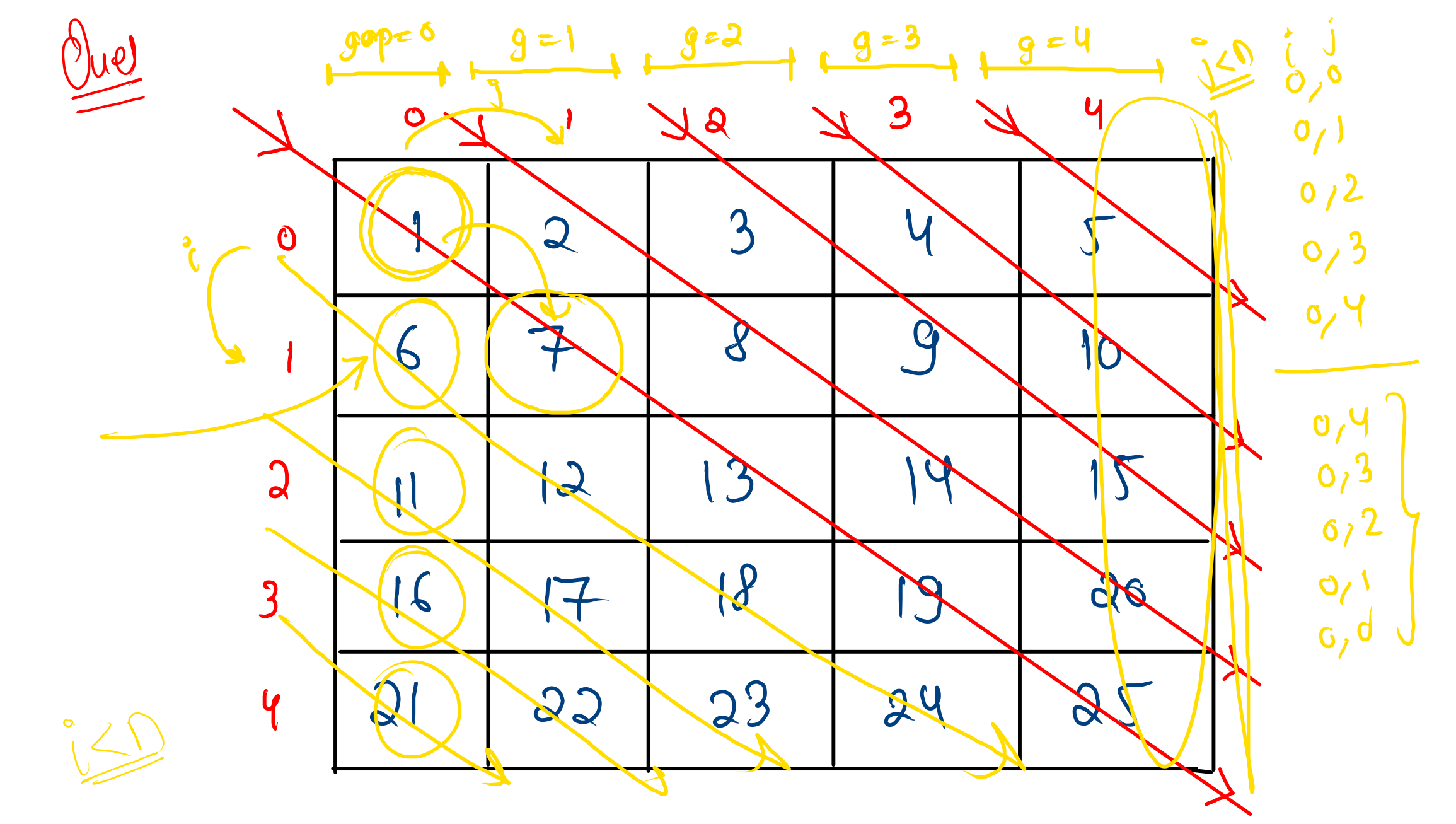

Que
gap=0
g=1
g=2
g=3
g=4
i<0
i j
0,0
0,1
0,2
0,3
0,4
0,4
0,3
0,2
0,1
0,0
0 1 2 3 4
0 1 2 3 4
1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16 17 18 19 20
21 22 23 24 25
i
i<0

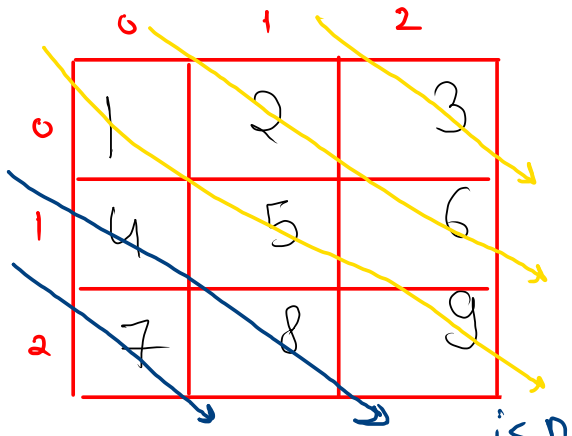
```java
public static void main(String[] args) {
    int[][] arr = {
        {1, 2, 3, 4},
        {1, 5, 3, 5},
        {9, 8, 3, 0},
        {2, 4, 3, 9}
    };
    // 1 5 3 9 2 3 0 3 5 4
    int n = arr.length;
    for (int gap = 0; gap < n; gap++) {
        for ( int i = 0, j = gap; j < n ; i++, j++) {
            System.out.print(arr[i][j] + " ");
        }
    }
}
```

i = 0
j = gap

```java
public static void main(String[] args) {
    int[][] arr = {
        {1, 2, 3, 4},
        {1, 5, 3, 5},
        {9, 8, 3, 0},
        {2, 4, 3, 9}
    };
    // 4 3 5 2 3 0 1 5 3 9 1 8 3 9 4 2
    int n = arr.length;
    for (int gap = n - 1; gap >= 0; gap--) {
        for ( int i = 0, j = gap; j < n ; i++, j++) {
            System.out.print(arr[i][j] + " ");
        }
    }

    for (int gap = 1; gap < n; gap++) {
        for ( int i = gap, j = 0; i < n ; i++, j++) {
            System.out.print(arr[i][j] + " ");
        }
    }
}
```

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



gap = 2, i=0, j=2 (2<3)

gap = 1, i=0, j=1 (1<3)
i=1, j=2 (2<3)

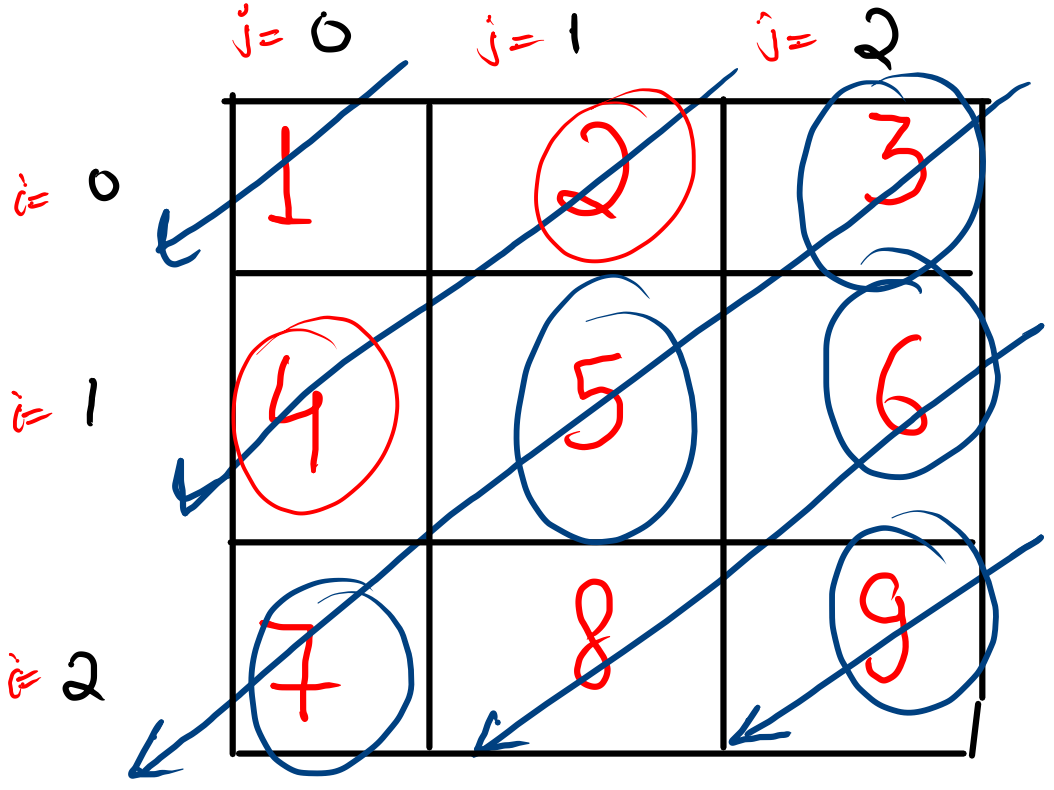
gap = 0, i=0, j=0 (0<3)
i=1, j=1 (1<3)
i=2, j=2 (2<3)

i<n

gap = 1, i=1, j=0 (1<3)
i=2, j=1 (2<3)

gap = 2, i=2, j=0 (2<3)

Print the matrix left-diagonal wise



pattern :- 1, 2, 4, 3, 5, 7, 6, 8, 9

0,0
0,1
0,2

i j
1, 2
2, 2

start { i = 0 , i = 1 → 2
j = 0 → 2 , j = 2

when to stop { j >= 0 , i < n

how to move { i++ , i++
j-- , j--

Code

```java
public static void solve(int[][] arr, int n) {

    for (int gap = 0; gap < n; gap++) {
        for ( int i = 0, j = gap; j >= 0; i++, j--) {
            System.out.print(arr[i][j] + " ");
        }
    }

    for (int gap = 1; gap < n; gap++) {
        for ( int i = gap, j = n - 1; i < n; i++, j--) {
            System.out.print(arr[i][j] + " ");
        }
    }

}
```

array

$(M) =$

1	2	3
4	5	6
7	8	9

$(M)^T =$

1	4	7
2	5	8
3	6	9

transpose of array

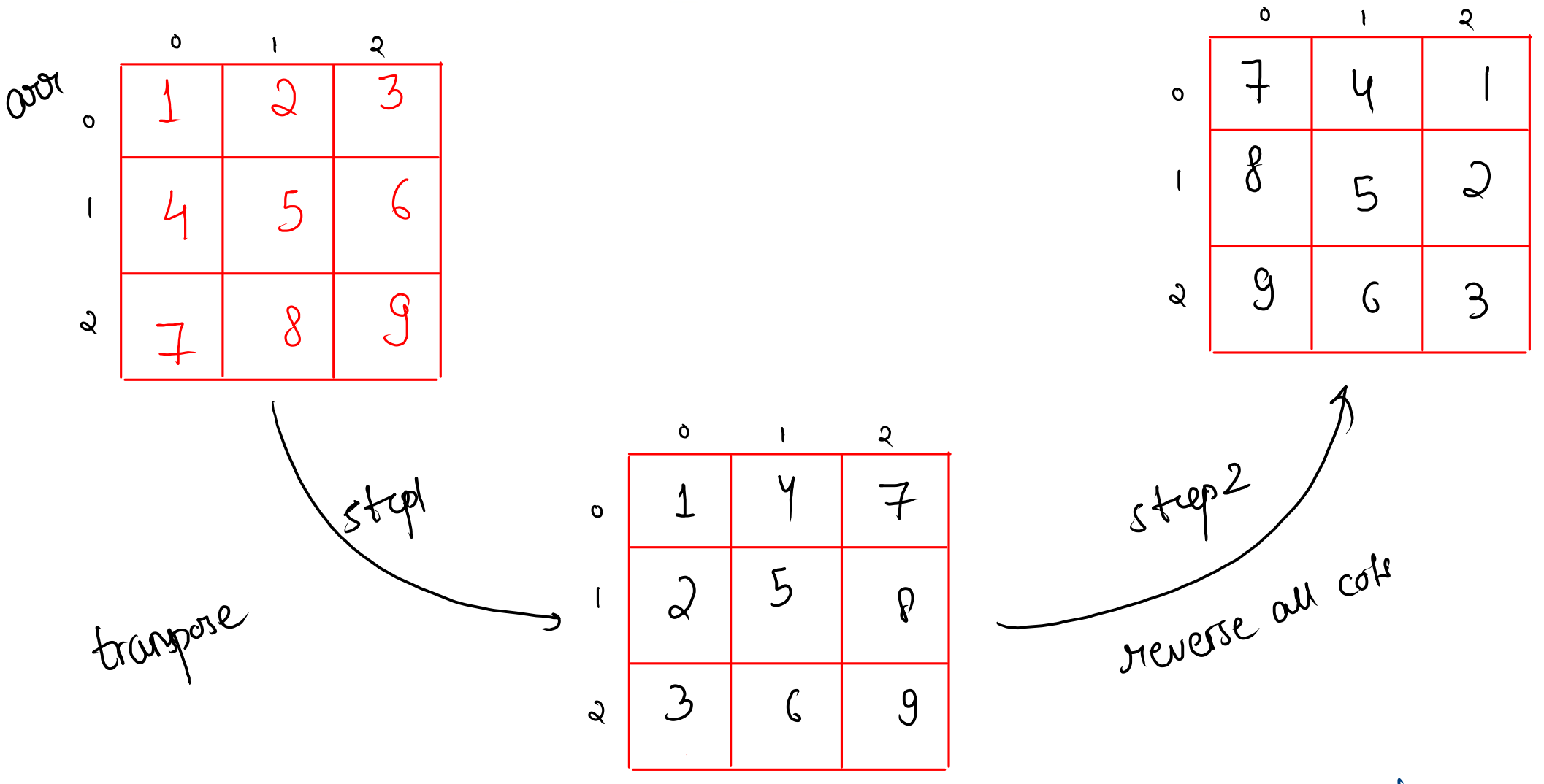
all rows will be changed to col and all cols will be changed to rows

code

```java
public static void transpose(int[][] arr) {
    int n = arr.length;
    for (int i = 0; i < n; i++) {
        for (int j = i; j < n; j++) {
            int temp = arr[i][j];
            arr[i][j] = arr[j][i];
            arr[j][i] = temp;
        }
    }

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

Rotate The Matrix by 90 Degree



consider each row as 1d array and reverse it

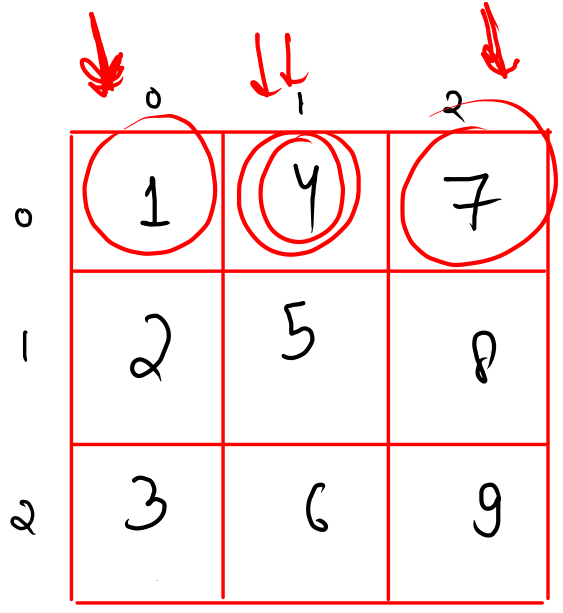
```java
public static void solve(int[][] arr) {
    // step 1
    transpose(arr);
    // step 2
    reverseCols(arr);

    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();
    }
}
```

print

```java
public static void transpose(int[][] arr) {
    int n = arr.length;
    for (int i = 0; i < n; i++) {
        for (int j = i; j < n; j++) {
            int temp = arr[i][j];
            arr[i][j] = arr[j][i];
            arr[j][i] = temp;
        }
    }
}
```

```java
public static void reverseCols(int[][] arr) {
    int n = arr.length;
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n / 2; j++) {
            int temp = arr[i][j];
            arr[i][j] = arr[i][n - 1 - j];
            arr[i][n - 1 - j] = temp;
        }
    }
}
```



n=3

$T.C = O(N^2+N^2)$

$\cong O(N^2)$

3-1-1

i=0, j=0 1

arr[i][j], arr[i][n-1-j]

arr[0][0], arr[0][2]

arr[0][1], arr[0][1]

Rotate The Matrix by 180 Degree

```java
public static void solve(int[][] arr) {
    transpose(arr);
    reverseCols(arr);

    transpose(arr);
    reverseCols(arr);

    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();
    }
}
```

$T.C = 4 * N^2$

$T.C \cong O(N^2)$

$S.C = O(1)$