Today's agenda

↳ Intro to 2D Arrays

↳ Print matrix row wise

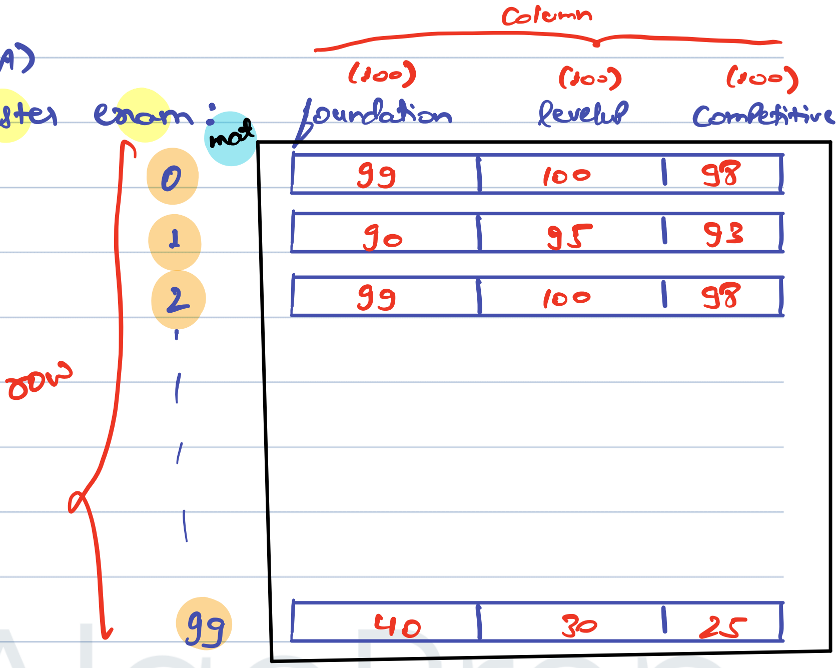
↳ Print matrix Colwise

↳ Print matrix in wave form

↳ Arraylist → {Dynamic array}

// Intro

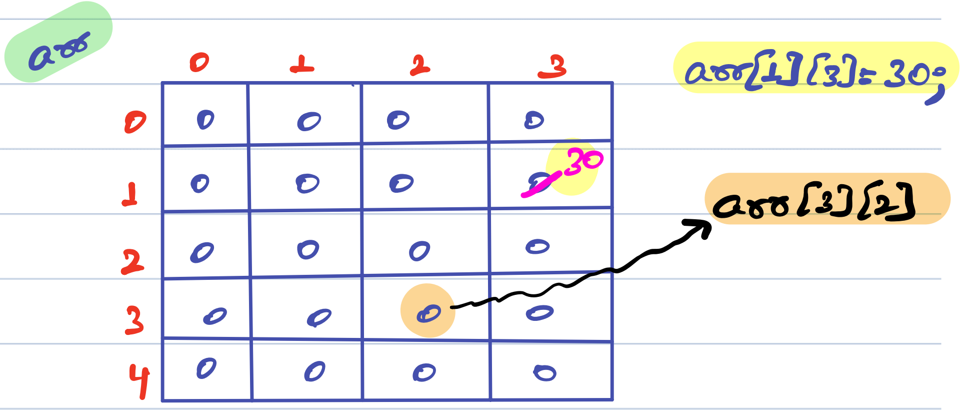
↳ mid (DSA) semester exam :



	(100) foundation	(100) levelup	(100) Competitive
0	99	100	98
1	90	95	93
2	99	100	98
⋮			
99	40	30	25

// Syntax

```
int[][] arr = new int[5][4];
```



	0	1	2	3
0	0	0	0	0
1	0	0	0	~~0~~ 30
2	0	0	0	0
3	0	0	0	0
4	0	0	0	0

arr[1][3] = 30;

arr[3][2]

5*4 = 20 elements

Q) Print matrix rowwise

↳ Print the given mat[n][m] rowwise.

arr[4][5]:

	0	1	2	3	4
0	10	20	30	40	50
1	60	70	80	90	100
2	110	120	130	140	150
3	160	170	180	190	200

4x5

```
10  20  30  40  50
60  70  80  90  100
110 120 130 140 150
160 170 180 190 200
```

	i j		
0,0	1,0	2,0	3,0
j+1	j+1		
0,1	1,1	2,1	3,1
j+1	j+1		
0,2	1,2	2,2	3,2
j+1	j+1		
0,3	1,3	2,3	3,3
j+1	j+1		
0,4	1,4	2,4	3,4

```
for (int i=0 ; i<n ; i++){
    for( j=0; j<m; j++){
        System.out.print(arr[i][j] + " ");
    }
    System.out.println();
}
```

T.C: O(n*m)

S.C: O(1)

n=4
m=5

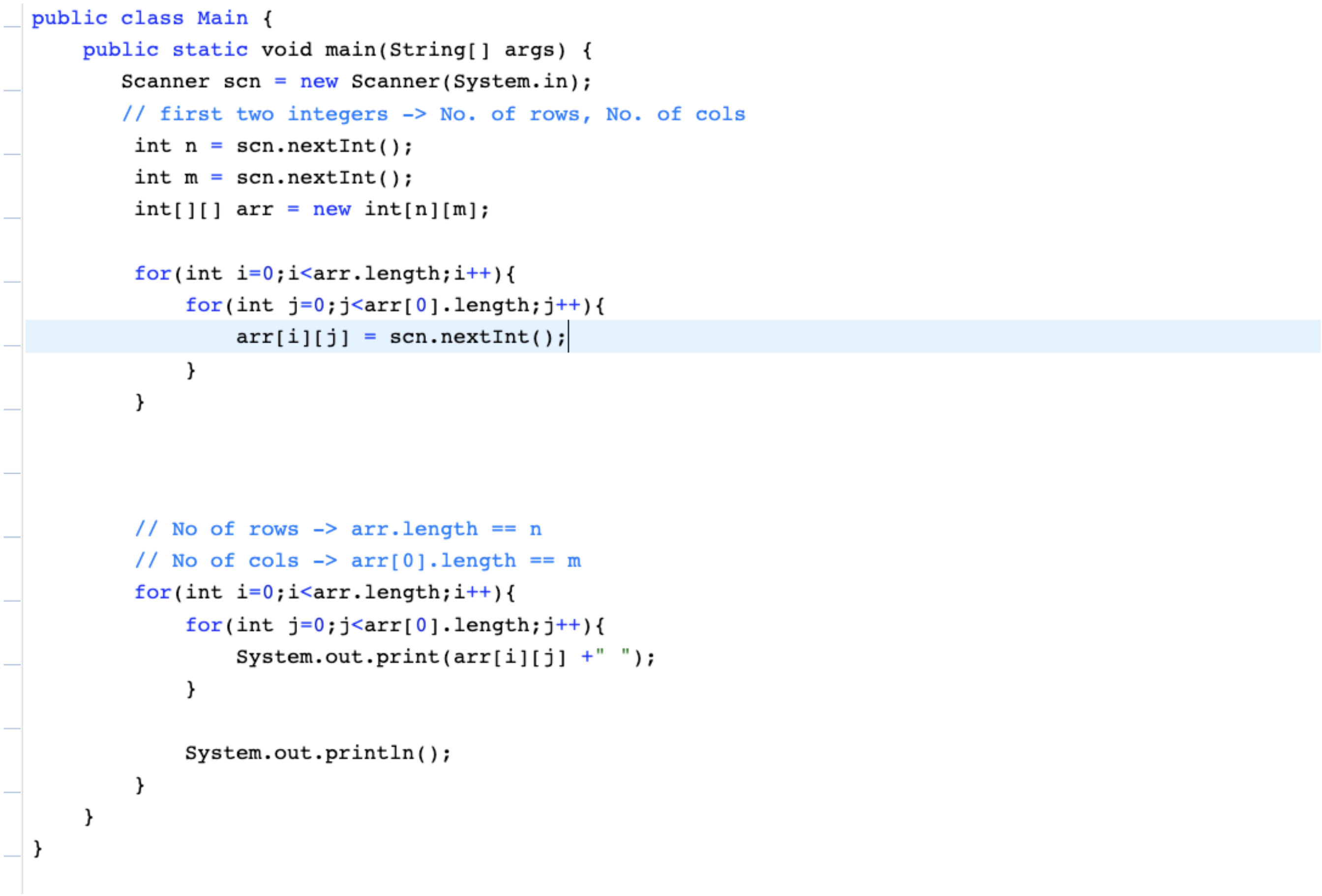
```
for (int i=0 ; i<n ; i++){
    for( j=0; j<m ; j++){

        System.out.point(arr[i][j] + " ");

    }
    System.out.pointln();
}
```

arr	0	1	2	3	4
0	10	20	30	40	50
1	60	70	80	90	100
i → 2	110	120	130	140	150
3	160	170	180	190	200

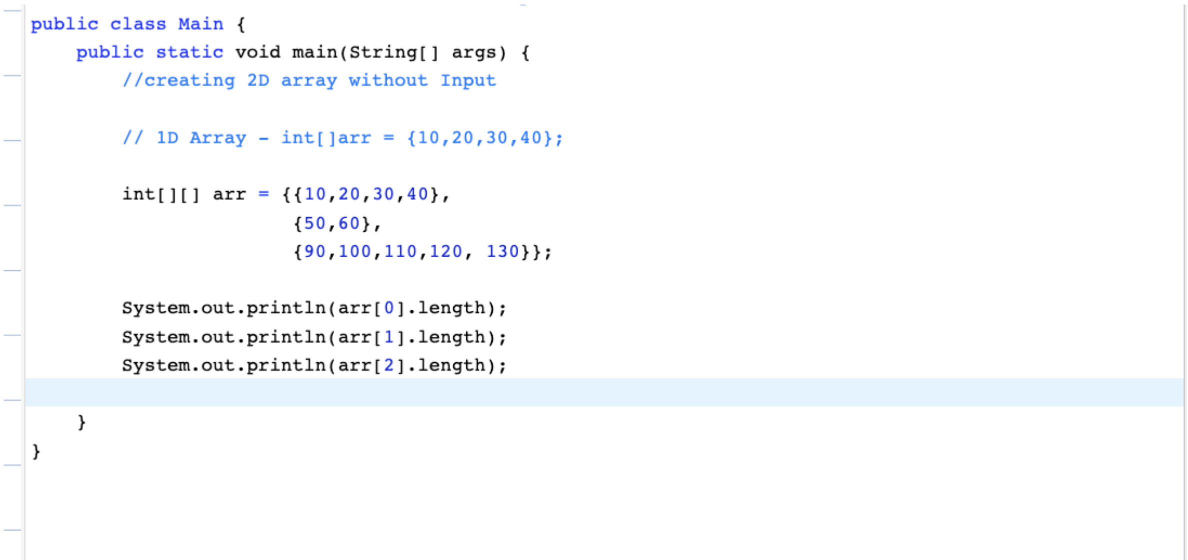
4x5

```
10   20   30   40   50
60   70   80   90   100
110  120  130  140  150
160  170  180  190  200
```

i	j
0	0 1 2 3 4 5 exit
1	0 1 2 3 4 5 exit
2	0 1 2 3 4 5 exit
3	0 1 2 3 4 5 exit
4 exit	



```
public class Main {
    public static void main(String[] args) {
       Scanner scn = new Scanner(System.in);
       // first two integers -> No. of rows, No. of cols
        int n = scn.nextInt();
        int m = scn.nextInt();
        int[][] arr = new int[n][m];

        for(int i=0;i<arr.length;i++){
            for(int j=0;j<arr[0].length;j++){
                arr[i][j] = scn.nextInt();
            }
        }



        // No of rows -> arr.length == n
        // No of cols -> arr[0].length == m
        for(int i=0;i<arr.length;i++){
            for(int j=0;j<arr[0].length;j++){
                System.out.print(arr[i][j] +" ");
            }

            System.out.println();
        }
    }
}
```



```
public class Main {
    public static void main(String[] args) {
        //creating 2D array without Input

        // 1D Array - int[]arr = {10,20,30,40};

        int[][] arr = {{10,20,30,40},
                       {50,60},
                       {90,100,110,120, 130}};

        System.out.println(arr[0].length);
        System.out.println(arr[1].length);
        System.out.println(arr[2].length);

    }
}
```

Break till 9:34 pm

Q) Print matrix Colwise

↳ Print the given mat[n][m] colwise.

arr[4][5]:

$$\begin{array}{c|ccccc} & 0 & 1 & 2 & 3 & 4 \\ \hline 0 & 10 & 20 & 30 & 40 & 50 \\ 1 & 60 & 70 & 80 & 90 & 100 \\ 2 & 110 & 120 & 130 & 140 & 150 \\ 3 & 160 & 170 & 180 & 190 & 200 \end{array}$$

```
10 60 110 160
20 70 120 170
30 80 130 180
40 90 140 190
50 100 150 200
```

```
        i j
00      0 1      0 2      0 3      0 4
i++
10      1 1      1 2      1 3      1 4
i++     i++
20      2 1      2 2      2 3      2 4
i++     i++
30      3 1      3 2      3 3      3 4
```

T.C: O(m*n)

S.C: O(1)

```
for (int j=0; j<m; j++){
    for (int i=0; i<n; i++){
        System.out.print(arr[i][j] + " ");
    }
    System.out.println();
}
```

Q) Print matrix in wave form

↳ Print the given mat[n][m] in wave form.

arr[4][5]:

	0	1	2	3	4	
0	10	20	30	40	50	L-R → 10 20 30 40 50
1	60	70	80	90	100	R-L → 100 90 80 70 60
2	110	120	130	140	150	L-R → 110 120 130 140 150
3	160	170	180	190	200	R-L → 200 190 180 170 160

L-R → 0, 2, 4 . . . (even row)

R-L → 1, 3, 5 . . . (odd row)

even no: No. divisible by 2

```
for (int i=0 ; i<n ; i++){
    if (i%2==0){
        for( j=0; j<m; j++) {
            System.out.print(arr[i][j] + " ");
        }
    }
    else {
        for( j=m-1; j>=0; j--) {
            System.out.print(arr[i][j] + " ");
        }
    }
    System.out.println();
}
```

T.C: O(m*n)

S.C: O(1)

	0	1	2	3	4
0	10	20	30	40	50
1	60	70	80	90	100
2	110	120	130	140	150
3	160	170	180	190	200

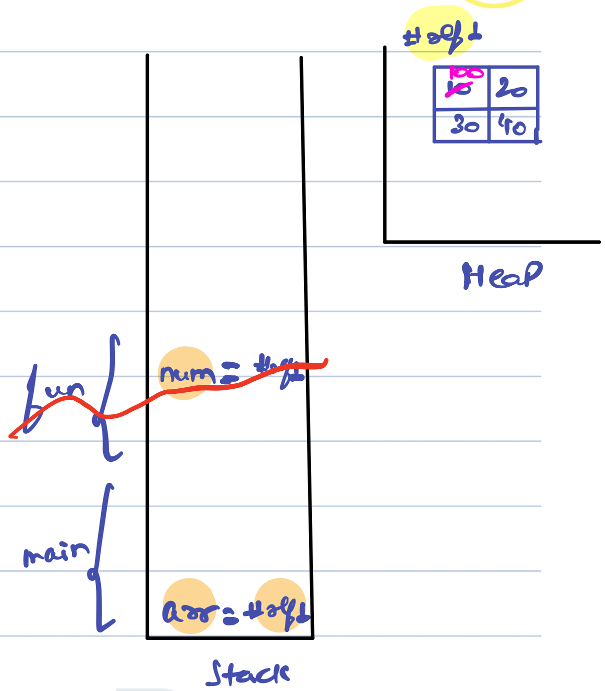
4×5

```
for (int i=0 ; i<n ; i++){
    if (i%2==0){
        for( j=0; j<m; j++){
            System.out.point(arr[i][j] + " ");
        }
    }
    else {
        for( j=m-1; j>=0; j--){
            System.out.point(arr[i][j] + " ");
        }
    }
    System.out.pointln();
}
```

i	j
0	0 1 2 3 4 5 ↳exit
1	4 3 2 1 0
2	0 1 2 3 4 5

```
10   20   30   40   50
→ 100  90   80   70   60
```

```
void  main () {

  int [] [] arr = { {10,20} , {30,40}};

      System.out.println(arr [0][0]); →10
        fun(~~arr~~ #off1);
      System.out.println(arr [0][0]); →100

}

Public Static void fun (int[] [] num){

      num[0][0] = 100;

}
```



int [] arr = new int [10];

// Arraylist or Dynamic array

ArrayList <Integer> list = new ArrayList<>();

name

0	1	2
10	20	30

→ list.add(10);

→ list.add(20);   → T.C: O(1)

→ list.add(30);

// To access index

S.O.P(list.get(1)); → 20

↳ T.C: O(1)

// update the value

lis~~t~~.get(2) = ~~2~~00;

list.set(1, 200);

↳ T.C: O(1)

// No. of element

list.size();

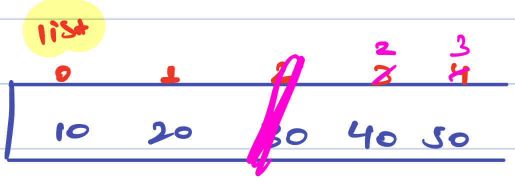
↳ T.C: O(1)

// remove index

List.remove(idx);

remove last idx → O(1)

remove 0th idx → O(N)



list.remove(2);

→ Arraylist in the background is just Array being used smartly.

// Doubt session

maxdiff 2: max(arr[i] - arr[j] + i - j)

maxdiff 3: max(arr[i] - arr[j] + j - i)

maxdiff 4: max(|arr[i] - arr[j]| + i - j)

↙

max(arr[i] - arr[j], -(arr[i] - arr[j]))

⇓

max(arr[i] - arr[j], arr[j] - arr[i])

⇓

ind

max {
~~arr[i] - arr[j] + i - j)~~

~~arr[j] - arr[i] + i - j~~ ind

i & j swap

arr[i] - arr[j] + j - i

$|x| = \max(x, -x)$

$|8| = \max(8, -8) = 8$

$|-8| = \max(-8, -(-8)) = 8$

{10   20   13   3 $\overset{di}{1}$ 35}

max = ~~-∞~~ ~~10~~ 20

2nd max = ~~-∞~~ ~~-∞~~ ~~10~~ 13