Project2: Matrix Multiplication Algorithms

PART 1 (*15% of the credit for Project2*): General algorithm for matrix multiplication

Submission type: in-person demo during lab hour; **`both partners must be present.`**
Due date: must demo *at* or *before* the beginning of the lab session on **Friday, 10/12/18**

Define a class named *MatrixWork* containing two *static* methods: *matrixProduct* and ***main***

- ***matrixProduct***: Given matrices A and B (i.e. two 2-dimensional arrays) of sizes $n \times \boldsymbol{k}$ and $\boldsymbol{k} \times m$ respectively, this method returns the product-matrix C (i.e. a 2-dimensional array) of size $n \times m$ (i.e. C= A·B). Elements of C are computed based on the following formula (each $c_{i,j}$ element is individually calculated as a sum). **Note**: indexing in this formula is ***natural***.

$$c_{i,j} = \sum_{l=1}^{k} a_{i,l} \cdot b_{l,j}$$

The method-signature is as follows (you can assume that #rows>0, #columns>0 in A and B):

public static int[][] matrixProduct (int [][] A, int[][] B)

Important: to multiply A and B matrices, the **number of columns in A must be the same as the number of rows in B**. Your method must first do this validity check and throw ***IllegalArgumentException*** type exception if the requirement is not met.

Reminder: number of rows in a two-dimensional array *arr* is *arr.lenght* and number of columns is *arr[0].length* (assuming all rows have the same number of elements, which is the case when representing a matrix).

- ***main***: In this method you will do the following (*no need to do any validity checks in main*):

1. Prompt the user to enter input-file's name, then input the file-name and set up a scanner.
2. Create two 2-dimensional arrays for matrices A and B, and fill A and B with numbers, inputting values from the input-file as described below.

   In the input-file you will have integer numbers only (there will not be any other symbols or signs). All numbers will be separated by whitespaces. The first two numbers will indicate the size of the A matrix (i.e. the number of rows and columns respectively), then values of the A matrix will follow (row by row). After that, next two numbers will indicate the size of matrix the B (i.e. the number of rows and columns respectively), then values of the B matrix will follow (row by row). *You can assume that the 4 numbers representing A and B matrix-sizes will be positive numbers.* An example file-content is given below; there is no formatting requirement for the file but this is a clear and visually convenient look for you to interpret your input data (you should **not** assume/expect any mandatory line-breaks or empty lines).

   2 3
   1 2 3
   4 5 6

   3 3
   1 2 3
   4 5 6
   7 8 9

3. Call *matrixProduct* method giving A and B matrices, and get the product-matrix (matrix C). Note: be aware of a potential *IllegalArgumentException* from *matrixProduct* – arrange to output an error message if this happens. ***You should not let your program crash.***
4. Output the content of matrix C on the screen, in an appropriate "matrix"-format: one line per matrix-row, separating values on the same row with space(s).

   For the above example your output should look like this (**only bold italic text is the output**):

   Product matrix:
   30 36 42 //$c_{1,1}$=1·1+2·4 + 3·7=30 $c_{1,2}$=1·2+2·5 + 3·8=36 $c_{1,3}$=1·3+2·6 + 3·9=42
   66 81 96 //$c_{2,1}$=4·1+5·4 + 6·7=66 $c_{2,2}$=4·2+5·5 + 6·8=81 $c_{2,3}$=4·3+5·6 + 6·9=96

Compile and run your program, test it thoroughly:
Make sure the program works as expected and gives correct results (there are many online tools for matrix-multiplication so you can use one of them to check your data).
Try different size input matrices, including matrices that have only one row or only one column. Also make sure that your program works as expected when input matrices are not eligible/valid for matrix-multiplication (they do not meet the above mentioned size-requirement).

How to get checked:
You will get a sheet with demo instructions. Run your program according to these instructions, keep the results of **all** tests on the screen, put your name on the board, and wait till the professor approaches you for demo. **Both partners must be present for the demo.**