3. Rotate by 90°.

✓

$$\begin{array}{ccc} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{array} \rightarrow \begin{array}{ccc} 7 & 4 & 1 \\ 8 & 5 & 2 \\ 9 & 6 & 3 \end{array}$$

A 1. Transpose(A)

$$\begin{array}{ccc} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{array} \xrightarrow[\text{all rows}]{\text{2. reversed}} \begin{array}{ccc} 7 & 4 & 1 \\ 8 & 5 & 2 \\ 9 & 6 & 3 \end{array}$$

2. 1D to 2D.

$n = 9$

$$\begin{array}{ccccccccc} 5 & 7 & 2 & 4 & 1 & 9 & 6 & 3 & 4 \\ 0 & 1 & 2 & 3 & 4 & 5 & 6 & \boxed{7} & 8 \end{array}$$

p	q
3	3

$$\begin{array}{c|c|c|c} & 0 & 1 & 2 \\ \hline 0 & 5 & 7 & 2 \\ \hline 1 & 4 & 1 & 9 \\ \hline 2 & 6 & \boxed{3} & 4 \end{array}$$

$x = i / q = 7/3 = 2$

$y = i \% q = 7 \% 3 = 1$

3. Spiral matrix.

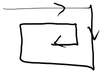
$$\begin{array}{c|cccc} & \text{minc } 0 & 1 & 2 & \text{maxc } 3 \\ \hline \text{minr } 0 & 1 & 2 & 3 & 4 \\ 1 & 5 & 6 & 7 & 8 \\ 2 & 9 & 10 & 11 & 12 \\ \text{maxr } 3 & 13 & 14 & 15 & 16 \end{array}$$

// 4x4

$t = 16$.

$c = 0$

$c < t$



$c < t$.

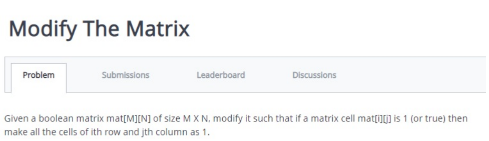
A — [minr] [$0 \le j \le 3$ { minc → maxc }]

B — [$1 \le i \le 3$ { minr to maxr }] [maxc]

C — [maxr] [$2 \le j \le 0$ { maxc → minc }]

D — [$2 \le i \le 1$ { maxr → min r }] [minc]

A minr → maxr [minc].



Modify The Matrix

Given a boolean matrix mat[M][N] of size M X N, modify it such that if a matrix cell mat[i][j] is 1 (or true) then make all the cells of ith row and jth column as 1.

Sample Input 0

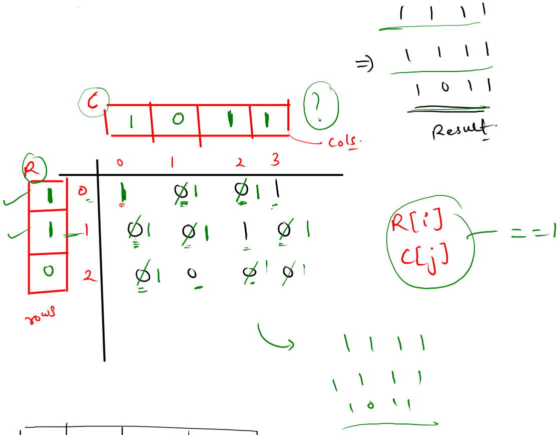
```
3
4
1 0 0 1
0 0 1 0
0 0 0 0
```

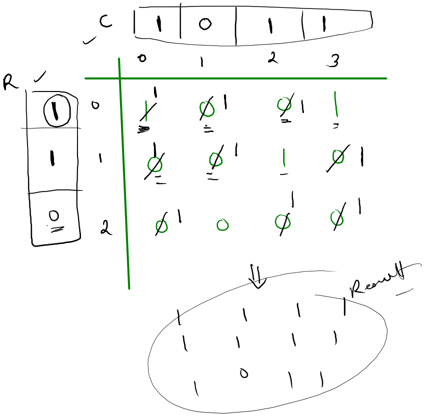
Sample Output 0

```
1 1 1 1
1 1 1 1
1 0 1 1
```

1. Check.
2. Update.

2 → 1D.



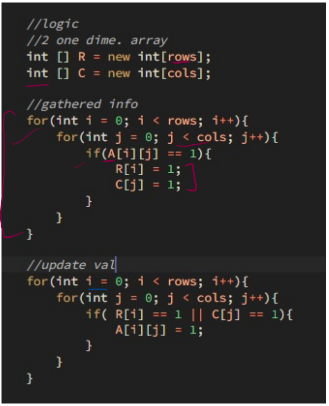


2nd.

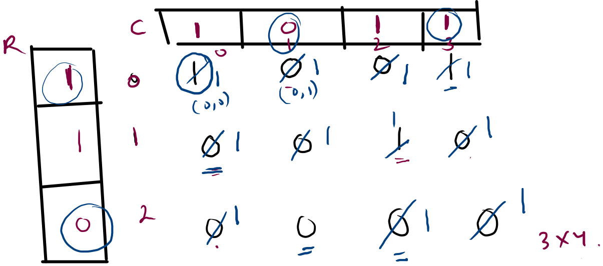
```
if (R[i]==1 ||
        C[j]==1 )
{
    A[i][j]=1;
}
```

1st

```
{
if ( A[i][j]==1)
{  R[i]=1;
   C[j]=1;
}
}
```



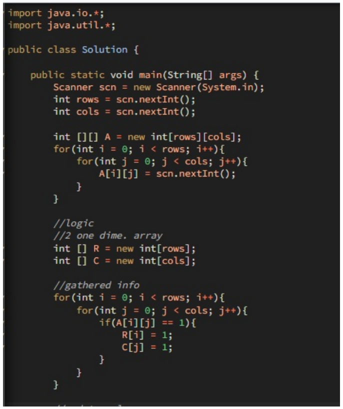
```
//logic
//2 one dime. array
int [] R = new int[rows];
int [] C = new int[cols];

//gathered info
for(int i = 0; i < rows; i++){
    for(int j = 0; j < cols; j++){
        if(A[i][j] == 1){
            R[i] = 1;
            C[j] = 1;
        }
    }
}

//update val
for(int i = 0; i < rows; i++){
    for(int j = 0; j < cols; j++){
        if( R[i] == 1 || C[j] == 1){
            A[i][j] = 1;
        }
    }
}
```



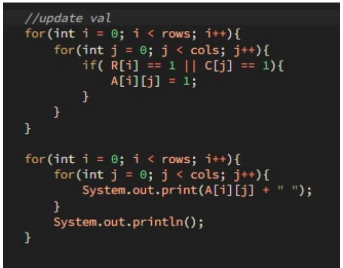
A(0)(3)==1

R(0)==1   C(0)==1

R(0)==1 or C(0)==1



```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int rows = scn.nextInt();
        int cols = scn.nextInt();

        int [][] A = new int[rows][cols];
        for(int i = 0; i < rows; i++){
            for(int j = 0; j < cols; j++){
                A[i][j] = scn.nextInt();
            }
        }

        //logic
        //2 one dime. array
        int [] R = new int[rows];
        int [] C = new int[cols];

        //gathered info
        for(int i = 0; i < rows; i++){
            for(int j = 0; j < cols; j++){
                if(A[i][j] == 1){
                    R[i] = 1;
                    C[j] = 1;
                }
            }
        }
```



```
//update val
for(int i = 0; i < rows; i++){
    for(int j = 0; j < cols; j++){
        if( R[i] == 1 || C[j] == 1){
            A[i][j] = 1;
        }
    }
}

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

Shift Matrix Row-Wise

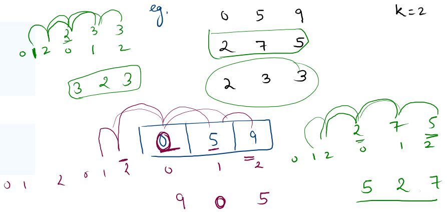
Given a n*n matrix and an integer k. Shift the matrix elements row-wise by k. Print the final matrix such that all elements of the row in one line.

Sample Input 0

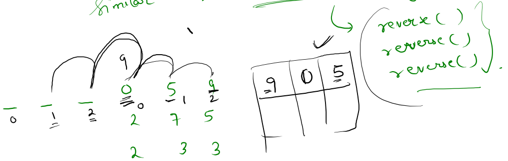
```
3
0 5 9
2 7 5
2 3 3
2
```

Sample Output 0

```
9 0 5
5 2 7
3 2 3
```



Similar → Rotate 1D Array.



	0	1	2
0	0	5	9
1	2	7	5
2	2	3	3

k=2

```
rev( 0, k-1)
rev( k, n-1)
reve( 0, n-1)
```

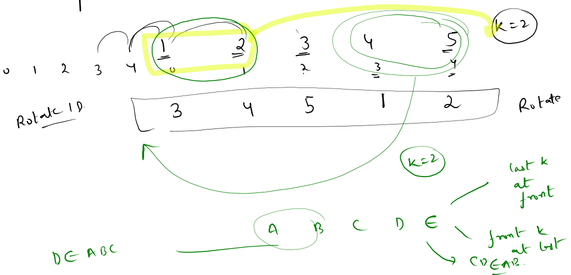
0 5 9

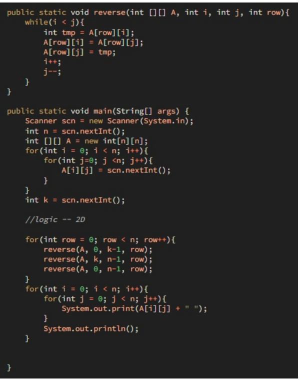
0 1 2

5 0 9 ⇒ 9 0 5

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

k=2





```
public static void reverse(int [][] A, int i, int j, int row){
    while(i < j){
        int tmp = A[row][i];
        A[row][i] = A[row][j];
        A[row][j] = tmp;
        i++;
        j--;
    }
}

public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int [][] A = new int[n][n];
    for(int i = 0; i < n; i++){
        for(int j=0; j <n; j++){
            A[i][j] = scn.nextInt();
        }
    }
    int k = scn.nextInt();

    //logic -- 2D

    for(int row = 0; row < n; row++){
        reverse(A, 0, k-1, row);
        reverse(A, k, n-1, row);
        reverse(A, 0, n-1, row);
    }
    for(int i = 0; i < n; i++){
        for(int j = 0; j < n; j++){
            System.out.print(A[i][j] + " ");
        }
        System.out.println();
    }


}
```

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

K=2

$$\begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & 0 & 5 & \boxed{9} \\ 1 & 2 & 7 & 5 \\ 2 & 2 & 3 & 3 \end{array} \Rightarrow \begin{array}{ccc} 9 & 0 & 5 \\ 5 & 2 & 7 \\ 3 & 2 & 3 \end{array}$$

r(0,k-1)

K=2

1	2	3	4
0	1	2	3

3 4 1 2

0,1

K, n-1

2,2

0,n-1

K=2

0 1 2 3 4

1	2	3	4	5
0	1	2	3	4

rotate 1D

3 4 5 1 2

2 1 5 4 3

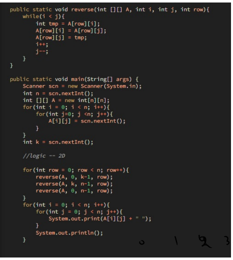
K=2

3	4	5
2	3	4

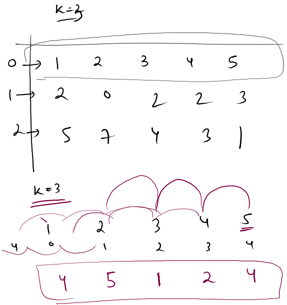
K

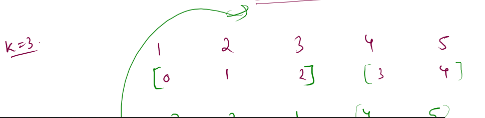
0, n-1

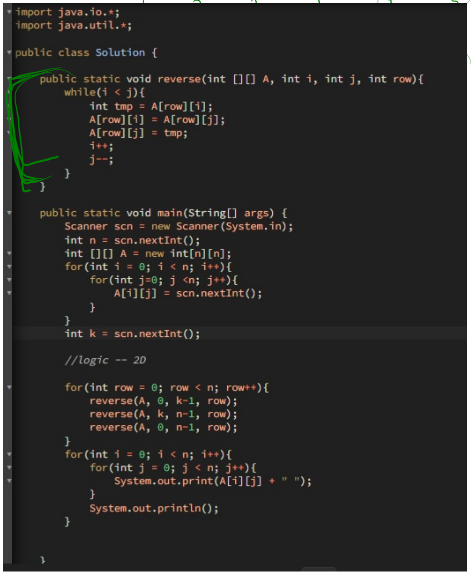
3 4 5 1 2



```java
public static void reverse(int [][] A, int i, int j, int row){
    while(i < j){
        int tmp = A[row][i];
        A[row][i] = A[row][j];
        A[row][j] = tmp;
        i++;
        j--;
    }
}

public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int [][] A = new int[n][n];
    for(int i = 0; i < n; i++){
        for(int j=0; j <n; j++){
            A[i][j] = scn.nextInt();
        }
    }
    int k = scn.nextInt();

    //logic -- 2D

    for(int row = 0; row < n; row++){
        reverse(A, 0, k-1, row);
        reverse(A, k, n-1, row);
        reverse(A, 0, n-1, row);
    }
    for(int i = 0; i < n; i++){
        for(int j = 0; j < n; j++){
            System.out.print(A[i][j] + " ");
        }
        System.out.println();
    }
```







```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void reverse(int [][] A, int i, int j, int row){
        while(i < j){
            int tmp = A[row][i];
            A[row][i] = A[row][j];
            A[row][j] = tmp;
            i++;
            j--;
        }
    }

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int n = scn.nextInt();
        int [][] A = new int[n][n];
        for(int i = 0; i < n; i++){
            for(int j=0; j <n; j++){
                A[i][j] = scn.nextInt();
            }
        }
        int k = scn.nextInt();

        //logic -- 2D

        for(int row = 0; row < n; row++){
            reverse(A, 0, k-1, row);
            reverse(A, k, n-1, row);
            reverse(A, 0, n-1, row);
        }
        for(int i = 0; i < n; i++){
            for(int j = 0; j < n; j++){
                System.out.print(A[i][j] + " ");
            }
            System.out.println();
        }


    }
```

for (row = 0 ____ n)

{

re.

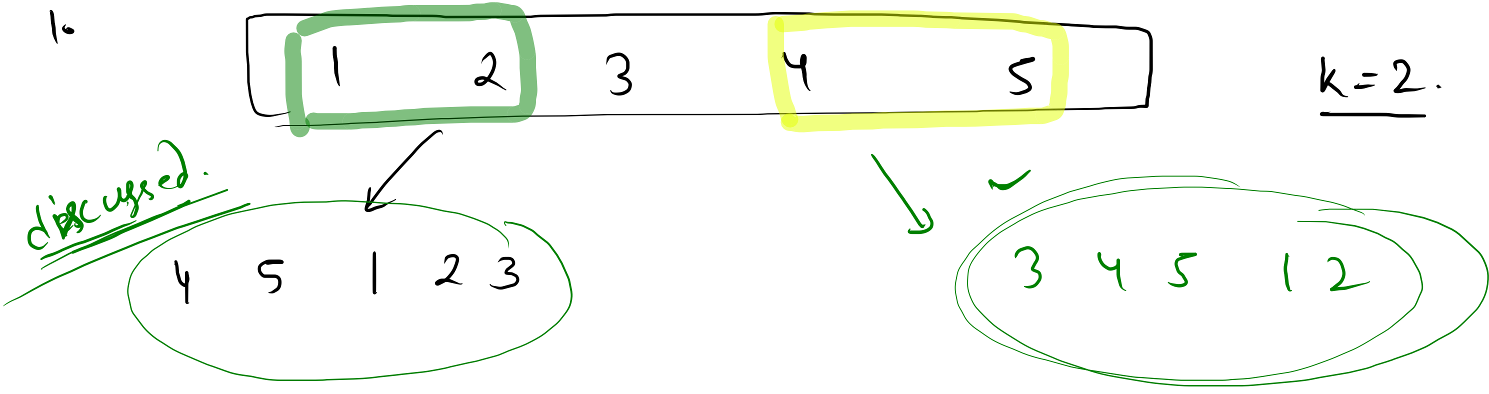
reverse()

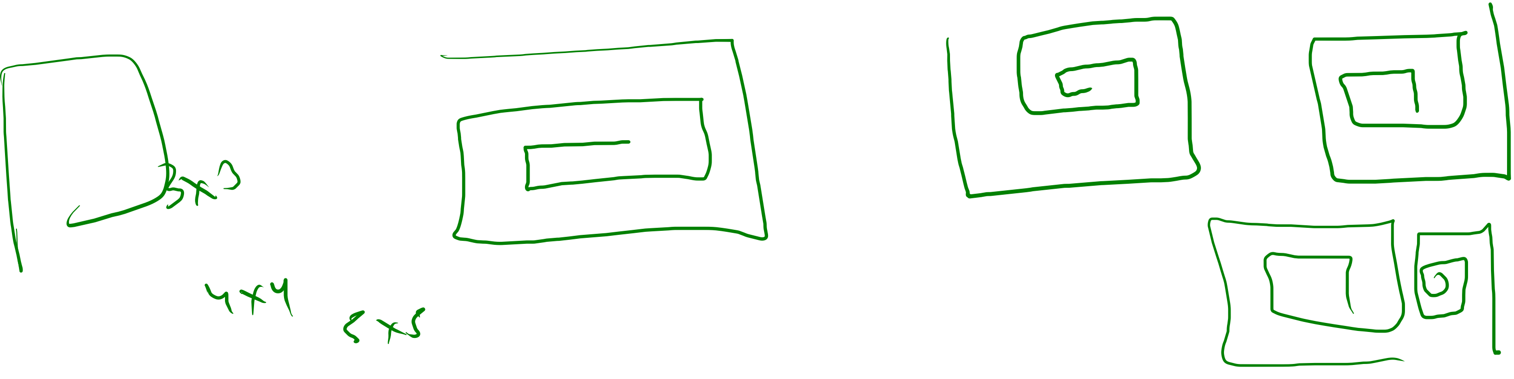
reverse()

}

1.

k = 2.





for each.