Joanna Klukowska
joannakl@cs.nyu.edu

Assignment 5

Due date: March 22, 11:55PM EST.

You may discuss any of the assignments with your classmates and tutors (or anyone else) but **all work for all assignments must be entirely your own**. Any sharing or copying of assignments will be considered cheating.

You should not use any features of Java that have not been covered in class. If you have doubt if you are allowed to use certain structures, just ask your instructor.

Problem 1 (60 points): Matrix Manipulations

Write a program that allows the user to manipulate matrices.

Program Description

A matrix is is a rectangular array of numbers arranged in rows and columns.

For example: $\begin{bmatrix} 1.5 & 17 & -16 & 12 \\ 2 & -5 & 18.5 & 13.1 \\ -3.75 & 13 & 12.3 & 0 \end{bmatrix}$, or $\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$, or $\begin{bmatrix} 5 & 17 \\ 14 & 3 \end{bmatrix}$.

Your program should prompt the user for the number of rows and the number of columns in the matrix and then generate a matrix of such dimensions with random integers in the range of -10 to 10.

Your program should display the matrix to the screen and then display a menu to the user with the following options

H horizontal flip - each row is reversed,

V vertical flip - each column is reversed,

T transpose - rows become columns (and vice versa),

R rowMax - find largest value in each row,

C columnSum - find sum of the values in each column,

Q quit

(The program should accept uppercase and lowercase letter as the selections from the menu. For example, both **H** and **h** should trigger the horizontal flip.)

If the user selects one of the first five options, the operation should be applied to the matrix of numbers and either the new matrix should be displayed or the results computed by the operation should be shown. Then the menu with the above choices should be shown again.

On each iteration of the program the modifications should be performed on the **original** matrix that the program generated randomly (not on the matrix that results from the most recent operation).

If the user selects the quit option, the program should terminate.

If the user selects an invalid option, the program should print an error message and the menu should be redisplayed.

Implementation Requirements

The program has to implement the following methods

- `public static void printMatrix (int [][] matrix )`

  This method takes as a parameter a 2D array representing a matrix and prints it to the console. The columns should be aligned in all the rows. The method may assume that no number has more than 3 digits (including the sign symbol). The numbers in columns should be aligned to the right hand side of each column. A sample printed output might look like this:

Joanna Klukowska
joannakl@cs.nyu.edu

```
10   6   -7    9
 0  -2    4  -10
10   1    2   -3
```

Hint: You will need to use `printf()` method to format the output.

- **`public static int [][] horizontalFlip (int [][] matrix )`** This method returns a new matrix whose rows are created by reversing each row of the matrix passed as a parameter. It does **not** modify the matrix that is passed to it. The above matrix would produce

```
  9  -7    6   10
-10   4   -2    0
 -3   2    1   10
```

- **`public static int [][] verticalFlip (int [][] matrix )`** This method returns a new matrix whose columns are created by reversing each column of the matrix passed as a parameter. It does **not** modify the matrix that is passed to it. The above matrix would produce

```
10   1    2   -3
 0  -2    4  -10
10   6   -7    9
```

- **`public static int [][] transpose (int [][] matrix )`** This method returns a new matrix whose columns are rows of the matrix passed as a parameter. It does **not** modify the matrix that is passed to it. The above matrix would produce

```
10   0   10
 6  -2    1
-7   4    2
 9 -10   -3
```

- **`public static int [] rowMax (int [][] matrix )`** This method returns an array of numbers such that the value at index **`i`** is the largest element of the **`i`**'th row in the matrix passed as a parameter. The above matrix would produce

```
10   4   10
```

- **`public static int [] columnSum (int [][] matrix )}`** This method returns an array of numbers such that the value at index i is the sum of all element of the **`i`**'th column in the matrix passed as a parameter. The above matrix would produce

```
20   5   -1  -4
```

The program has to provide verification of the user choices:

- The dimensions of the matrix have to be positive numbers between 1 and 5 for the number of rows and columns.
- The selection of the menu options should be verified. Your program should allow lowercase and uppercase letters. If the user enters an invalid choice, the program should print an error message and the menu should be redisplayed.

Call your file: **`Matrix.java`**.
Make sure to document your program using Javadoc style comments.

Problem 2 (40 points): Rectangle Class

Following the example of the **`Circle`** class in your textbook, design a class named **`Rectangle`** to represent a rectangle. The class should contain:

- Two **`double`** data fields named width and **`height`** that specify the width and height of the rectangle. The data fields should be private. The default values are **`0`** for both width and **`height`**.
- A no-arg constructor that creates a default rectangle.
- A constructor that creates a rectangle with the specified width and **`height`**.

Joanna Klukowska
joannakl@cs.nyu.edu

- Two setter methods that allow the user to modify the values of width and `height`.
- Two getter methods that allow the user to access the values of width and `height`.
- A method named `getArea()` that returns the area of this rectangle.
- A method named `getPerimeter()` that returns the perimeter of this rectangle.

To test the implementation of your class, use the following `TestRectangle` class:

```
1
2 public class TestRectangle {
3
4   public static void main(String[] args) {
5     // create a default Rectangle object
6     Rectangle rec1 = new Rectangle () ;
7     System.out.printf("rectangle 1:"
8         + "\n %-12s %.3f x %.3f"
9         + "\n %-12s %.3f"
10        + "\n %-12s %.3f\n\n",
11        "dimensions:", rec1.getWidth() , rec1.getHeight(),
12        "area:", rec1.getArea(), "perimeter:", rec1.getPerimeter());
13
14    // create a default Rectangle object
15    //and then set values of width and height
16    Rectangle rec2 = new Rectangle () ;
17    rec2.setWidth(3.33333333);
18    rec2.setHeight(5.1);
19    System.out.printf("rectangle 2:"
20        + "\n %-12s %.3f x %.3f"
21        + "\n %-12s %.3f"
22        + "\n %-12s %.3f\n\n",
23        "dimensions:", rec2.getWidth() , rec2.getHeight(),
24        "area:", rec2.getArea(), "perimeter:", rec2.getPerimeter());
25
26    // create a Rectangle object using 2-param constructor
27    Rectangle rec3 = new Rectangle (13.7, 2.3) ;
28    System.out.printf("rectangle 2:"
29        + "\n %-12s %.3f x %.3f"
30        + "\n %-12s %.3f"
31        + "\n %-12s %.3f\n\n",
32        "dimensions:", rec3.getWidth() , rec3.getHeight(),
33        "area:", rec3.getArea(), "perimeter:", rec3.getPerimeter());
34  }
35 }
36
```

Call your file: `Rectangle.java`.
(Submit both files: `Rectangle.java` and `TestRectangle.java`.)
Make sure to document your program using Javadoc style comments.

Grading

Does the program compile? If not, you will loose all the points for that problem.

Is the program properly documented using Javadoc style comments? (worth approximately 20% of each problem)

Proper documentation at this point in the course includes:

- preamble with the name of the author, date of creation and brief description of the program (the description should specify what the program does, not that it is a solution to problem 1 of homework 1);
- method comments - description of what the method does, description of each parameter, description of the return value

Joanna Klukowska
joannakl@cs.nyu.edu

- inline comments - comments inside the code describing steps needed to be taken to accomplish the goal of the program;
- appropriately chosen variable names, i.e., descriptive names (a good name for the variable that stores the bonus amount in the last problem is `bonus`, not `x`);
- appropriate formatting, indentation and use of white space to make the code readable.

Remember that the code is read by humans and it should be easy to read for people who were not involved in its development.

Is the program well developed? (worth approximately 40% of each problem) Make sure you create variables of appropriate types, use control statements (conditionals and loops) that are appropriate for the task, accomplish your task in a well designed and simple way (not a convoluted algorithm that happens to produce the correct output for some unknown reason). You should also design a friendly and informative user interface.

Is the program correct? (worth approximately 40% of each problem) Make sure that your program produces valid results that follow the specification of the problem every time it is run. At this point you can assume a "well behaved user" who enters the type of data that you request. If the program is not completely correct, you get credit proportional to how well it is developed and how close you got it to the completely correct code.

What and how to submit?

You should submit three source code files combined into a single **zip** file to NYU Classes. Do not submit all the files that Eclipse creates, just the source code files that have `.java` extensions. Name your classes as specified in the problems.

If you wish to use your (one and only) freebie for this project (one week extension, no questions asked), then complete the form at `http://goo.gl/forms/fpUJrF64b5` **before the due date for the assignment**. All freebies are due seven days after the original due date and should be submitted to NYU Classes.

Questions

Post any questions you have regarding this assignment to Piazza under the "homeworks" topic.