CSCI 2134 Lab 6: Exceptions and Assertions

Winter 2020

Due Date: Thursday, March 5, 2020 at 11:59pm, submitted via Git

Objective

In this lab, you will work on your own or with another student to improve the robustness of existing code using exceptions and assertions.

Preparation

1. Ensure that you have your Integrated Development Editor (IDE) installed.
2. Ensure that you are able to write and run a JUnit test case in the IDE. You will not have the time to both debug your IDE environment and complete the lab task within the lab time. The lab time should be able to concentrate on the lab task.
3. Clone the Lab 6 repo: https://git.cs.dal.ca/courses/2020-winter/csci-2134/lab6/????.git where ???? is your CSID.
4. Review the provided code listed in the Resources section below by reading the code of the class you will be fortifying (`Matrix.java`) and the unit tests (`MatrixTest.java`). These files are located in the `src` and `test` directories of the cloned project.
5. You may also wish to read the provided primer on matrix operations, found in the `docs` directory.

Resources

- Primer on matrices in the **`docs`** directory of the Lab 6 repository.
- Code base to be debugged is in the **`src`** directory of the Lab 6 repository.
- Test code is in the **`test`** directory of the Lab 6 repository.

Procedure

Set-up

1. Open the project you created in preparation for this lab

Part 1: Add Exceptions to `Matrix.java`

1. Review the code for `getElem()` and the constructors in Matrix.java as an example of what you are being asked to do.
2. For each of following methods review the updated specifications in `Matrix.java`. Instead of returning `null`, the method should be modified to throw an exception. Implement this change in the following methods:
   - `add()`
   - `multiplyWithScalar()`
   - `multiplyWithMatrix()`
   - `setElem()`
3. Run the regression tests (MatrixTest.java) to ensure that no tests have failed.
4. **Commit and push back to the repository.**

Part 2: Add Tests to `MatrixTest.java`

5. Review the code for `getElemIndexOutOfBounds()` in `MatrixTest.java` as an example of what you are being asked to do.
6. For each of the methods that you modified in Matrix.java, add a test method to MatrixTest.java (named using the same pattern as for `getElem()`. These methods should test that the exceptions are being properly thrown.
7. **Commit and push the fix back to the repository.**

Part 3: Assertions

1. Review the methods in Matrix.java. Determine **three (3)** places in the code where an assert statement would be helpful. See the `write()` method in Matrix.java for an example. Note: placing the same assertion in multiple methods still only counts as one assertion. ☺
2. Note the suggested locations in a comment called ASSERTION LIST at the top of the file. Include the method, the line number, and assertion condition
3. Add the assertion to the code
4. **Commit and push the fix back to the repository.**

Grading

The lab will be marked out of 4 points:

Task	2 Points	1 Point	0 Points
Part 1 and 2	The prescribed methods have been modified and tested	The prescribed methods have been modified but not tested, or not all the methods have been modified and tested.	No work done
Part 3	Three or more (reasonable) assertions have been specified.	At least one reasonable assertion has been specified	No assertions specified.

A Primer on Matrices

A matrix of size *m × n* is a 2D grid of values, typically decimal numbers, with *m* rows and *n* columns. Element *E[i,j]* of a matrix, refers to the value in row *i* and column *j*. Element *E[1,1]* is in the top left corner of the matrix and element *E[m,n]* is the bottom right corner of the matrix. A matrix can be negated, added with another matrix, multiplied by a scalar, or multiplied by another matrix.

Two matrices are equal if their elements-wise are the same. A matrix is negated by negating all the elements. E.g.,

$$\begin{bmatrix} -1 & -2 \\ -3 & -4 \end{bmatrix} = -\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

Two matrices can be added only if they have the same size. Addition is performed element-wise, meaning that if *C = A + B*, where *C*, *B*, and *A*, are matrices, then element *C[i, j] = A[i, j] + B[i, j]* for all elements of *C*. E.g.,

$$\begin{bmatrix} 6 & 8 \\ 10 & 12 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} + \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix}$$

Any matrix can be multiplied by a scalar, which is typically a decimal value. Scalar multiplication is performed element-wise, meaning that if *C = sA*, where s is a scalar value and A is a matrix, then *C[i,j] = s × A[i,j]*. E.g.,

$$\begin{bmatrix} -3 & -6 \\ -9 & -12 \end{bmatrix} = -3\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

Two matrices can be multiplied only if the width of the first matrix is equal to the height of the second matrix. If *C = AB*, where *A* is a matrix of size *m×n*, *B* is a matrix of size *n×p*, and *C* is a matrix of size *m×p*, then *C[i, j] = A[i, 1] × B[1, j] + A[i, 2] × B[2, j] + A[i, 3] × B[3, j] + . . . + A[i, n] × B[n, j]* E.g.,

$$\begin{bmatrix} 9 & 12 & 15 \\ 19 & 26 & 33 \\ 29 & 40 & 51 \\ 39 & 54 & 69 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \\ 7 & 8 \end{bmatrix}\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$