Midterm 3 Prep 3

Create a class Matrix2D. Matrices are very useful, almost universal tools in Mathematics and Engineering. A two dimensional matrix consists of two rows and two columns of floating point numbers $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$.

They support various operations. There is addition and subtraction:

$$\begin{pmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{pmatrix} + \begin{pmatrix} y_{1,1} & y_{1,2} \\ y_{2,1} & y_{2,2} \end{pmatrix} = \begin{pmatrix} x_{1,1}+y_{1,1} & x_{1,2}+y_{1,2} \\ x_{2,1}+y_{2,1} & x_{2,2}+y_{2,2} \end{pmatrix},$$

$$\begin{pmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{pmatrix} - \begin{pmatrix} y_{1,1} & y_{1,2} \\ y_{2,1} & y_{2,2} \end{pmatrix} = \begin{pmatrix} x_{1,1}-y_{1,1} & x_{1,2}-y_{1,2} \\ x_{2,1}-y_{2,1} & x_{2,2}-y_{2,2} \end{pmatrix}.$$

These are implemented with the dunders __add__, __sub__, __iadd__, and __isub__.

The negative of a matrix is defined by setting all its coefficients to their negatives. It is implemented with the dunder __neg__.

$$-\begin{pmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{pmatrix} = \begin{pmatrix} -x_{1,1} & -x_{1,2} \\ -x_{2,1} & -x_{2,2} \end{pmatrix}.$$

As always, the dunder needs to return an object of type Matrix.

We define a function that checks whether the matrix is the zero matrix, i.e. if **all** its coefficients are zero. Create a method isZero that returns True if the matrix is the zero matrix and False otherwise.

Matrix multiplication is a bit more involved.

$$\begin{pmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{pmatrix} \cdot \begin{pmatrix} y_{1,1} & y_{1,2} \\ y_{2,1} & y_{2,2} \end{pmatrix} = \begin{pmatrix} (x_{1,1}y_{1,1}+x_{1,2}y_{2,1} & x_{1,1}y_{1,2}+x_{1,2}y_{2,2} \\ x_{2,1}y_{1,1}+x_{2,2}y_{2,1} & x_{2,1}y_{1,2}+x_{2,2}y_{2,2} \end{pmatrix}.$$

Implement this with dunders __mul__ and __imul__.

A matrix can also be multiplied with a scalar:

$$\alpha \cdot \begin{pmatrix} y_{1,1} & y_{1,2} \\ y_{2,1} & y_{2,2} \end{pmatrix} = \begin{pmatrix} \alpha y_{1,1} & \alpha y_{1,2} \\ \alpha y_{2,1} & \alpha y_{2,2} \end{pmatrix}.$$

This has to be implemented with __rmul__.

A matrix has a determinant:

$$\text{det}\begin{pmatrix} y_{1,1} & y_{1,2} \\ y_{2,1} & y_{2,2} \end{pmatrix} = y_{1,1}y_{2,2} - y_{1,2}y_{2,1}.$$

Create a method determinant.

Finally, sometimes, the matrix has an inverse. The inverse is given by the following formula:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \text{det}(\begin{pmatrix} a & b \\ c & d \end{pmatrix})^{-1}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix}^{-1}.$$

Of course, if the determinant is zero, the inverse does not exist. You should raise a ValueError if the determinant is zero.