

**МИНОБРНАУКИ РОССИИ**

**федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технологический университет «СТАНКИН»**

**(ФГБОУ ВО «МГТУ «СТАНКИН»)**

|  |  |
| --- | --- |
| **Институт  информационных  технологий** | **Кафедра**  **информационных систем** |

**Основная образовательная программа 09.03.02  
«Информационные системы и технологии»**

**Отчет о выполнении лабораторной работы №3**

**по дисциплине «Веб-программирование»**

|  |  |
| --- | --- |
| Студент  группы ИДБ-19-08 | Николаева А.Ю. |
|  |  |
| Старший преподаватель | Адамова Ю.С. |

**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ 3](#_Toc87257059)

[ЗАДАНИЕ 4](#_Toc87257060)

[РЕШЕНИЕ ЗАДАНИЯ 4](#_Toc87257061)

[РЕЗУЛЬТАТЫ РАБОТЫ ПРОГРАММЫ 6](#_Toc87257062)

[ЗАКЛЮЧЕНИЕ 8](#_Toc87257063)

ВВЕДЕНИЕ

В рамках лабораторной работы №3 необходимо ознакомиться с основами работы c классами на языке Python, освоить способы создания методов класса и работы с ними.

ЗАДАНИЕ

Написать класс Matrix, который будет прообразом математического объекта (для простоты квадратная матрица). Определить для него методы сравнения на основе детерминанта этой матрицы - если детерминант первой матрицы больше, то и матрица больше. Для остальных операций аналогично. Так же реализовать операции сложения, умножения двух матриц.

РЕШЕНИЕ ЗАДАНИЯ

import copy

class Matrix:

def \_\_init\_\_(self, mat):

self.mat = mat

@staticmethod

def str(A):

for j in A:

for i in j:

print(f"{i:2}", end=" ")

print()

return ''

@staticmethod

def minor(mas,k):

res=[]

for r in mas[1:]:

row=[]

for j in range(len(r)):

if j != k:

row.append(r[j])

res.append(row)

return res

@staticmethod

def det(mas):

n=len(mas)

if n==2:

return mas[0][0]\*mas[1][1]-mas[0][1]\*mas[1][0]

determinant = 0

sign = 1

for i in range(n):

determinant=determinant+sign\*mas[0][i]\*Matrix.det(Matrix.minor(mas,i))

sign=-sign

return determinant

def \_\_gt\_\_(self, other):

return (Matrix.det(self.mat) > Matrix.det(other.mat))

def \_\_lt\_\_(self, other):

return (Matrix.det(self.mat) < Matrix.det(other.mat))

def \_\_eq\_\_(self, other):

return (Matrix.det(self.mat) == Matrix.det(other.mat))

def \_\_add\_\_(self, other):

A = copy.deepcopy(self.mat)

for i in range(len(A)):

for i2 in range(len(other.mat[i])):

A[i][i2] = self.mat[i][i2] + other.mat[i][i2]

return Matrix.str(A)

def \_\_mul\_\_(self, other):

s = 0

A = copy.deepcopy(self.mat)

for i in range(len(A)):

for i2 in range(len(other.mat[i])):

for z in range(len(A[i2])):

s = s + self.mat[i][z] \* other.mat[z][i2]

A[i][i2] = s

s = 0

return Matrix.str(A)

m1 = Matrix([[1,1],

[2,2]])

m2 = Matrix([[1,2],

[3,4]])

if m1 > m2:

print('m1 > m2')

if m1 < m2:

print('m1 < m2')

if m1 == m2:

print('m1 = m2')

print('Результат сложения двух матриц: ')

print( m1 + m2)

print('Результат произведения двух матриц: ')

print(m1 \* m2)

РЕЗУЛЬТАТЫ РАБОТЫ ПРОГРАММЫ

Результат работы программы представлен на рисунке 1.

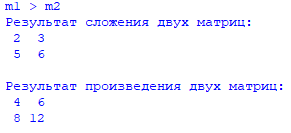


Рис.1. Результат работы программы для первого случая

Для того, чтобы продемонстрировать работу метода def \_\_lt\_\_(self, other) изменим введенные значения матриц на следующие значения:

m1 = Matrix([[1,2,3],

[6,5,4],

[7,8,9]])

m2 = Matrix([[1,2,5],

[3,4,20],

[8,1,2]])

В результате будут выведены значения как на рисунке 2.

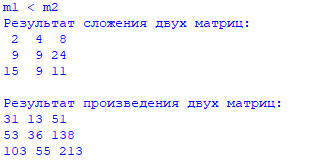


Рис.2. Результат работы программы для второго случая

Для того, чтобы продемонстрировать работу метода def \_\_eq\_\_(self, other) также изменим введенные значения матриц на следующие значения:

m1 = Matrix([[2,1,-1],

[1,2,3],

[1,2,3]])

m2 = Matrix([[1,3,0],

[-1,2,1],

[-2,4,2]])

В результате будут выведены значения как на рисунке 3.

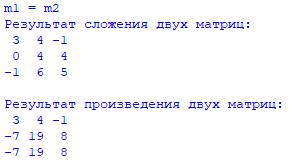


Рис.3. Результат работы программы для третьего случая

ЗАКЛЮЧЕНИЕ

В рамках выполнения лабораторной работы №3 освоены навыки работы с классами на языке Python, были созданы методы класса с перегрузкой операторов с помощью магических методов.