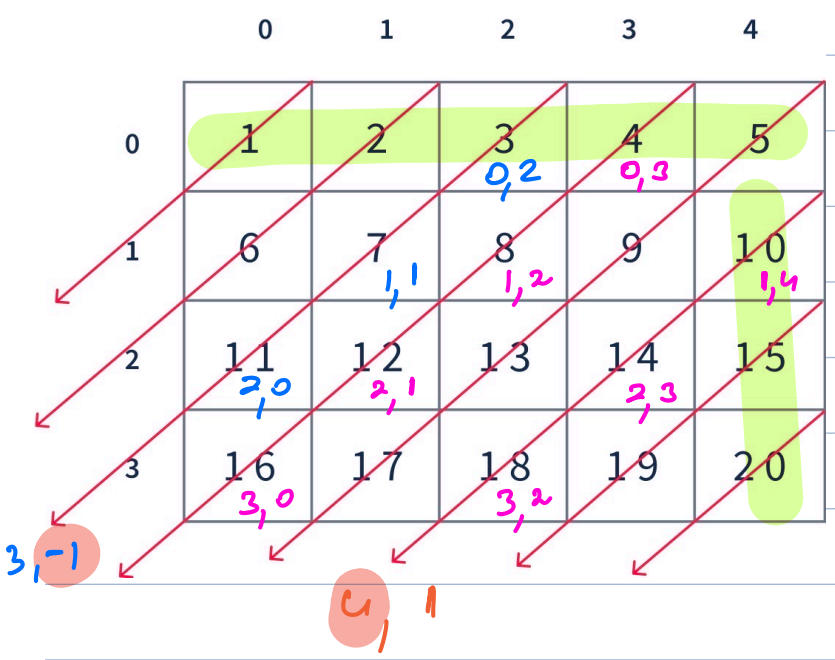
S.P. of an A.D.

S.P. of an A.D.

Quiz 7

T.C → O(m*N).
S.C → O(1).

10:22.



i j row col

0 0 0 3

1

2

3

-

N = 4

m = 5

o/p

1 2

2 6 2

3 7 11

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

S.P. of on A.D

3

3

3 * 4 ⟶ 4 * 3

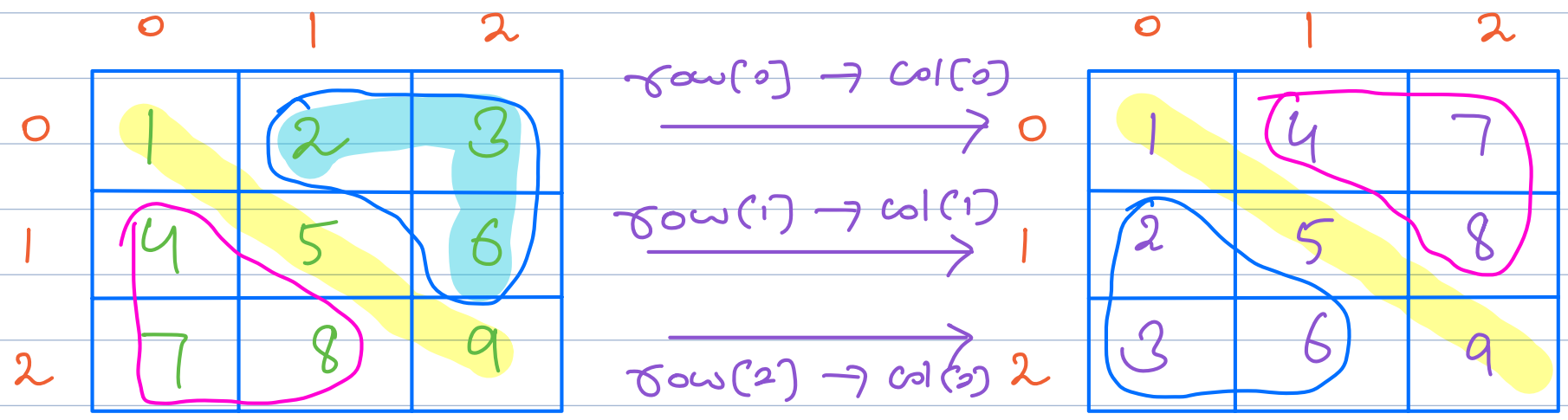
Q:- Convert a given square matrix to its transpose.

(in-place)

transpose ⟶ Rows & Cols gets interchanged.

3 * 3

ex:-



Input

o/p.

$\frac{(N^2 - N)}{2}$

$O(N^2)$

Observation :-

1. Principal Diagonal has remained same.
2. Elements above the P.D. have swapped their positions with below ie. elements $i, j \rightarrow j, i$

Quiz 8

TC ⟶ $O(N^2)$

SC ⟶ $O(1)$.

Approach :- Iterate on all the elements above the P.D. Swap mat[i][j] with mat[j][i].

Code

```
int[][] transpose (int mat[][], int N){
    for (row = 0; row < N; row++){
        for (col = row+1; col < N; col++){
            temp = mat[row][col];
            mat[row][col] = mat[col][row];
            mat[col][row] = temp;
        }
    }
    return mat;
}
```

$N = 3$.

	0	1	2
0	1	2 4	3 7
1	4 2	5	6 8
2	7 3	8 6	9

row = 0, col = 3

row ≠ 3

temp = 6;

mat[1][2] = 7

mat[2][1] = 6

row	col	
0	[1, N-1]	N-1
1	[2, N-1]	N-2
2	[3, N-1]	N-3.
⋮		
N-1		1.

$\frac{N(N+1)}{2}$ → Replace N with N-1

$\frac{(N-1)(N)}{2}$

Qu: Rotate a square matrix 90° clockwise.

	0	1	2
0	1	2	3
1	4	5	6
2	7	8	9

90° rotation clockwise →

	0	1	2
0	7	4	1
1	8	5	2
2	9	6	3

transpose ↓

	0	1	2
0	1	4	7
1	2	5	8
2	3	6	9

reverse each row →

O/P :-

Solution :- Can be broken down into 2 steps.

(1) Create a transpose of that matrix.

(2) Reverse each row of the transposed matrix.

Code.

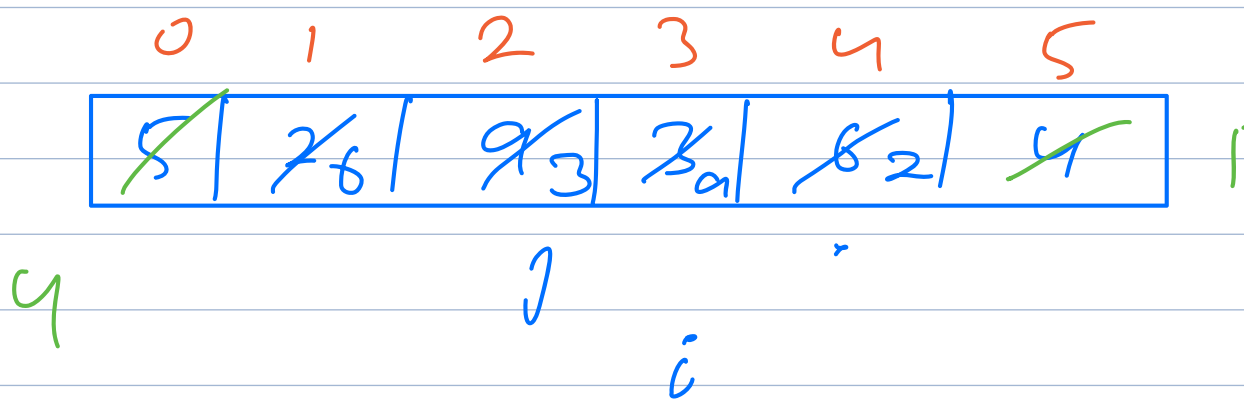
```
int[][] rotate90Clockwise(int[][] mat, N){
    mat = transpose(mat, N); } N^2

    for(row = 0; row < N; row++){
N^2/2 {     reverse(mat[row]);
    }
}

return mat;
```

TC → O(N²).
S.C → O(1).

Memory Management:



vector < int >

vector<vector<int>>

ArrayList<ArrayList<Integer>>

array = new ArrayList<>();