

2-D

Transpose of Matrix of N*N (16 July)

Problem

Submissions

Leaderboard

Discussions

Write a program to find the transpose of a square matrix of size N*N. Transpose of a matrix is obtained by changing rows to columns and columns to rows.

n = 4
4 x 4

A →

	0	1	2	3
0	15	12	19	20
1	5	9	11	13
2	1	7	4	9
3	21	25	52	57

→

	0	1	2	3
0	15	5	1	21
1	12	9	7	25
2	19	11	4	52
3	20	13	9	57

for i = 0 → ~~swap (0,0)~~
 i = 1 → (1,0)
 i = 2 → (2,0), (2,1)
 i = 3 → (3,0), (3,1), (3,2)
 0 1 2 3

0	15			
1		9		
2			8	
3				57

0 → (0,1), (0,2), (0,3)
 1 → (1,2), (1,3)
 2 → (2,3)

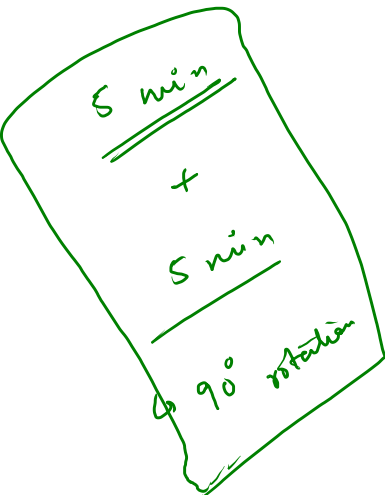
for (int i = 1; i < n; i++)
 {
 for (int j = 0; j < i; j++)

swap the elements

}

for (int i = 0; i < n; i++)
 {
 for (int j = i + 1; j < n; j++)
 {
 swap the elements
 }
 }

3 mins



```
public class Solution {
    public static void transpose(int arr[][],int n){
        for(int i=1;i<n;i++){
            for(int j=0;j<i;j++){
                int temp = arr[i][j];
                arr[i][j] = arr[j][i];
                arr[j][i] = temp;
            }
        }
    }

    public static void displayArray(int arr[][],int n){
        for(int i=0;i<n;i++){
            for(int j=0;j<n;j++){
                System.out.print(arr[i][j] + " ");
            }
            System.out.println();
        }
    }

    public static void main(String[] args) {
        /* Enter your code here. Read input from STDIN. Print output to STDOUT */
        Scanner scn = new Scanner(System.in);
        int n =scn.nextInt();
        int arr[][] = new int[n][n];

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

        transpose(arr,n);
        displayArray(arr,n);
    }
}
```

Rotate By 90 Degree (15 July)

Problem

Submissions

Leaderboard

Discussions

1. You are given a number n , representing the number of rows and number of columns.
 2. You are given $n*n$ numbers, representing elements of 2d array a .
 3. You are required to rotate the matrix by 90 degree clockwise and then display the contents using display function.
- Note - you are required to do it in-place i.e. no extra space should be used to achieve it.*

Sample Input 0

```
4 → n
11
12
13
14
21
22
23
24
31
32
33
34
41
42
43
44
```

Sample Output 0

```
41 31 21 11
42 32 22 12
43 33 23 13
44 34 24 14
```

	0	1	2	3
0	11	12	13	14
1	21	22	23	24
2	31	32	33	34
3	41	42	43	44

② Transpose

	0	1	2	3
0	41	31	21	11
1	42	32	22	12
2	43	33	23	13
3	44	34	24	14

	0	1	2	3
0	41	31	21	11
1	12	22	32	42
2	13	23	33	43
3	14	24	34	44

while (left < right)

for (int i = 0; i < n; i++)

```
left = 0;
right = n-1;
while (left < right) {
    swap
    left++;
    right--;
}
```

while (left < right)

90° rotation clockwise

① Transpose

② swapping rows element.

arr[i][left] ↔ arr[i][right]

	0	1	2	3
0	41	31	21	11
1	12	22	32	42
2	13	23	33	43
3	14	24	34	44

arr[0][0] ↔ arr[0][3]

arr[i][left] ↔ arr[i][right]

arr[0][0] ↔ arr[0][3]

```

public static void transpose(int arr[][],int n){
    for(int i=1;i<n;i++){
        for(int j=0;j<i;j++){
            int temp = arr[i][j];
            arr[i][j] = arr[j][i];
            arr[j][i] = temp;
        }
    }
}

public static void swappingRowElements(int arr[][],int n){
    for(int i=0;i<n;i++){
        int left=0,right=n-1;
        while(left<right){
            int temp = arr[i][left];
            arr[i][left] = arr[i][right];
            arr[i][right] = temp;

            left++;
            right--;
        }
    }
}

public static void displayTheArray(int arr[][],int n){
    for(int i=0;i<n;i++){
        for(int j=0;j<n;j++){
            System.out.print(arr[i][j] + " ");
        }
        System.out.println();
    }
}

```

```

public static void main(String[] args) {
    /* Enter your code here. Read input from STDIN.
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int arr[][] = new int[n][n];

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

    transpose(arr,n);
    swappingRowElements(arr,n);
    displayTheArray(arr,n);
}

```

Saddle Price (15 July)

Problem

Submissions

Leaderboard

Discussions

1. You are given a square matrix of size 'n'. You are given n*n elements of the square matrix.
2. You are required to find the saddle price of the given matrix and print the saddle price.
3. The saddle price is defined as the least price in the row but the maximum price in the column of the matrix.

Saddle Price

↳ least in row (min*)

↳ maximum value in col*

	0	1	2	3
→ 0	51	12	13	14
→ 1	21	22	23	24
→ 2	31	32	33	34
→ 3	41	42	43	44

min in row / max in col

→ 0	11	12	13	14
→ 1	22	21	23	24
→ 2	32	31	33	34
→ 3	42	41	43	44

→ Iterate over every row

→ min in row
→ max in col

-1

	12	13
41	42	43

4 → n
11
12
13
14
21
22
23
24
31
32
33
34
41
42
43
44

Sample Output 0

41

H.W.

for a given,
there is only
2 saddle price
possible

```

public class Solution {
    public static void saddlePrice(int arr[][], int n){
        for(int i=0; i<n; i++){
            int min = arr[i][0];
            int col = 0;
            boolean flag = true;
            for(int j=1; j<n; j++){
                if(min > arr[i][j]){
                    min = arr[i][j];
                    col = j;
                }
            }
            for(int k=0; k<n; k++){
                if(min < arr[k][col]){
                    flag = false;
                    break;
                }
            }
            if(flag == true){
                System.out.println(min);
                return;
            }
        }
        System.out.println(-1);
    }
}

```

\downarrow
 0 1 2
 $\rightarrow 0$

11	12	13
22	21	23
42	42	41

 $\rightarrow 1$
 $\rightarrow 2$

~~flag = true~~ \rightarrow false

$i=0 < 3$ (T)
 $min = arr[0][0]$ (0)
 $col = 0$

$j=1 < 3$ (T) $11 > 12$ X
 $j=2 < 3$ (T) $11 > 13$ X
 $j=3 < 3$ (F)

~~min = 11~~
 $col = 0$

$k=0 < 3$ (T)

$11 < 11$ (F)

$k=1 < 3$ (T)

$11 < arr[1][0]$

~~11 < 22~~
~~X~~

n

$5 \min$

$5 \min$

```
public class Solution {
    public static void saddlePrice(int arr[][],int n){
        for(int i=0;i<n;i++){
            int min = arr[i][0];
            int col = 0;
            boolean flag = true;
            for(int j=1;j<n;j++){
                if(min>arr[i][j]){
                    min = arr[i][j];
                    col = j;
                }
            }

            for(int k =0;k<n;k++){
                if(min <arr[k][col]){
                    flag = false;
                    break;
                }
            }

            if(flag == true){
                System.out.println(min);
                return ;
            }
        }
        System.out.println(-1);
    }
}
```

→

11	12	13
21	22	23
31	32	33

$m = 0$

~~sum = 0~~ 11
sum = 0

$j = 0$