TCSS143
Fundamentals of Object-Oriented Programming-Theory and Application
Programming Assignment 2

DUE: See Canvas Programming Assignment 2 link.

The purpose of this programming project is to demonstrate understanding and use of 2D Arrays, a special class known as a Module (only contains static methods and possibly constants), File Input/Output (I/O), try/catch blocks, multi-dimensional arrays, and the basic concept of exceptions.

You will submit a file named **Programming2.zip** (details below).

This program will perform a basic addition, subtraction, and multiplication of two matrices. The rules for matrix addition, subtraction, and multiplication are described at:

Matrix Addition (scroll about half way down this web page):
http://www.intmath.com/matrices-determinants/3-matrices.php

Matrix Multiplication:
http://www.intmath.com/matrices-determinants/4-multiplying-matrices.php

Your program will read data into each 2D array (4 in all, A & B for addition and subtraction, and C & D for multiplication) from an input file: in2.txt and send all output to: out2.txt. The input file will contain all the numbers required to fill 4 arrays (first two sets of numbers are for the A & B arrays, the remaining two sets of numbers are for the C & D arrays). Each array set of numbers will be preceded by 2 integers; the first for the row dimension and the second for the column dimension as illustrated below (all data will be integers):

```
3   3
16   -32    1
25      8   12
-5      0  -16

3   3
4     2    -1
11  44     13
25  -8       0

2   3
3  4  -2
-1 0  -2

3  2
3  5
-3 2
1  0
```

First 2 arrays added should produce a 3 X 3 matrix =

```
 20  -30   0
 36   52  25
 20   -8 -16
```

Subtraction of the first minus the second should produce a 3 X 3 matrix =

```
 12  -34   2
 14  -36  -1
-30    8 -16
```

Multiplying the remaining two arrays produces a 2 X 2 matrix =

```
-5  23
-5  -5
```

The 3 array operations (add, subtract, and multiply are all static methods) should be contained in their own class (ArrayMath) in a separate file named **ArrayMath.java:**

```
// All proper class documentation goes here according to the Documentation Requirements for this course.
public class ArrayMath {
        // all static methods (matrix math operations) should go here including their correct documentation.
}
```

The main method of class Assign_2 will access the ArrayMath methods by referring to the class in which they are defined (Note the period between the class name and method name):

```
        int[ ][ ] addResult = ArrayMath.addArrays(a, b);  // adds arrays a & b and returns the resulting array, etc.
```

The main method of class Assign_2 will:

- ➢ create the arrays (most likely 7 2D arrays in all- input A, B, C, D, & results for addition, subtraction, and multiplication)
- ➢ open the files for I/O
- ➢ populate the 4 arrays (A, B, C, & D) with data from the input file “in2.txt”.
- ➢ call the methods of ArrayMath to perform the matrix math and return the resulting array to be assigned to the result arrays in your main method.
- ➢ produce clear, meaningful output (out2.txt) to display:
  - ✓ **the arrays on which operations are being performed, i.e. A, B, C, & D**
  - ✓ **the operations resulting arrays**

You should name your class Assign_2 (file name will be Assign_2.java). You should also create <u>at least 2</u> helper methods:

- ✓ **one that is passed the input file Scanner and returns a 2D array**(see “2D Array Info.pdf” for example).
- ✓ **one that displays the arrays to an output file.**

You should **<u>NOT</u>** use throws FileNotFoundException. Open your files with try/catch blocks as follows:

```
// Multiline comments, imports, javadoc, class header, etc. goes here.
public static void main(String[] theArgs) {
    Scanner input = null;
    PrintStream output = null;
    String inFileName = "in2a.txt";
    String outFileName = "out.txt";
    boolean filesOk = false; // Indicates if the files are accessable.
    // other variables such as arrays, should also be declared here.

    try {
      input = new Scanner(new File(inFileName));
      output = new PrintStream(new File(outFileName));
      filesOk = true;
      }
    catch (FileNotFoundException e) {
      System.out.println("Can't open file - " + e);
    }
    if (filesOk) {
     ...  // This block is where the rest of the processing should
           // occur, such as, input of the arrays, matrix math operations and
           // results stored, and output of results.
     }
   } // End of main.
// Helper methods, such as input and output, go here
// and don’t forget the closing brace for the class
```

Document your program throughout based on the Documentation Requirements we discussed and is listed on Canvas (Continue to do so on ALL programming assignments).

Be sure to save both Assign_2.java and ArrayMath.java in the same folder. This way when you compile Assign_2.java, all calls or references to the methods of ArrayMath class will also be compiled.

You will note, I have included 2 sample input files for you to use at will (in2a.txt & in2b.txt). Create your own with perhaps much larger arrays if you like.

Because you will have 2 files to submit, you HAVE TO compress both files into a folder and submit this single compressed folder (Programming2.zip) to Canvas. To properly zip your files, see Compress and uncompress files.pdf under the Programming Assignment 2.

Upload Programming2.zip on Canvas by the due date/time.

For full credit, your output should (at very least) be formatted to display the input arrays (with appropriate labelling) and the resulting arrays (also labelled) Note the equal column widths, e.g.:

```
Array A:
    16   -32     1
    25     8    12
    -5     0   -16

Array B:
     4     2    -1
    11    44    13
    25    -8     0

Array C:
     3     4    -2
    -1     0    -2

Array D:
     3     5
    -3     2
     1     0

A + B =
    20   -30     0
    36    52    25
    20    -8   -16

A – B =
    12   -34     2
    14   -36    -1
   -30     8   -16

C X D =
    -5    23
    -5    -5
```

10 points extra credit if you format the output with the input and result arrays appearing left to right with operators and labeling as seen in the included example output file: "**IO Results Assignment 2b.pdf**." In these examples you might note that all columns' width are consistently sized based on the largest value. Including this consistent column width feature will earn another 10 points for a total of 20 points extra credit.