Program na02: Matrix IO

Overview: Input matrices and vectors from a file, and output augmented matrices.

Introduction: Many of the problems that we will be solving will require us to work with systems of equations. Mathematically, we represent a system of linear equations as **Ax** = **b**, where **A** is the coefficient matrix and **b** is the constant vector. For example, the system

$$\begin{aligned} 2x - y + z &= 1 \\ 3x + 2z &= 5 \\ x + y + z &= 4 \end{aligned}$$

Has coefficient matrix **A** and constant vector **b** given by

$$\mathbf{A} = \begin{bmatrix} 2 & -1 & 1 \\ 3 & 0 & 2 \\ 1 & 1 & 1 \end{bmatrix} \text{ and } \mathbf{b} = \begin{bmatrix} 1 \\ 5 \\ 4 \end{bmatrix}$$

To present the entire system at a glance, we use the augmented matrix

$$[\mathbf{A} \mid \mathbf{b}] = \left[\begin{array}{ccc|c} 2 & -1 & 1 & 1 \\ 3 & 0 & 2 & 5 \\ 1 & 1 & 1 & 4 \end{array}\right]$$

We will need to input matrices from a file and output them to the screen.

Specifications: Two program files are required. The first file, named **`matIO.py`**, is a module containing two functions, **`getMat(`*`text`*`)`** and **`printAugMat(a,b)`**, which you must write. The second file is the main program file, named **`na02.py`**. The main program will import these functions from **`matIO.py`** with the line

```
from matIO import *
```

The first line of input will be a single integer giving the number of augmented matrices to be printed. The main program opens the text file, *`text`*, and calls **`getMat(`*`text`*`)`** to return a coefficient matrix **a**. A second call to **`getMat(`*`text`*`)`** will return a constant vector **b**. Note that there will always be one or more lines with no data between matrices in the input file. Then **`printAugMat(a,b)`** will be called to output the augmented matrix [**a**|**b**]. This will be repeated until all augmented matrices have been printed.

The function **`getMat(`*`text`*`)`** reads in the next matrix from the standard input file. Leading blank lines and comments (parts of lines following '#') are ignored. An array with entries of type float will be returned. The name 'array' is to be imported from **`numpy`**.

A vector (matrix with only one row or column) should be returned as an array with only one dimension, as we will not distinguish between row and column vectors. For example, both the 3x1 matrix [[0],[1],[2]] and the 1x3 matrix [[0, 1, 2]] should be returned as the vector [0,1,2]. To eliminate the unwanted dimension, use this syntax:

```
if 1 in b.shape:
     b = b.flatten() #Return a 1 dimensional array.
```

The function **`printAugMat(a,b)`** should not combine the two matrices **a** and **b** into a single matrix, but should print out each row in the format suggested by the sample output.

Input: All input will be done from a text file called **`input02.txt`**. This file will contain a number of coefficient matrices and corresponding constant vectors separated by one or more blank and/or commented lines. The size of each matrix will be determined by the number of rows and columns input. A vector can be entered as either a single row or a single column. Here is a sample input file:

```
2          #This is the number of systems to be input.
           #Blank lines and comments should be ignored.
#Here is the first coefficient matrix
1   2   #Comments to the right of numbers are ignored.
3   4

#First constant vector written as a 1x2 matrix
5   6

#Coefficient matrix 2
1.11111    2.22222    3.33333
4.44444    5.55555    6.66666
7.77777    8.88888    9.99999

#Second constant vector written as a 3x1 matrix
1.11111
2.22222
3.33333
```

Output: All output will be to the standard out window (the screen). For the input file above, here is the expected output: You will need to supply the characters '[', 'l', and ']'.

```
 [ 1.0 2.0 | 5.0 ]
 [ 3.0 4.0 | 6.0 ]

 [ 1.11111 2.22222 3.33333 | 1.11111 ]
 [ 4.44444 5.55555 6.66666 | 2.22222 ]
 [ 7.77777 8.88888 9.99999 | 3.33333 ]
```

***Extra for experts: na02X*.py** Pass in a format such as '5.2f' to be applied to all entries of the matrix and a string such as '[A|b] = ', to be printed halfway down the augmented matrix. The output will look like this:

```
>>>
[A|b] =  [  1.00  2.00 |  5.00 ]
         [  3.00  4.00 |  6.00 ]

         [  1.11  2.22  3.33 |  1.11 ]
[A|b] =  [  4.44  5.56  6.67 |  2.22 ]
         [  7.78  8.89 10.00 |  3.33 ]

>>>
```

Writeup: There is no separate writeup for this problem.

Checklist:

Module: **matIO.py**
Input File: **na02in.txt**
Program File: **na02.py** OR **na02X.py**