Problem 139: Normal Math

Difficulty: Medium

Author: Steve Brailsford, Marietta, Georgia, United States

Originally Published: Code Quest 2021

Problem Background

Regardless of how advanced computer graphics are, it all boils down to a bunch of numbers. Graphics engines make use of large tables of numbers, called matrices, in order to keep track of every detail being rendered on your screen. Rotating, moving, or zooming into or out of an image can be represented by performing a number of mathematical operations on the matrix representing that image. However, matrix operations can be very complicated, and it's a good practice to check your work to ensure it is correct.

Lockheed Martin Aeronautics is trying to develop an augmented reality system to help train people to maintain the aircraft they create. However, the team working on the graphics overlays are having difficulty keeping things oriented the right way when the user moves their head. They'd like to add an error-detection system to help identify where the problem is coming from.

Problem Description

A computer graphics expert has suggested you calculate the Frobenius Norm of each matrix you create for use in the AR system. This is a number that can be used to identify when changes have taken place to a matrix. It's calculated using the formula below:

$$\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2}$$

...Or to put it in MUCH simpler terms, it's calculated by taking the absolute value of each element in the matrix, squaring them, adding them all together, then taking the square root of that sum. For example, let's consider a small 2x2 matrix:

$$A = \begin{bmatrix} 3 & -2 \\ 4 & 5 \end{bmatrix}$$

We can calculate the Frobenius Norm by taking the absolute value of each element...

$$|3| = 3 \quad |-2| = 2 \quad |4| = 4 \quad |5| = 5$$

...squaring those...

$$3^2 = 9 \quad 2^2 = 4 \quad 4^2 = 16 \quad 5^2 = 25$$

...adding all those squares together...

$$9 + 4 + 16 + 25 = 54$$

...then getting the square root of that.

$$\|A\|_F = \sqrt{54} \approx 7.35$$

Now why is this number useful? Because it's calculated using each of the individual values, any attempt to simply move the numbers in the matrix around - as might be done when rotating or translating it - will result in a matrix with the same Frobenius Norm. Scaling the values in a matrix (which might be done when zooming into or out of an image) will produce a different Frobenius Norm, but it will be equal to the original Frobenius Norm multiplied by the scale value.

$$A_R = \begin{bmatrix} 5 & 4 \\ = 2 & 3 \end{bmatrix}$$

$$\|A_R\|_F \approx 7.35$$

$$B = A * 4 = \begin{bmatrix} 12 & -8 \\ 16 & 20 \end{bmatrix}$$

$$\|B\|_F = \|A\|_F * 4 \approx 29.39$$

As a result of this, the Frobenius Norm can be used to quickly identify the presence of errors in matrix rotation, translation, or scaling. It may not tell you what the exact problem is, but it'll at least help you figure out where to start looking!

Sample Input

The first line of your program's input, received from the standard input channel, will contain a positive integer representing the number of test cases. Each test case will include:

- A line containing two positive integers, M and N, separated by spaces, representing the number of rows and columns (respectively) contained in a matrix. Values for both M and N will range from 1 to 30 inclusive.
- M lines, each containing N integers, separated by spaces, representing the content of the matrix.

```
4
2 2
3 -2
4 5
2 2
5 4
-2 3
2 2
12 -8
16 20
3 3
1 2 3
4 5 6
7 8 9
```

Sample Output

For each test case, your program must print the Frobenius Norm of the given matrix, rounded to two decimal places. Include any trailing zeroes.

```
7.35
7.35
29.39
16.88
```