EX - 3 MATRIX ADT

T SADAKOPA RAMAKRISHNAN | IT - B | 3122225002109

AIM:

To create a Matrix class that creats a matrix object on which arithmetic operations such as addition, subtraction, scalar multiplication as well as finding its determinant and transpose. The Matrix object can be initialized using 2 integers for number of rows and columns and sets each element to a value of 0

ALGORITHM:

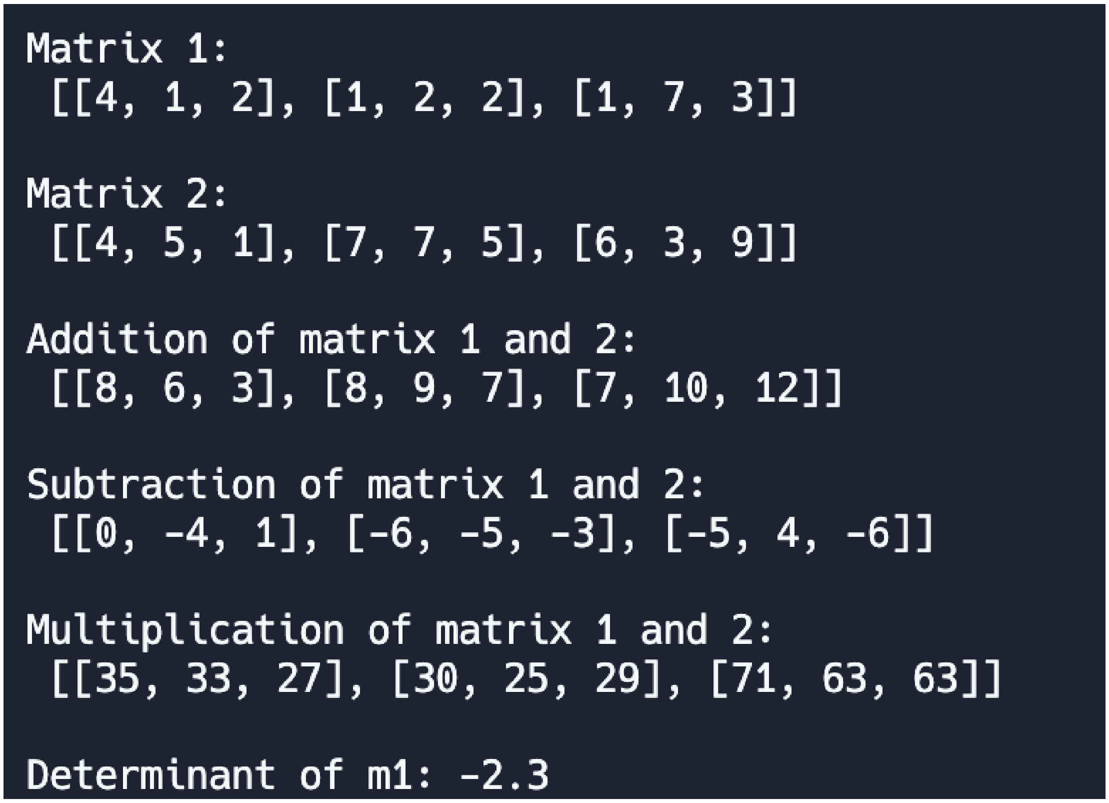
```
Step 1: Import the required modules: random and numpy
Step 2: Define a class Matrix with the following methods:
        a. init() method to initialize the instance variables of the Matrix class.
        b. getitem() method to get the value of a specific index in a matrix.
        c. setitem() method to assign a value to a particular index of a matrix.
        d. add() method to add two matrices.
        e. sub() method to subtract two matrices.
        f. mul() method to multiply two matrices.
        g. change_values() method to assign random values to each row and col of a
        matrix.
        h. det() method to find determinant of a matrix.
        i. str() method to print vector class object.
Step 3: Define test cases for the Matrix class:
        a. Create two matrices of 3x3.
        b. Assign random values to each row and col of the two matrices.
        c. Add the two matrices and print the result.
        d. Subtract the two matrices and print the result.
        e. Multiply the two matrices and print the result.
        f. Find the determinant of the first matrix and print the result.
Step 4: Execute the test cases if the file is not imported.
Step 5: End
```

CODE:

```
'''This module creates matrices of matrix class and
performs addition, subtraction, multiplication of two
matrices. This also finds the determinant of a matrix
and check if two matrices are equal.This is a part
```

```
of the exercises given under the course UIT2201 (Programming
and Data Structures).

In this source code I've executed my own logic and may contain bugs.
The source code has followed good coding practices.

Your comments and suggestions are welcome.

Created on Wed Apr 19 2023

Revised on Wed Apr 22 2023

Original Author: T. Sadakopa Ramakrishnan <sadakopa2210221@ssn.edu.in>
'''


#importing random module to assign random integer to row and col of a
matrix
import random

#importing numpy module to find determinant of a matrix.
import numpy as np

class Matrix:
    def __init__(self, r = 0, c = 0):
        '''Constructor to initialize the datamembers'''
        self.row = r
        self.col = c
        self.order = (self.row, self.col)
        self.matrix = []
        for i in range(r):
            l1 = [0] * c
            self.matrix.append(l1)

    def __getitem__(self, index):
        '''
        Overwriting __getitem__ to get the value of a specific index
        in a matrix
        '''
        return self.matrix[index[0]][index[1]]

    def __setitem__(self, index, value):
        '''
        Overwriting __setitem__ to assign a value to a particular
        index of a matrix
        '''
        self.matrix[index[0]][index[1]] = value
        return self.matrix

    def __add__(self, other):
        '''
        User defined method to add two matrices
        '''
        if self.order != other.order:
            raise ValueError("Dimensions of two matrices aren't
same.")
        else:
            result = [[0 for j in range(self.col)] for i in
range(self.row)]
            for i in range(self.row):
                for j in range(self.col):
                    result[i][j] = self.matrix[i][j] +
other.matrix[i][j]
            return result

    def __sub__(self, other):
        '''
        User defined method to subtract two matrices
        '''
        if self.order != other.order:
            raise ValueError("Dimensions of two matrices aren't
same.")
        else:
            result = [[0 for j in range(self.col)] for i in
range(self.row)]
            for i in range(self.row):
                for j in range(self.col):
                    result[i][j] = self.matrix[i][j] -
other.matrix[i][j]
            return result

    def __mul__(self, other):
        '''
        User defined method to multiply two matrices
        '''
        if self.order != other.order:
            raise ValueError("Dimensions of two matrices aren't
same.")
        else:
            result=[[0 for j in range(other.col)] for i in
range(self.row)]
            for i in range(self.row):
                for j in range(other.col):
                    for k in range(other.row):

result[i][j]+=self.matrix[i][k]*other.matrix[k][j]
        return result

    def change_values(self):
        '''
        User defined method to assign random values to each row and
col
        of a matrix
        '''
        for r in range(self.row):
            for c in range(self.col):
                self.matrix[r][c]+=random.randint(0,10)
        return self.matrix

    def det(self):
        '''
        User defined method to find determinant of a matrix.
        '''
        determinant = str(f'{np.linalg.det(self.matrix):.2e}')
        return determinant[0:4]


    def _str_(self):
        '''
        Overwriting __str__ to print vector class object.
        '''
        return str(self.matrix)
#End of Matrix Class

#Test cases for above source code.

if __name__ == "__main__":
    #This part of the program will not be executed when the file is
imported.

    #Creating two matrices of 3x3
    l1 = Matrix(3,3)
    l2 = Matrix(3,3)
    #assigning values to each row and col to matrix 1 and printing
matrix 1
    mat1=l1.change_values()
    print("Matrix 1:\n",mat1)
    print()

    #assigning values to each row and col to matrix 2 and printing
matrix 2
    mat2=l2.change_values()
    print("Matrix 2:\n",mat2)
    print()

    #Adding two matrices and printing it
    m_add=(l1+l2)
    print("Addition of matrix 1 and 2:\n",m_add)
    print()

    #Subtracting two matrices and printing it
    m_sub=(l1-l2)
    print("Subtraction of matrix 1 and 2:\n",m_sub)
    print()

    #Multiplying two matrices and printing it
    m_mul=l1*l2
    print("Multiplication of matrix 1 and 2:\n",m_mul)
    print()

    #Finding determinant of matrix 1
    det1 = l1.det()
    print("Determinant of m1:", det1)
    print()
```

OUTPUT:



```
Matrix 1:
 [[4, 1, 2], [1, 2, 2], [1, 7, 3]]

Matrix 2:
 [[4, 5, 1], [7, 7, 5], [6, 3, 9]]

Addition of matrix 1 and 2:
 [[8, 6, 3], [8, 9, 7], [7, 10, 12]]

Subtraction of matrix 1 and 2:
 [[0, -4, 1], [-6, -5, -3], [-5, 4, -6]]

Multiplication of matrix 1 and 2:
 [[35, 33, 27], [30, 25, 29], [71, 63, 63]]

Determinant of m1: -2.3
```