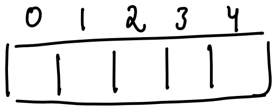
Today's Content:

→ 2D matrix syntax

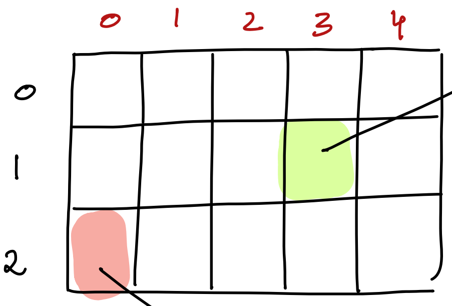
→ Problems

- Print values row by row
- Print value col by col
- A.length, A[0].length
- row wise sum
- col wise sum
- Check if A[][] is identity matrix or not
- Add 2 matrix return result
- Multiply 2 matrix return result
- Return transpose of given A[][] → {If time permits}

// Syntax:



→ int ar[] = new int[5]



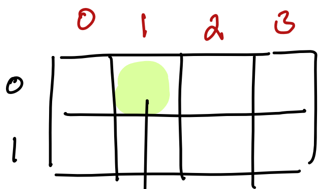
mat[1][3] → 1st row 3rd col element

noofrows

no: of cols

→ int mat[][] = new int[3][5]

mat[2][0] → 2nd row & 0th col element

// 

→ int mat[][] = new int[2][4]

mat[0][1]

// Need matrix with 4 rows & 5 columns

Writing ——— Java

int mat[4][5] — C++ | * |

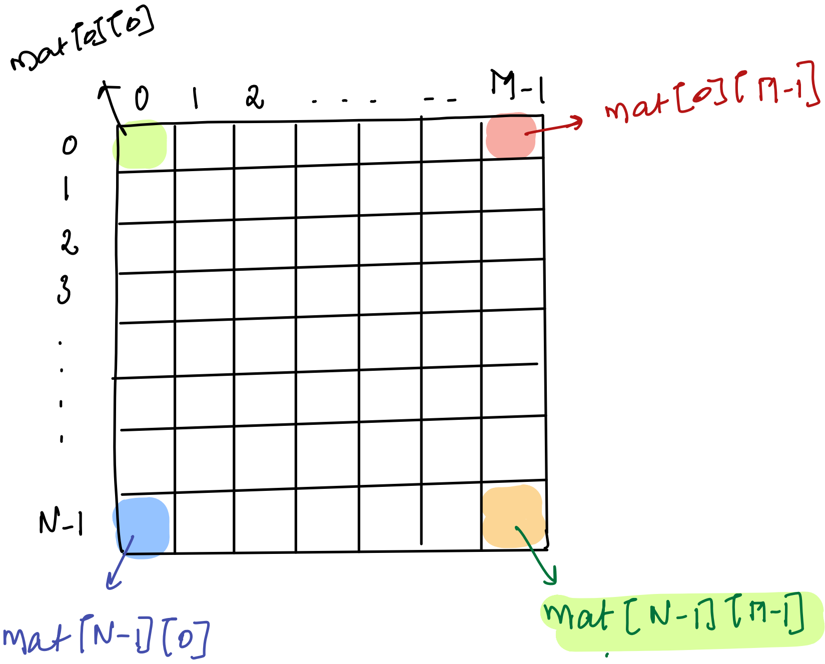
int mat[4,5] → * | * |

#rows #cols

int mat[][] = new int[4][5]

// Create matrix with N rows & M Columns

int A[][] = new int[N][M]



// int mat[][] = new int[3][4] → In Input data is given row by row

$$\begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 3 & 2 & 6 & 7 \\ 1 & 9 & 4 & 7 & 3 \\ 2 & 2 & 9 & 3 & 6 \end{array}$$

```
Scanner sc = new Scanner(System.in)
for(int i=0; i<3; i++){          → row number
    // We need to read data in ith row
    for(int j=0; j<4; j++){      → col number
        // read data in ith row & jth col
        mat[i][j] = sc.nextInt()
    }
}
```

// Given a matrix print data row by row

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

// mat[$\overset{\text{row}}{r}$][$\overset{\text{cols}}{c}$], print data row by row

for matrix input is already given

i → indicating row, j → indicating col

```
for(int i=0; i<r; i=i+1) {
    for(int j=0; j<c; j=j+1) {
        System.out.print(mat[i][j])
    }
    System.out.println()
}
```

// Given a matrix print data col by col

int mat[][] = new int[4][6] → output by output:

rows

col

Data is readed in mat

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

```
2 3 8 4
4 9 1 3
1 4 3 5
6 10 4 6
8 6 7 3
9 7 9 2
```

mat[4][1] → Error/ Out of Bounds/

↳ 4th row is not even present

mat[0][5] → 9

mat[0][6] → out of bounds

```
for( int j = 0; j < 6; j++){
    // print jth col
    for(int i = 0; i < 4; i++){
        // ith row & jth col
        print (mat[i][j])  → 1st ver
        print (mat[j][i])  → 2nd vers
    }
}
```

// Given a matrix, check if its identity matrix or not?

→ A matrix is said to be identity matrix if

- → matrix is square matrix → { # rows = # cols }
- → mat[i][j] == 1, if i == j
- → mat[i][j] == 0, if i ≠ j

If matrix fails even in 1 one of condition it's not a identity matrix

Ex1:

$$\begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 \end{array}$$

→ (YES)

$$\begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 \end{array}$$

→ mat[0][2] idealy data should be 0

Ex2:

$$\begin{array}{c|ccccc} & 0 & 1 & 2 & 3 & 4 \\ \hline 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 0 \\ 3 & 0 & 0 & 0 & 1 & 0 \\ 4 & 0 & 0 & 0 & 0 & 1 \end{array}$$

$$\begin{array}{c|ccccc} & 0 & 1 & 2 & 3 & 4 \\ \hline 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 0 \\ 3 & 0 & 0 & 0 & 1 & 1 \\ 4 & 0 & 0 & 0 & 0 & 0 \end{array}$$

mat[3][4] → 0 idealy it should be 0

mat[4][4] = 1, it should have been 1

```
boolean   isIdentity( int A[][])  → we are getting matrix as parameter
                                   → 2D matrix
                    ↳ how to get dimensions ?

    int r = A.length;   → it will give no: of rows

    int c = A[0].length  → it will given no: of columns
             ↳ 0th row

}
```

```
psvm() {

    int mat[][] = new mat[4][4]

    Reading Input

    boolean  ans  =  isIdentity(mat)  → passing mat[][] as a
                                         paremter
}
```

// Given 2 matrices A[][] & B[][] add & return new matrix

Note: We can only add 2 matrices if their dimensions are same

Note2: In given Question please assume both have same dimensions

Ex1: A[3][3] + B[3][3] → C[3][3]

$$\begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & 3 & 6 & 2 \\ 1 & 9 & 4 & 8 \\ 2 & 6 & 3 & 7 \end{array} \qquad \begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & 3 & 6 & 3 \\ 1 & 2 & 9 & 8 \\ 2 & 7 & 6 & 4 \end{array} \qquad \begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & 6 & 12 & 5 \\ 1 & 11 & 13 & 16 \\ 2 & 13 & 9 & 11 \end{array}$$

Ex1: A[3][4] + B[2][4] → * { not possible }

return type to return on int[][] matrix

```
int[][] Add( int A[][], int B[][]){
    int r = A.length
    int c = A[0].length
    int C[][] = new int[r][c]
    for(int i=0; i<r; i++){
        for(int j=0;j<c;j++){
            C[i][j] = A[i][j] + B[i][j]
        }
    }
    // We should return resultant matrix
    return C;
           └ entire matrix
}
```

// Given 2 matrices A[][] & B[][] multiply both of them & return final mat[][]?