

Transpose of Matrix of N*N

Problem	Submissions	Leaderboard	Discussions
---------	-------------	-------------	-------------

Sophie is a computer science student who loves solving programming challenges. One day, her professor gives her an interesting task - to **write a program that finds the transpose of a square matrix** of size **N * N**.

NOTE:- After answering the question, attempt the related question in the linked resource to improve your understanding of this question. Click [here](#)

Transpose · $n \times n$ · $n=4$

	0	1	2	3
0	X	2	3	4
1	1	2	2	2
2	3	3	3	3
3	4	4	4	4

ans.

1	2	3	4
1	2	3	4
1	2	3	4
1	2	3	4

rows \longleftrightarrow cols

idx
0,2
2,0

$(i, j) \rightarrow (j, i)$

$ij \rightarrow ji$

	0	1	2	3
0	1	1	1	1
1	2	2	2	2
2	3	3	3	3
3	4	4	4	4

$i=0$
1 2 3

$i=1$
1 2 3

$i=2$
2 3

i, j ji
 $(0,1)$ $1,0$
 $(0,2)$ $2,0$
 $(0,3)$ $3,0$
 $(1,2)$ $2,1$
 $(1,3)$ $3,1$
 $(2,3)$ $3,2$

Sample Input 0

```
4
1 1 1 1
2 2 2 2
3 3 3 3
4 4 4 4
```

Sample Output 0

```
1 2 3 4
1 2 3 4
1 2 3 4
1 2 3 4
```

```

1 import java.io.*;
2 import java.util.*;
3
4 public class Solution {
5
6     public static void main(String[] args) {
7         Scanner scn = new Scanner(System.in);
8         int n = scn.nextInt();
9         int [][] A = new int[n][n];
10        for(int i = 0; i < n; i++){
11            for(int j = 0; j < n; j++){
12                A[i][j] = scn.nextInt();
13            }
14        }
15
16        //transpose
17        for(int i = 0; i < n; i++){
18            for(int j = i + 1; j < n; j++){
19                int tmp = A[i][j];
20                A[i][j] = A[j][i];
21                A[j][i] = tmp;
22            }
23        }
24
25        //print
26        for(int i = 0; i < n; i++){
27            for(int j = 0; j < n; j++){
28                System.out.print(A[i][j] + " ");
29            }
30            System.out.println();
31        }
32    }
33 }

```

n=3

		0		
		1 ✓	2 ✓	3 ✓
0		1 ✓	2	2
1		1	2	2
2		1	3	3

$i=0$
 $j=0 \times 2$

$i=1$
 $\underline{j=0}$

Transpose → $n \times n$

Rotate The Matrix by 90 Degree

Problem

Submissions

Leaderboard

Discussions

Take a square matrix of size $n \times n$ as input, and rotate the matrix by 90 degree.

$n=3$

1	2	3
4	5	6
7	8	9

90°

1. Transpose

1	4	7
2	5	8
3	6	9

2. reverse all rows.

↳ same array.

7	4	1
8	5	2
9	6	3

rotate 90°.

Sample Input 0

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

Sample Output 0

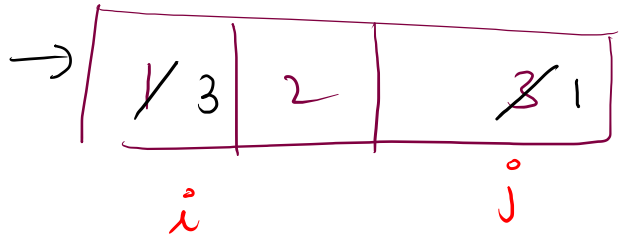
```
7 4 1
8 5 2
9 6 3
```

Reverse rows

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

\rightarrow row = 0

$i < j$



row = 0

$i = 0$

$j = n - 1$

$tmp = A[0][0] = 1$
 $A[0][0] = A[0][2]$
 $A[0][2] = 1$

$tmp = A[row][i]$
 $A[row][i] = A[row][j]$
 $A[row][j] = tmp$

```

1 import java.io.*;
2 import java.util.*;
3
4 public class Solution {
5     public static void print(int [][] A){
6         int n = A.length;
7         for(int i = 0; i < n; i++){
8             for(int j = 0; j < n; j++){
9                 System.out.print(A[i][j] + " ");
10            }
11            System.out.println();
12        }
13    }
14    public static void transpose(int [][] A){
15        int n = A.length;
16        for(int i = 0; i < n; i++){
17            for(int j = i+1; j < n; j++){
18                int tmp = A[i][j];
19                A[i][j] = A[j][i];
20                A[j][i] = tmp;
21            }
22        }
23    }
24    public static void reverseRows(int [][] A){
25        int n = A.length;
26        for(int row = 0; row < n; row++){
27            //logic to reverse single row
28            int i = 0;
29            int j = n-1;
30            while(i < j){
31                int tmp = A[row][i];
32                A[row][i] = A[row][j];
33                A[row][j] = tmp;
34                i++;
35                j--;
36            }
37        }
38    }
39 }

```

```

40 public static void main(String[] args) {
41     Scanner scn = new Scanner(System.in);
42     int n = scn.nextInt();
43     int [][] A = new int[n][n];
44     for(int i = 0; i < n; i++){
45         for(int j = 0; j < n; j++){
46             A[i][j] = scn.nextInt();
47         }
48     }
49
50     //1. Transpose
51     transpose(A);
52     //2. Reverse rows
53     reverseRows(A);
54     //3. print ans
55     print(A);
56 }
57 }

```

$$n^2 + n^2 + n^2$$

$$O(n^2)$$

Rotate by 180°.

Sample Input 0

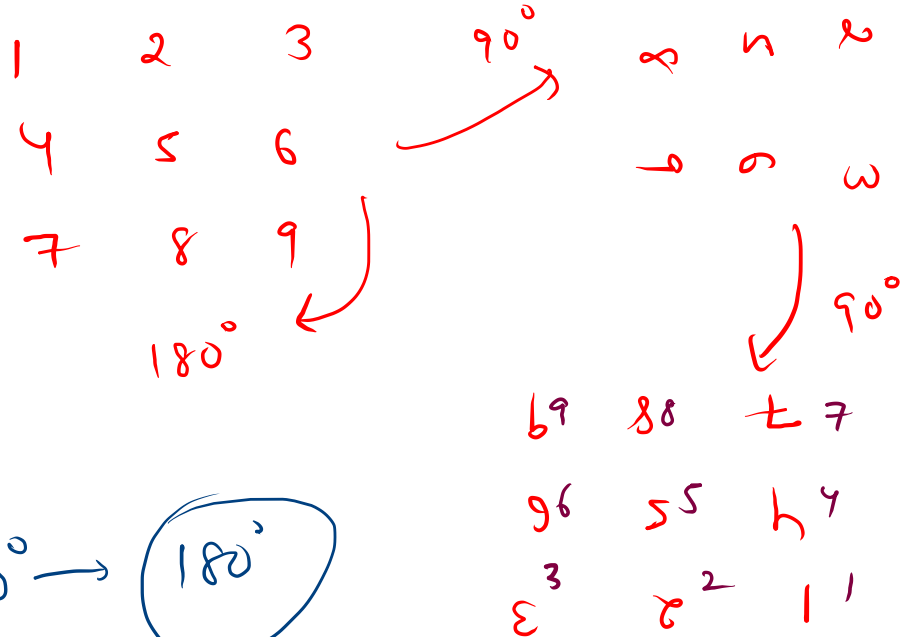
```
3
1 2 3
4 5 6
7 8 9
```

Sample Output 0

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

?

$90^\circ + 90^\circ \rightarrow 180^\circ$



Print row wise with condition

Sample Input 0

```
3
3
3 1 2
3 0 2
4 5 4
```

Sample Output 0

```
3 1 2
2 0 3
4 5 4
```

Problem

Submissions

Leaderboard

Discussions

Once upon a time, there was a programmer named Alex who was given the task of printing a **matrix row-wise**. However, there was a twist - the **even-numbered rows** had to be printed from **left to right**, and the **odd-numbered rows** had to be printed from **right to left**.

help Alex and write a program that would **iterate** through each **row** of the **matrix** and check if it was an **even or odd row**. If it was an **even row**, the program would traverse it from **left to right**, and if it was an **odd row**, the program would traverse it from **right to left**.

4 → m
3 → n

→

0	3	1	2
1	3	0	2
2	4	5	14
3	9	8	7

row → even
L to R

odd
R to L

row = 0

3 1 2

row = 1

2 0 3

row = 2

4 5 14

row = 3

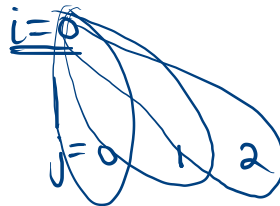
7 8 9

```

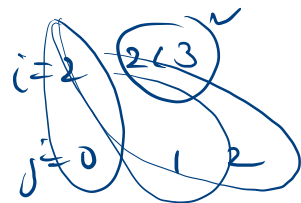
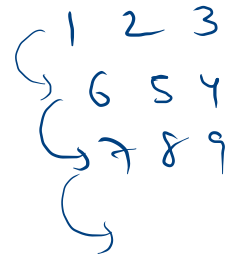
1 import java.io.*;
2 import java.util.*;
3 public class Solution {
4     public static void main(String[] args) {
5         Scanner scn = new Scanner(System.in);
6         int m = scn.nextInt();
7         int n = scn.nextInt();
8         int [][] A = new int[m][n];
9         for(int i = 0; i < m; i++){
10             for(int j = 0; j < n; j++){
11                 A[i][j] = scn.nextInt();
12             }
13         }
14         //print
15         for(int i = 0; i < m; i++){
16             if(i % 2 == 0){
17                 //L to R
18                 for(int j = 0; j < n; j++){
19                     System.out.print(A[i][j] + " ");
20                 }
21             }
22             else{
23                 // R to L
24                 for(int j = n-1; j >= 0; j--){
25                     System.out.print(A[i][j] + " ");
26                 }
27             }
28             System.out.println();
29         }
30     }
31 }

```

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



$0 < 3$ ✓



$i=3$ $3 < 3$ ✗

Convert 1-D Array to 2-D Array

Problem

Submissions

Leaderboard

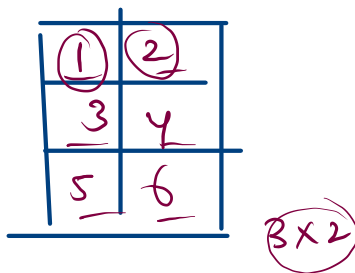
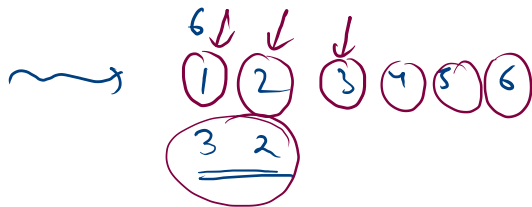
Discussions

Take an **array** of size **N** as input, representing a 1-D array.

There are many possible factors of N, for eg: **$p * q = N$** .

Now take **p and q** as input and print the 2-D array with dimensions as **$p * q$** .

Note: It is **guaranteed** that a 2-D array will be formed



Sample Input 0

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

Sample Output 0

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

1. create 2D
 $p * q$

```

1 import java.io.*;
2 import java.util.*;
3 public class Solution {
4     public static void main(String[] args) {
5         Scanner scn = new Scanner(System.in);
6         int n = scn.nextInt();
7         int [] one = new int[n];
8         for(int i = 0; i < n; i++){
9             one[i] = scn.nextInt();
10        }
11        int p = scn.nextInt();
12        int q = scn.nextInt();
13        //solve
14        int [][] two = new int[p][q];
15        int idx = 0;
16        for(int i = 0; i < p; i++){
17            for(int j = 0; j < q; j++){
18                two[i][j] = one[idx];
19                idx++;
20            }
21        }
22        //print
23        for(int i = 0; i < p; i++){
24            for(int j = 0; j < q; j++){
25                System.out.print(two[i][j] + " ");
26            }
27            System.out.println();
28        }
29    }
30 }
31 }

```

eg. $n=6$

one $\begin{matrix} 3 & 6 & 4 & 7 & 1 & 4 \\ 0 & 1 & 2 & 3 & 4 & 5 \end{matrix}$

$p=3$ $q=2$

$i=0$

$i=1$

$i=2$

3	6
4	7
1	4

$p \times q$
 3×2

idx = 0 1 2 3

one[0]

one[1]

one[2]

one[3]