CS1020 Take-home Lab #1

Exercise #3: Matrix Transformation

(http://www.comp.nus.edu.sg/~cs1020/3_ca/labs.html)

Objective

To test students' understanding on two-dimensional array and manipulation of 2D array.

Task Statement

Given a square matrix, output the final state of the matrix after performing the given operations. There are 3 valid operations:

1. **Rotate r:** Rotate the matrix clockwise by *r* degree (*r* can be 90, 180, or 270).
2. **Reflect x:** Reflect the matrix across the x-axis.
3. **Reflect y:** Reflect the matrix across the y-axis.

Input

The first line of the input contains an integer **N** ($1 \leq \mathbf{N} \leq 100$). The subsequent **N** lines contain the values of the **N** x **N** elements. The following line is an integer **K** ($1 \leq \mathbf{K} \leq 100$), where **K** is the number of operations to be performed. The next line is the query with format "Rotate r", ($r \in \{90, 180, 270\}$), "Reflect x" or "Reflect y".

(The above symbols **N** and **K** are used to ease explanation. In your program, you should give more descriptive variable names and follow Java naming convention.)

Output

The output is the final state of the matrix.

Sample Input

```
3
1 2 3
4 5 6
7 8 9
3
Rotate 90
Reflect x
Reflect y
```

Sample Output

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

Explanation

1. *Initial matrix:*
   1 2 3
   4 5 6
   7 8 9
2. After 90° clockwise rotation:
   7 4 1
   8 5 2
   9 6 3
3. *After reflection across x axis:*
   9 6 3
   8 5 2
   7 4 1
4. *After reflection across y axis:*
   3 6 9
   2 5 8
   1 4 7

Skeleton Program

The following skeleton is given. Normally, you would have put the **Matrix** class (the server) and **Transform** class (the client) in separate files, but to reduce the number of files submitted, you are to put them in a single file **Transform.java**.

Note that in the **Matrix** class, only the constructors, **operate()** and **toString()** are public methods; the rest are private methods and hence are not accessible to client programs. The data attributes, constructors and **toString()** method are given and you are not to change them.

```
/*
 * CS1020 (AY2012/3 Sem2)
 * Lab #1 Ex3
 * Author    :
 * Matric no.:
 * Description of program:
 */
import java.util.*;

class Matrix {

   // Data attributes
   int size;
   int matrix[][];

   // Constructors
   // Default constructor creates a 1x1 matrix
   public Matrix() {
      this(1);
   }

   public Matrix(int size) {
      this.size = size;
      this.matrix = new int[size][size];
   }

   // Write a description for this method
   private void rotate(int degree) {

   }

   // Write a description for this method
   private void reflectX() {

   }

   // Write a description for this method
   private void reflectY() {

   }

   // To determine which operation to perform  based on the operation
   // and type parameters
   public void operate(String operation, String type) {

   }
```

```
    // String representation of matric
    public String toString() {
        String output = "";

        for (int i = 0; i < size; i++) {
            for (int j = 0; j < size; j++) {
                if (j > 0)
                    output += " ";
                output += matrix[i][j];
            }
            output += "\n";
        }
        return output;
    }
}

class Transform {

    public static void main(String[] args) {
        // Declare a Scanner object to read input


        // Declare the necessary variables
        Matrix result = ...


        // Read input and process them accordingly


        // Output the result
        // Stick to this statement to ensure that your output is
        // in the right format; the following call makes use of
        // the toString() method of Matrix class implicitly
        System.out.print(result);
    }
}
```