```
package com.itbulls.learnit.javacore.methods.hw;

import java.util.Scanner;

public class MatrixRotation {

    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);

        System.out.print("Please, enter matrix size: ");
        int size = in.nextInt();
        double[][] matrix = generateMatrix(size);

        System.out.println("How you want to rotate matrix:" + System.lineSeparator() +
                "\t1 - 90" + System.lineSeparator() +
                "\t2 - 180" + System.lineSeparator() +
                "\t3 - 270");
        int mode = in.nextInt();

        System.out.println(System.lineSeparator() + "Base matrix:" +
System.lineSeparator());
        printMatrixToConsole(matrix);
        System.out.println();

        if (rotateMatrix(matrix, mode)) {
            printMatrixToConsole(matrix);
        }
    }

    private static double[][] generateMatrix(int size) {
        double[][] matrix = new double[size][size];
        for (int i = 0; i < matrix.length; i++) {
            for (int j = 0; j < matrix.length; j++) {
                matrix[i][j] = Double.valueOf(Integer.toString(i) + "."
                        + Integer.toString(j));
            }
        }
        return matrix;
    }

    private static boolean rotateMatrix(double[][] matrix, int mode) {
        switch (mode) {
            case 1:
                System.out.println("90 degrees rotated:" + System.lineSeparator());
                rotate90(matrix);
                break;
            case 2:
                System.out.println("180 degrees rotated:" +
System.lineSeparator());
                rotate180(matrix);
                break;
            case 3:
                System.out.println("270 degrees rotated:" +
System.lineSeparator());
                rotate270(matrix);
                break;
            default:
                System.out.println("You selected non-existing mode!");
                return false;
            }
        return true;
    }
```

```
private static void transposeMatrix(double[][] matrix) {
    double temp;
    for (int i = 0; i < matrix.length; i++) {
        for (int j = 0; j < i; j++) {
            temp = matrix[i][j];
            matrix[i][j] = matrix[j][i];
            matrix[j][i] = temp;
        }
    }
}

private static void verticalReflection(double[][] matrix) {
    double temp;
    for (int i = 0; i < matrix.length; i++) {
        for (int j = 0; j < matrix.length / 2; j++) {
            temp = matrix[i][j];
            matrix[i][j] = matrix[i][matrix.length - 1 - j];
            matrix[i][matrix.length - 1 - j] = temp;
        }
    }
}

private static void horizontalReflection(double[][] matrix) {
    double temp;
    for (int i = 0; i < matrix.length / 2; i++) {
        for (int j = 0; j < matrix.length; j++) {
            temp = matrix[i][j];
            matrix[i][j] = matrix[matrix.length - 1 - i][j];
            matrix[matrix.length - 1 - i][j] = temp;
        }
        }
    }

    public static void rotate90(double[][] matrix) {
        transposeMatrix(matrix);
        verticalReflection(matrix);
    }

    public static void rotate180(double[][] matrix) {
        verticalReflection(matrix);
        horizontalReflection(matrix);
    }

    public static void rotate270(double[][] matrix) {
        transposeMatrix(matrix);
        horizontalReflection(matrix);
    }

    private static void printMatrixToConsole(double[][] matrix) {
        for (int i = 0; i < matrix.length; i++) {
            for (int j = 0; j < matrix.length; j++) {
                System.out.print(matrix[i][j] + "\t");
            }
            System.out.println();
        }
    }



}
```