**Matrix Layer Rotation**

You are given a 2D matrix, a, of dimension MxN and a positive integer R. You have to rotate the matrix R times and print the resultant matrix. Rotation should be in anti-clockwise direction.

Rotation of a 4x5 matrix is represented by the following figure. Note that in one rotation, you have to shift elements by one step only (refer sample tests for more clarity).

It is guaranteed that the minimum of *M* and *N* will be even.

**Input Format**   
First line contains three space separated integers, *M*, *N* and *R*, where *M* is the number of rows, *N* is number of columns in matrix, and *R* is the number of times the matrix has to be rotated.   
Then *M* lines follow, where each line contains *N* space separated positive integers. These *M* lines represent the matrix.

**Output Format**   
Print the rotated matrix.

**Constraints**   
2 <= *M*, *N* <= 300   
1 <= *R* <= 109   
min(*M, N*) % 2 == 0   
1 <= *aij* <= 108, where *i* ∈ *[1..M]* & *j* ∈ *[1..N]*

**Sample Input #00**

4 4 1

1 2 3 4

5 6 7 8

9 10 11 12

13 14 15 16

**Sample Output #00**

2 3 4 8

1 7 11 12

5 6 10 16

9 13 14 15

**Sample Input #01**

4 4 2

1 2 3 4

5 6 7 8

9 10 11 12

13 14 15 16

**Sample Output #01**

3 4 8 12

2 11 10 16

1 7 6 15

5 9 13 14

**Explanation**   
Sample Case #00: Here is an illustration of what happens when the matrix is rotated once.

1 2 3 4 2 3 4 8

5 6 7 8 1 7 11 12

9 10 11 12 -> 5 6 10 16

13 14 15 16 9 13 14 15

Sample Case #01: Here is what happens when to the matrix after two rotations.

1 2 3 4 2 3 4 8 3 4 8 12

5 6 7 8 1 7 11 12 2 11 10 16

9 10 11 12 -> 5 6 10 16 -> 1 7 6 15

13 14 15 16 9 13 14 15 5 9 13 14

Solution :

import java.io.\*;

import java.util.\*;

public class Solution {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int m = sc.nextInt();

int n = sc.nextInt();

int r = sc.nextInt();

int a[][] = new int[m][n];

int tmp1=0,tmp2=0,p,q,e;

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

{

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

{

a[i][j] = sc.nextInt();

}

}

int z = ((m-1)\*2)+((n-1)\*2);

int x = r%z;

p = m;q = n;e=0;

int inner\_loops = Math.min(m, n) / 2;

//System.out.println(inner\_loops);

for(int y=0;y<inner\_loops;y++) //move to inner loop

{

for(int R=0;R<x;R++)

{

tmp1 = a[p-1][q-1];

for(int k=p-1;k>=y+1;k--)

{

tmp2 = tmp1;

tmp1 = a[k-1][q-1];

a[k-1][q-1] = tmp2;

}

for(int k=q-1;k>=y+1;k--)

{

tmp2 = tmp1;

tmp1 = a[y][k-1];

a[y][k-1] = tmp2;

}

for(int k=y;k<=p-2;k++)

{

tmp2 = tmp1;

tmp1 = a[k+1][y];

a[k+1][y] = tmp2;

}

for(int k=y;k<=q-2;k++)

{

tmp2 = tmp1;

tmp1 = a[p-1][k+1];

a[p-1][k+1] = tmp2;

}

}

e++;

p = p-1;

q = q-1;

if((p-e)==1 || (q-e)==1)

{ x=0; }

else

{ z = ((p-1-e)\*2)+((q-1-e)\*2);

//System.out.println("\t"+x+"\t"+z);

if(z!=0){ x = r%z; }

else{ x=0; }

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Print matrix \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

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

{

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

{

System.out.print(a[i][j]+" ");

}

System.out.println();

}

}// end of main

}