Basic Java Assignment: Equation Solver

Conditions

Solve the assignment individually, on your own.

Download latest OpenJDK if you have issues: [https://jdk.java.net/20/ (Links to an external site.)](https://jdk.java.net/20/)

Download (and rename to junit5all.jar): [https://repo1.maven.org/maven2/org/junit/platform/junit-platform-console-standalone/1.9.2/junit-platform-console-standalone-1.9.2.jar (Links to an external site.)](https://repo1.maven.org/maven2/org/junit/platform/junit-platform-console-standalone/1.9.2/junit-platform-console-standalone-1.9.2.jar)

Download junit5-demo.zip from Canvas and extract checkthat.jar. Both checkthat.jar and junit5all.jar may need to be on the classpath depending on which test files you run.

The [provided tester files](https://canvas.elte.hu/courses/35309/files/2231682/download?wrap=1) should be unzipped to do JUnit and checkthat testing: GaussianEliminationTest.java and GaussianEliminationStructureTest.java as well as GaussianEliminationTestSuite.java providing a suite combining the prior two tests. All tests should be satisfied to get full marks.

Exercise

You will be implementing a linear algebra equation solver. This is done using Guassian elimination to reduce a matrix to its row echelon form. In fact, we will reduce it to reduced row echelon form to assist with the final solving step called back substitution.

The steps are as follows.

Task 1: (4 points)

Create a linear.algebra.GaussianElimination class which stores a copy of a matrix as an array of arrays of type double.

* It has getters for the rows, columns and matrix itself.
* It has a setter only for matrix. It is an illegal argument exception to set a new matrix with a different number of rows and columns.
* The visible constructor takes the rows, columns and double matrix as input.
* The method rowEchelonForm computes [reduced row Echelon form based on pseudo-code (Links to an external site.)](https://en.wikipedia.org/wiki/Gaussian_elimination#Pseudocode).
  + Take care that this pseudo-code uses 1-based indexing while Java uses 0-based indexing.

Task 2: (6 points)

Since row echelon form revolves around 3 operations, implement the following separately.

* argMax which takes a row index and column index, and returns the index of the row coming after row index, containing the highest absolute value (Math.abs) in the given column index.
* swapRows which takes two row indexes as arguments and swaps the rows.
* multiplyAndAddRow as it is more efficient as shown in the pseudo-code, to both multiply and add in a combined step, implement this method which takes an addRow index, a mulRow index and a colIndex. It subtracts from addRow, mulRow multiplied by the ratio of addRow and mulRow at colIndex as seen in the inner loop of the pseudo-code.
* multiplyRow which takes a row index and column index and divides the row by the value at that row and column index. Unlike the pseudo-code which increments the row and column pivot indexes and continues, we add one line before incrementing, multiplyRow(rowIndex, colIndex) which will produce the reduced row echelon form useful for obtaining a solution.

Task 3: (4 points)

* Implement the backSubstitution method which goes through the rows in reverse order, then for every other row before the current row, uses multiplyAndAddRow to subtract the already solved variables. Before entering the inner loop, check that the diagonal element is non-zero, otherwise throw an IllegalArgumentException as it is a system of equations without a unique solution!
* Implement a print method so that it prints the set of linear equations in a human understandable way:

+2.0x+1.0y-1.0z=8.0

-3.0x-1.0y+2.0z=-11.0

-2.0x+1.0y+2.0z=-3.0

Task 4: (6 points)

* Implement an linear.EquationSolver class with a main which takes a command line argument in form 2,1,-1,8 -3,-1,2,-11 -2,1,2,-3 if corresponding to the previous example of equations.
* Create stringsToDoubles: class-level helper method which converts an array of String to an array of double.
* Parse the command line, create a GaussianElimination object, call print, then call rowEchelonForm, then print then backSubstition and print again to see that it is working. For the previous example, the reduced row echelon forms and final solution will look like:

+1.0x+0.3333333333333333y-0.6666666666666666z=3.6666666666666665

+0.0x+1.0y+0.4000000000000001z=2.6

+0.0x+0.0y+1.0z=-0.9999999999999999

+1.0x+0.0y+0.0z=2.0

+0.0x+1.0y+0.0z=3.0

+0.0x+0.0y+1.0z=-0.9999999999999999

Don't forget to run the provided tests to make sure they pass successfully![EquationSolver-Tests.zip](https://canvas.elte.hu/courses/35309/files/2231682/download?wrap=1)