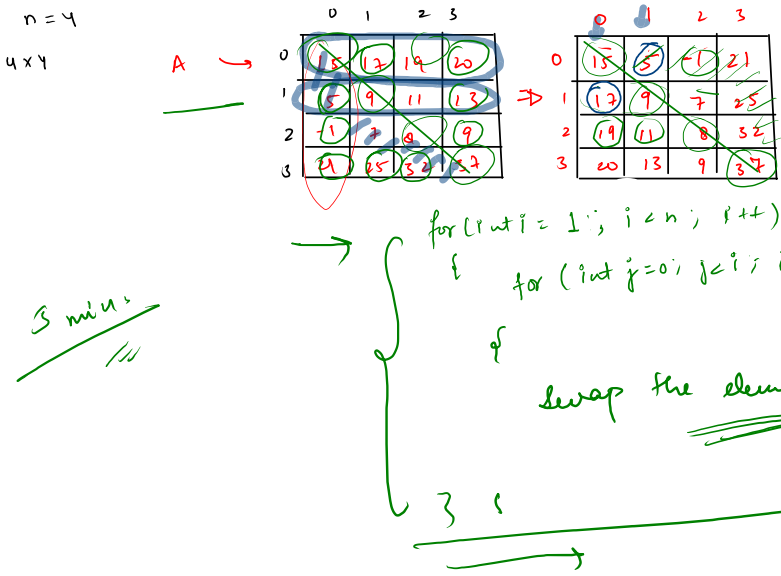
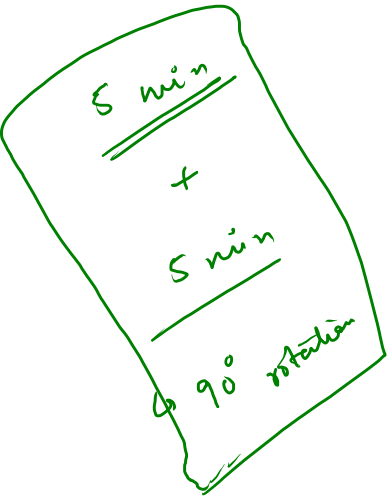
2 - D

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.






```java
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 STDO
        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.

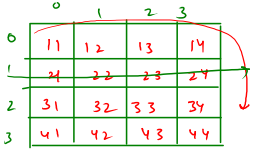
- Note - you are required to do it in-place i.e. no extra space should be used to achieve it .*

Sample Input 0

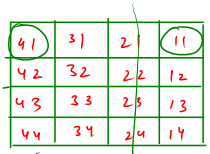
```
4
11
12
13
14
21
22
23
24
31
32
33
34
41
42
43
44
```

Sample Output 0

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



(a) → Transpose



while(left < right)

90° rotation clockwise

(a) Transpose

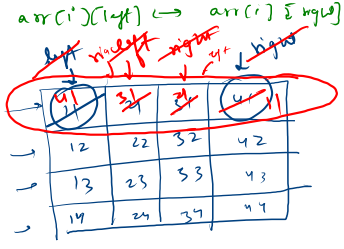
(b) swapping rows element.

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

while (left < right)

arr[i][left] ⟷ arr[i][right]

arr[0][0] ⟷ arr[0][3]



arr[i][left] ⟷ arr[i][right]

arr[0][0] ⟷ arr[0][3]

```java
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();
    }
}
```

```java
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)

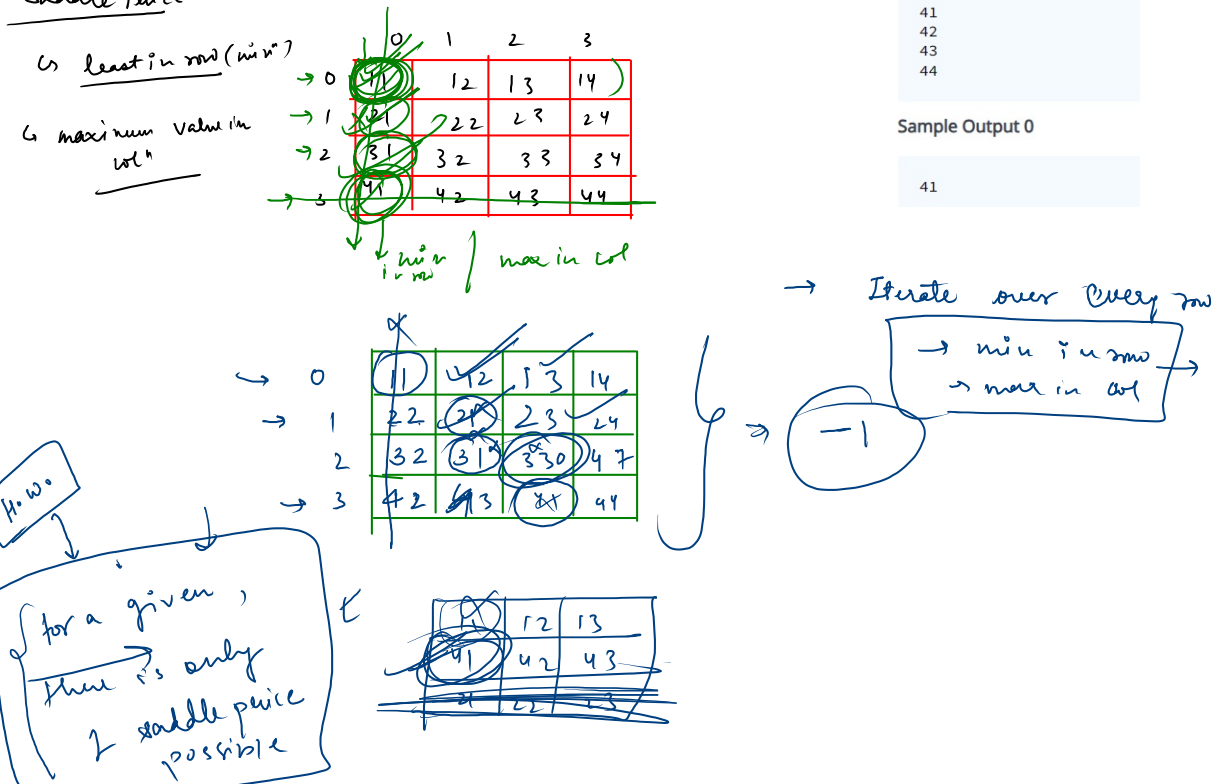
Problem | Submissions | Leaderboard | Discussions

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.

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

Sample Output 0

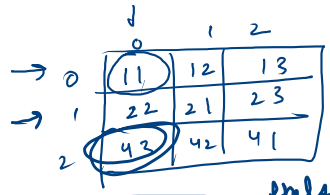
```
41
```



```java
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);
    }
```



flag = true → false

i=0 < 3 (T)

min = arr[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

```java
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);
    }
```