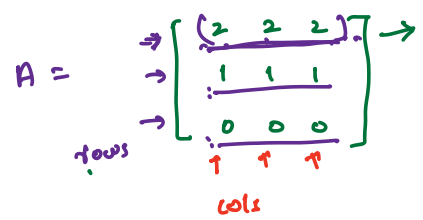
2D array → Create, take i/p and store, print 2D array, row by row, col by col, wave pattern.



A = [1, 2, 3, 4].
A.length - 1

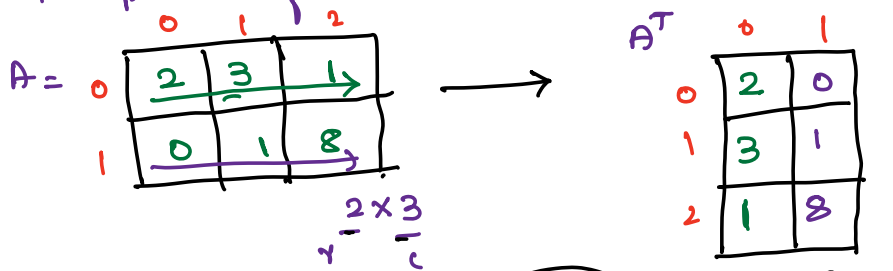
2DA = [A1, A2, A3];

row = A.length; // rn
col = A[0].length; // A[row-1];

2 x 3 => rows = 2
cols = 3.

int[][] A = rows = A.length
Col = A[0].length.

Q) Given a matrix A. write a function that returns the transpose of a matrix.



2 x 3 = 3 x 2
r c r c

Val	A r, c	A^T r, c
2	0, 0	0, 0
3	0, 1	1, 0
1	0, 2	2, 0
0	1, 0	0, 1
1	1, 1	1, 1
8	1, 2	2, 1

A i, j A^T (row, col)

i → j
j → i

row → col
col → row

21:40

```
static int[][] transpose(int[][] mat){
    int N = mat.length;
    int M = mat[0].length;
    int trans[][] = new int[M][N];
    for(int row=0; row<N; row++){
        for(int col=0; col<M; col++){
            trans[col][row] = mat[row][col];
        }
    }
    return trans;
}
```

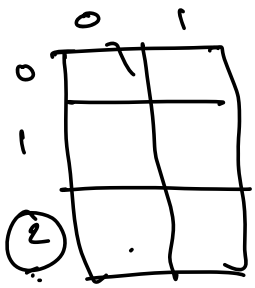
A → A^T

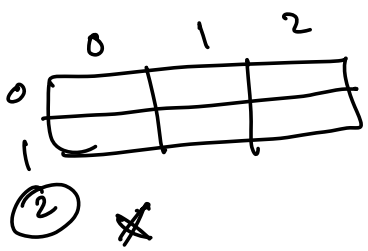
$N \times m$ → $M \times N$

col - row

row - col.

2 × 3 ↓ 3 × 2





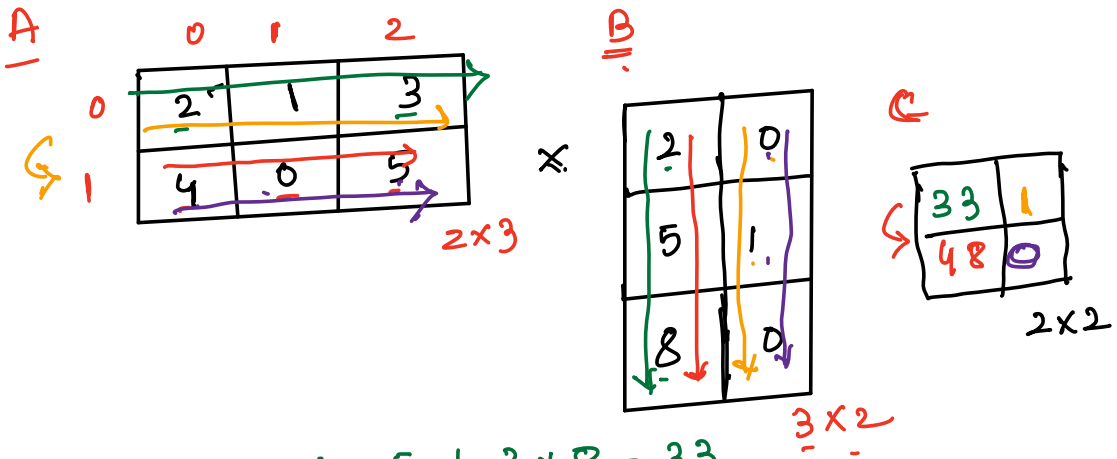
Q2) Given A and B matrices. write a function to return a Matrix C = A * B (matrix multiplication)

A →

B →

C → A * B. 2 x 3 = 2 * 3

✗ int C[][] = A * B; ✗



2 x 2 + 1 x 5 + 3 x 8 = 33

2 x 0 + 1 x 1 + 3 x 0 = 1

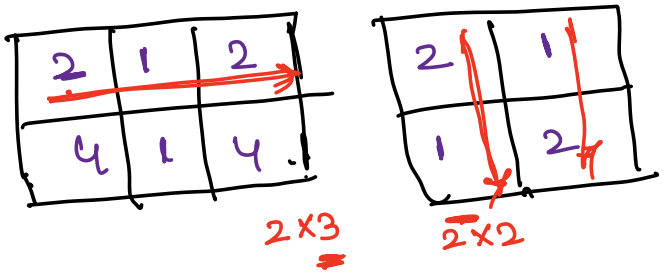
4 x 2 + 0 x 5 + 5 * 8 = 48

4 x 0 + 0 x 1 + 5 x 0 = 0

A * B = C

2 x 3 * 3 x 2 = 2 x 2

$r_1 \times c_1$ $r_2 \times c_2$ = $r_1 \times c_2$



$\Rightarrow$ 2×2

$2 \times 2 + 1 \times 1 + 2 \times$ [6] NO. $\rightarrow$

Matrix multiplication is not possible.

A * B $\Rightarrow$ No

2×2 4×2

r_1 c_1 r_2 c_2

$c = = r_2$

N = 2
M = 3

A B

N×M . N×M

A B

N×M M × anything

anything $\times$

A * B

3×4 4×5

Imp Note

i) To multiply two matrix A and B of size $r_1 \times c_1$ & $r_2 \times c_2$

$$\boxed{c_1 == r_2}$$

ii) Resultant matrix of A and B of size $r_1 \times c_1$ and $r_2 \times c_2$ will be of size

$$\boxed{r_1 \times c_2}$$

Quiz 2

A(3 x 4) B(4 x 7)

$\boxed{c_1 == r_2}$ ✓ Yes

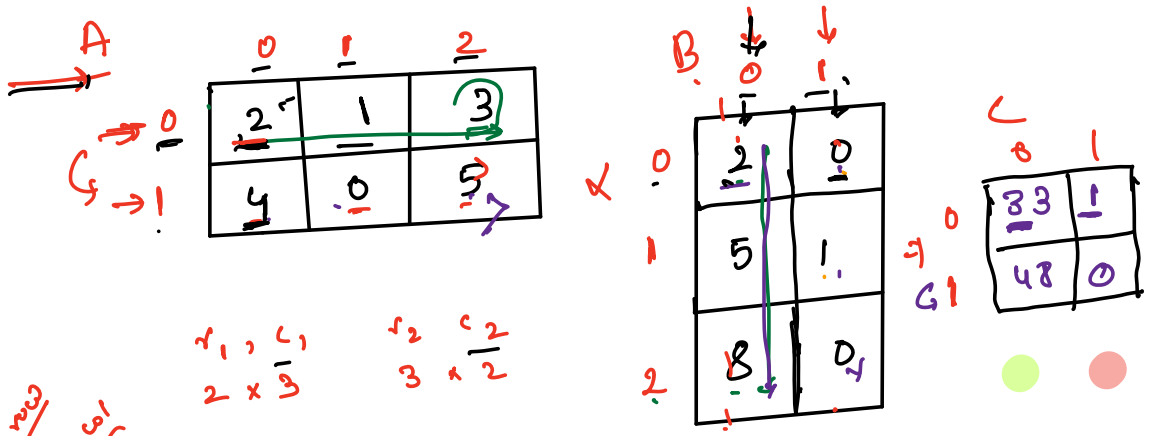
A(3 x 7) B(4 x 7) No.

A(2 x 2) B(2 x 2) Yes

A(3 x 4) B(4 x 7)

B * A

4 x 7 3 x 4 No



C[0][0] = A[0][0] * B[0][0] + A[0][1] * B[1][0] + A[0][2] * B[2][0]

C[0][1] = A[0][0] * B[0][1] + A[0][1] * B[1][1] + A[0][2] * B[2][1]

C[1][0] = A[1][0] * B[0][0] + A[1][1] * B[1][0] + A[1][2] * B[2][0]

C[1][1] = A[1][0] * B[0][1] + A[1][1] * B[1][1] + A[1][2] * B[2][1]

```
for(int row=0; row<r1; row++){
    for(int col=0; col<c2; col++){
        int sum = 0;
        for(int i=0; i<c1; i++){        // 0, 1, 2
            sum = sum + (A[row][i] * B[i][col]);
        }
        c[row][col] = sum;
    }
}
return c;
```

row = 0, 1

col = 0, 1

i, 0 1 2

A → row - constant

col - changing

B → row - changing

col → constant

Identity matrix

Doubt section

A [1, 2, 3, 4] → n=4

0 1 2 3

change the size

=> 1

k = 5

[1, 5, 2, 3, 4]

int new A[n+1]

[1, 2, 3, 4, 0]

[1, 5 →

① - A

② = B[n+1]

③ 0 — idx

④ idx → nil

B(0)

= -

AC)

idx → remve

[1, 2, 3, 4, 5]

if (idx == i)

continue

Then B(i) = A(i)

Inverse

A[2,0,1] =

A = [1 2 0]

A = 3 4 1 2 0

B[0] = A[3]

B[1] =

B[2] =

0 1 2 3

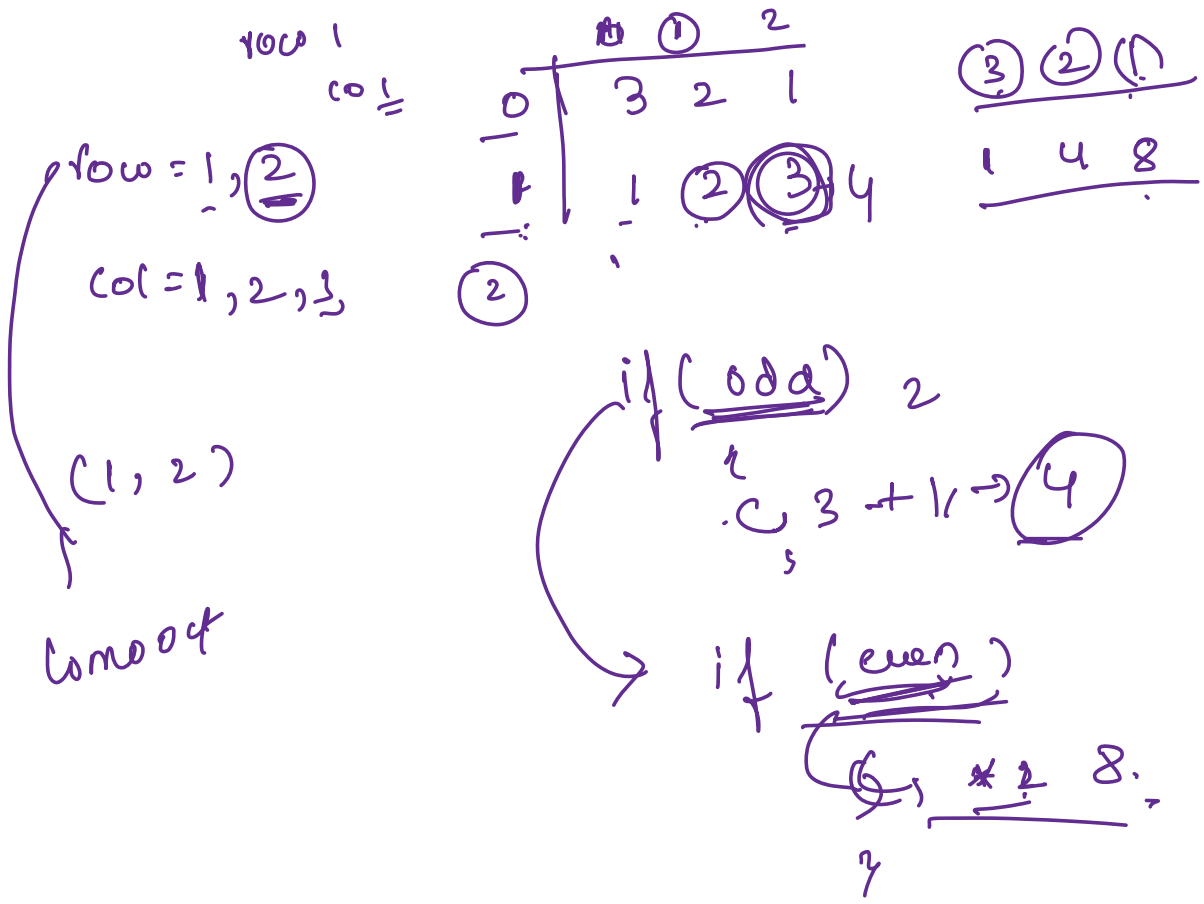
3 1 0 2

[2 1 3 0]

[1,0]

0,

```
int n = A.length;
int B = new int[n]
for(int i=0; i<n; i++)
    B[i] = A[A[i]];
}
```



static void printTwoDArray(int[][] mat) {

print all elements row by row

}

int mat[][].

int[] A => int A[].

Main() {

int[][] mat = new int[3][4]

A = {

int[] A = new int[5];

printArray(A);

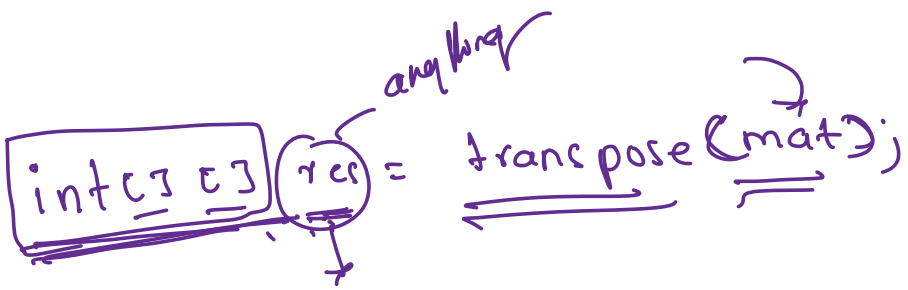
printNum(B);

print2Darray(A);

print2Darray(5);

A.length;

A[0].length;



int sq = squareNum(5);

int A = new int[5];

arr = new int[6];

1 Ref int

2

- https://www.interviewbit.com/snippet/0c8c2cb98106c79b7e00/