На вход поступает Матрица размером MxN:

1 2 3 4 5 6

2 3 4 5 6 7

3 4 5 6 7 8

4 5 6 7 8 9

Ты должен научиться вращать Матрицу относительно её центра по часовой стрелке.

Например, вращение на один шаг:

2 1 2 3 4 5

3 4 3 4 5 6

4 5 6 7 6 7

5 6 7 8 9 8

Функция

void MatrixTurn(string Matrix[], int M, int N, int T)

получает на вход (по ссылке) массив строк (M строк, каждая длиной N; M >= 2, N >= 2), и вращает его относительно центра по часовой стрелке на T шагов (T >= 1), как описано выше.   
То есть результат поворота (повёрнутая матрица) оказывается в исходном массиве Matrix, переданном в функцию по ссылке как аргумент.

Минимальное значение из чисел M,N обязательно чётно.

Пример вызова:

MatrixTurn(["123456", "234567", "345678", "456789"], 4,6, 3)

[как постить решение](https://skillsmart.ru/algo/lvl1/how.html)

решение вроде бы напрашивается простое

вычисляем координаты внешнего кольца, которое будем вращать

в данном случае 0,0 -> 0,6 -> 6,6 -> 6,0

если M или N больше двух – для следующего кольца будем перемещаться на 1,1 и так далее

дальше свигаем верхний ряд вправо (на T шагов), левый столбец вверх и так далее по кругу

ниже вариант с вращением нескольких колец (в примере задачи их два)

+ пошаговое вращение (T)

То есть сначала поворачиваем «внешнее» кольцо по часовой, потом «внутреннее» на 1 шаг.

Повторяем T раз

public class MatrixRotation {  
  
 public static void MatrixTurn(String Matrix[], int M, int N, int T) {  
  
 int ringAmount;  
 char[][] testArr = new char[M][N];  
 for (int i = 0; i < M; i++){  
 for (int j = 0; j < N; j++){  
 testArr[i][j] = Matrix[i].charAt(j);  
 }  
 }  
  
 //calculating amount of "rings", we will rotate them separately  
 if (M > N) {  
 ringAmount = N/2;  
 }  
 else ringAmount = M/2;  
 System.*out*.println("number of rings: " + ringAmount);  
  
 // ---- main loop (number of rotations) ----  
 for (int s = 0; s < T; s++){  
 System.*out*.println("rotation step " + s);  
 // ---- inner loop (rotating rings starting from outside one) ----  
 for (int r = 0; r < ringAmount; r++){  
 System.*out*.println("rotating ring " + r);  
  
 *rotationClockwise*(testArr, *currentRingBorders*(testArr, r));  
  
 }  
 System.*out*.println("===== matrix after rotaion " + s + " =========");  
 *printArray*(testArr);  
 System.*out*.println("----------------------");  
 }  
  
 }  
  
 //ring coordinates(top left corner, height, length  
 public static int[] currentRingBorders (char[][] inputArr1, int ring){  
 int[] borders = new int[3];  
 int s = ring;  
 System.*out*.println("coordinates: offset, height, length");  
  
 borders[0] = s;  
 borders[1] = inputArr1.length - 1 - s;  
 borders[2] = inputArr1[0].length - 1 - s;  
  
 for (int i = 0; i < 3; i++){  
 System.*out*.print(borders[i] + " ");  
 }  
  
 System.*out*.println();  
 return borders;  
 }  
  
 public static void printArray (char[][] arr2print){  
 for (int j = 0; j < arr2print.length; j++) {  
 for (int i = 0; i < arr2print[0].length; i++) {  
 System.*out*.print(arr2print[j][i]);  
 }  
 System.*out*.println();  
 }  
 System.*out*.println("----------");  
 }  
  
 public static void rotationClockwise (char[][] arr2rotate, int[] borders){  
 char temp;  
 temp = arr2rotate[borders[0]][borders[2]];  
  
 //top row moved right  
 for (int i = borders[2] - 1; i >= borders[0]; i--){  
 arr2rotate[borders[0]][i + 1] = arr2rotate[borders[0]][i];  
 }  
   
 //left column moved up  
 for (int i = 0; i < borders[1]; i++){  
 arr2rotate[borders[0] + i][borders[0]] = arr2rotate[borders[0] + 1 + i][borders[0]];  
 }  
  
 //bottom row moved left  
 for (int i = borders[0]; i < borders[2]; i++){  
 arr2rotate[borders[1]][i] = arr2rotate[borders[1]][i + 1];  
 }  
   
 //right column moved down + insert temp  
 for (int i = borders[1]; i > borders[0] + 1; i--){  
 arr2rotate[i][borders[2]] = arr2rotate[i - 1][borders[2]];  
 }  
 arr2rotate[borders[0] + 1][borders[2]] = temp;  
   
 }  
   
  
 public static void main(String[] args) {  
 String[] test1 = {"123456", "234567", "345678", "456789"};  
 int testM = 4;  
 int testN = 6;  
 int testT = 3;  
  
 char[][] testArr = new char[testM][testN];  
 for (int i = 0; i < testM; i++){  
 for (int j = 0; j < testN; j++){  
 testArr[i][j] = test1[i].charAt(j);  
 }  
 }  
  
 *printArray*(testArr);  
 // MatrixTurn(test1, testM, testN, testT);  
 // currentRingBorders(testArr, 1);  
 //rotationClockwise(testArr, currentRingBorders(testArr, 2));  
 // printArray(testArr);  
  
 *MatrixTurn*(test1, testM,testN,testT);  
  
  
  
 }  
}

в конце основного метода добавляем вывод в виде String[]

for (int i = 0; i < Matrix.length; i++){  
 Matrix[i] = new String(testArr[i]);  
}  
  
System.*out*.println(Arrays.*toString*(Matrix));