CS1020E | Lab 2 | Exercise 2

Matrix Transformation

Objective

The objective of this exercise is to practice basic OOP programming by defining and using simple classes.

Problem Description

Given a 2-dimensional square matrix, output the final state of the matrix after performing the given sequence of transformation operations. The followings are the valid operations:

1. **`Rotate`** *X*
   - Rotate the matrix by *X* degrees clockwise, and *X* can only be 90, 180, or 270.
2. **`Reflect x`**
   - Reflect the matrix about the *x*-axis.
3. **`Reflect y`**
   - Reflect the matrix about the *y*-axis.

In your program, you may want to read each input transformation operation into a `string` object, and you can compare the value in the `string` object to a string literal using the == operator, such as the followings:

```
string testStr;
cin >> testStr;
if (testStr == "Good") ...
```

For more info about the `string` class, see http://www.cplusplus.com/reference/string/string/.

Inputs

The first line of the input contains one integer *N*, where 1 <= *N* <= 100. The next *N* lines contain the *N* x *N* integers of the matrix.

The next line is an integer *K*, where 1 <= *K* <= 100, and it is the number of transformation operations to be performed.

Each of the subsequent *K* lines is the operation "**`Rotate`** *X*" (where is *X* is 90, 180 or 270), "**`Reflect x`**", or "**`Reflect y`**".

Outputs

The output is the final state of the matrix after the given sequence of transformation operations.

Sample Input

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

(*User inputs are shown in **bold red***.)

Sample Output

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

Explanation of Sample Output

1. Initial matrix:

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

2. After 90 degrees rotation:

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

3. After reflection about the *x*-axis:

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

4. After reflection about the *y*-axis:

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

Submission

You need to submit only your completed **`Matrix.cpp`**, **`Matrix.h`**, and **`Transformation.cpp`** to CodeCrunch (https://codecrunch.comp.nus.edu.sg/) before the specified deadline. We will take only your latest submission.

Late submissions will not be accepted. The submission system in CodeCrunch will automatically close at the deadline.