**Федеральное государственное образовательное бюджетное**

**учреждение высшего образования**

**«Санкт-Петербургский государственный университет**

**телекоммуникаций им. проф. М. А. Бонч-Бруевича»**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Факультет

инфокоммуникационных сетей и систем

Кафедра

Защищенных сетей и систем

**КУРСОВАЯ РАБОТА**

учебная дисциплина «Технологии и методы программирования»

Тема: **«**Реализация расчета матриц**»**

Выполнил(а) студент(ка) II курса группы

ИКБ-63 Рогозов Д.Ю.

Проверил Штеренберг С.И.

\_\_\_\_\_\_\_\_\_\_(подпись)

**Оглавление**

[**Введение** 3](#_Toc515485437)

[**1** **Реализация шифра Вернама.** 4](#_Toc515485438)

[**Заключение** 6](#_Toc515485439)

[**Список использованных интернет-ресурсов** 7](#_Toc515485440)

[**Приложение** 8](#_Toc515485441)

# **Введение**

В этой курсовой работе, будет рассмотрен пример реализации программы по расчёту матриц

**Задание по курсовой работе:** выполнить реализацию расчёта матриц на языке программирования “Python”.

# **Реализация Алгоритма.**

С помощью библиотеки numpy была реализованна

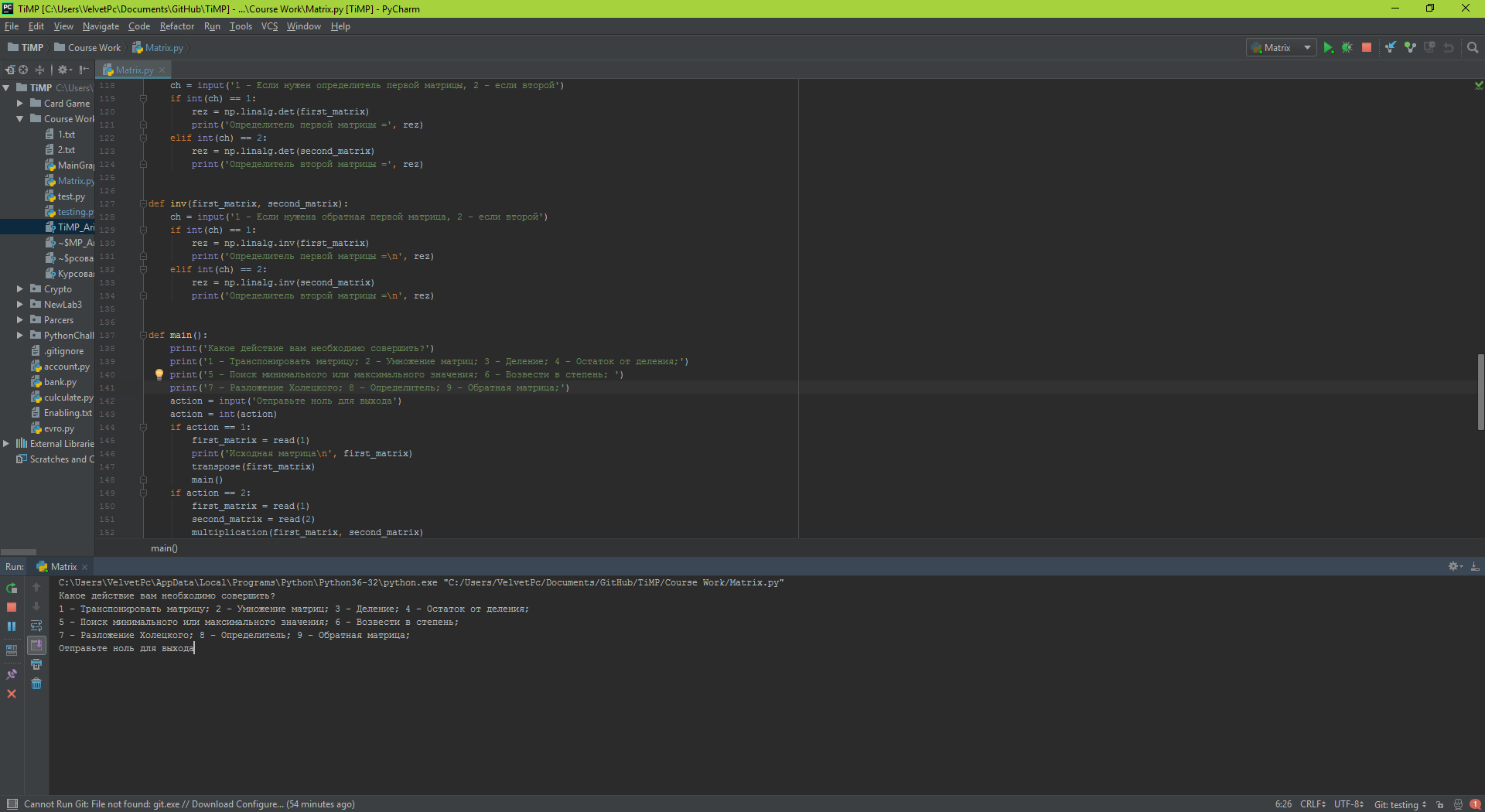


Рисунок 1. Интерфейс.

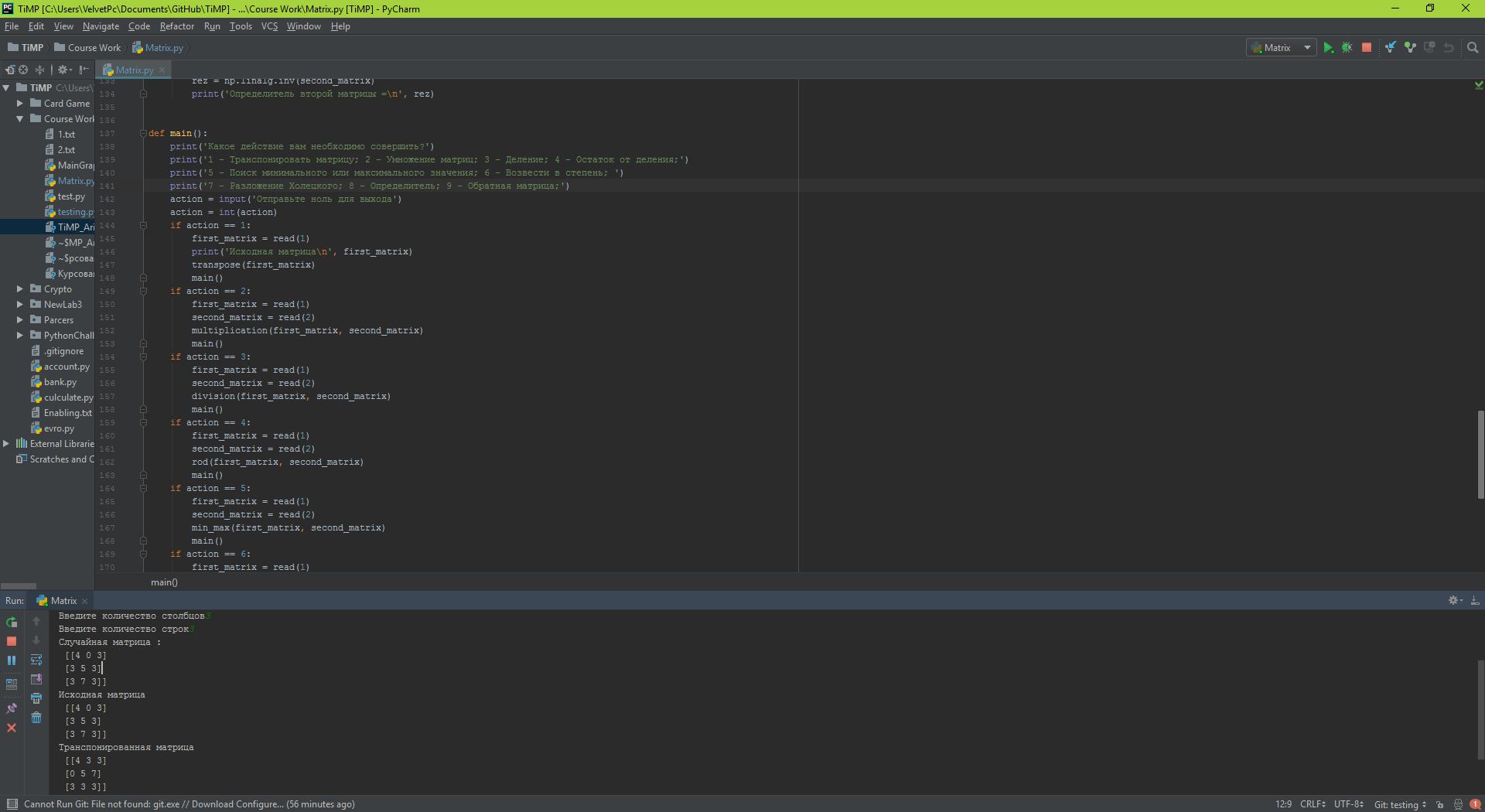


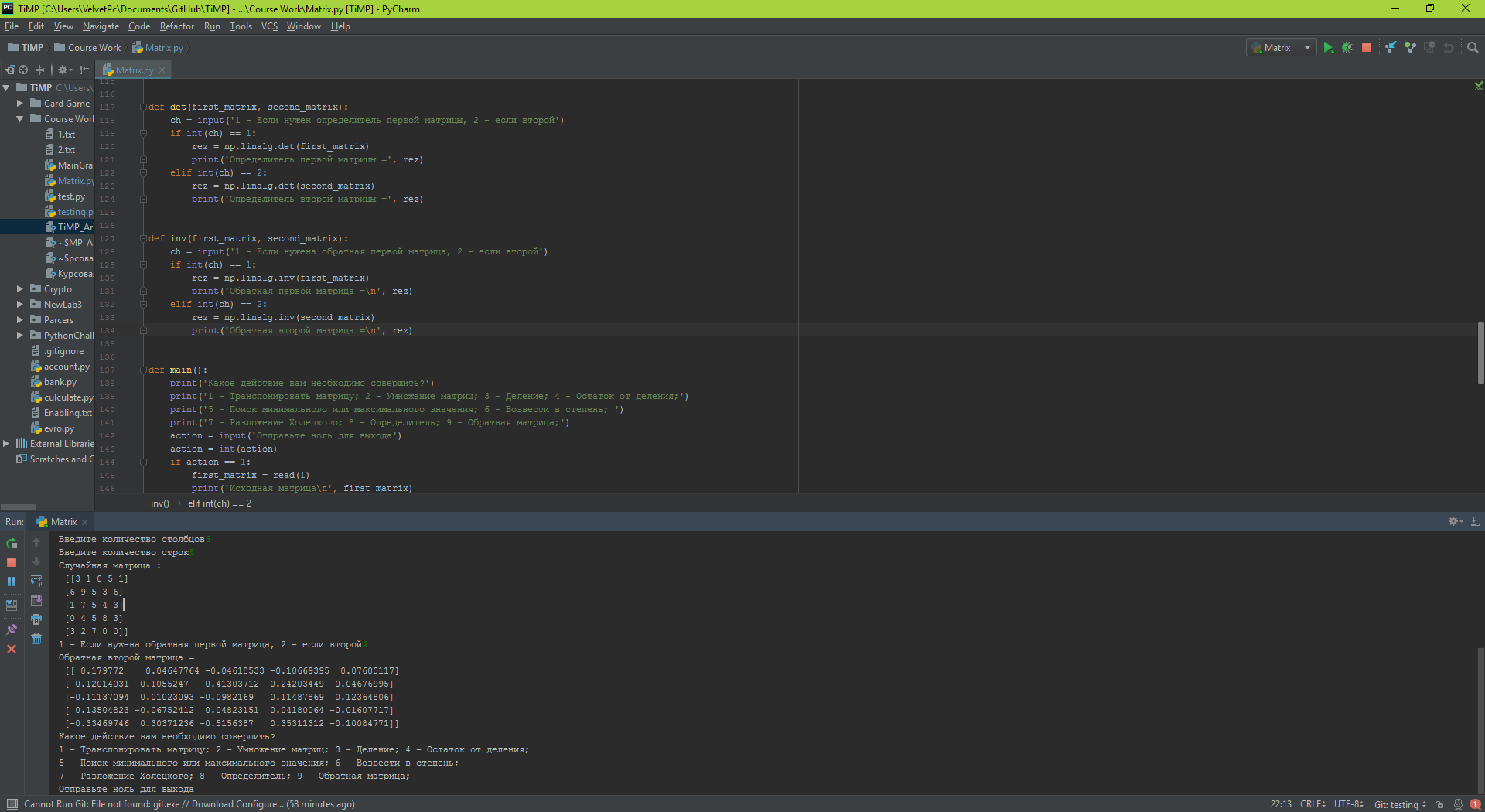
Рисунок 2. Пример транспонирования случайно сгенерированной матрицы

Рисунок 3. Пример нахождения обратной матрицы

# **Заключение**

В данной курсовой работе была выполнена программа выполняющая различные действия с матрицами на языке программирования “Python”.

# **Список использованных интернет-ресурсов**

<https://pythonworld.ru/numpy> - Информация и примеры решений NumPy

<http://www.numpy.org/> - Исходная документация

# **Приложение**

import numpy as np  
with open('1.txt', 'w') as f:  
 pass

with open('2.txt', 'w') as f:  
 pass

def read(num):  
 ch1 = input('Не хотите ввести случайную матрицу? 1/0')  
 if int(ch1) == 1:  
 matrix = random()  
 if num == 1:  
 first\_matrix = matrix  
 return first\_matrix  
 if num == 2:  
 second\_matrix = matrix  
 return second\_matrix  
 print('Введите вашу матрицу в файл 1.txt находящийся в папке с программой')  
 print('Вторую соответственно в файл 2.txt')  
 print('Все числа должны быть введены через запятую, кроме последнего элемента в строке')  
 print('Пример')  
 print('''1,2,3  
4,5,6  
7,8,9  
 ''')  
 ch = input('Отправьте единицу когда матрица будет введена')  
 if int(ch) == 1:  
 first\_matrix = np.genfromtxt("1.txt", delimiter=",")  
 second\_matrix = np.genfromtxt("2.txt", delimiter=",")  
 if num == 1:  
 return first\_matrix  
 if num == 2:  
 return second\_matrix  
  
  
def random():  
 columns = input('Введите количество столбцов')  
 rows = input('Введите количество строк')  
 columns = int(columns)  
 rows = int(rows)  
 matrix = np.random.randint(0, 10, (rows, columns))  
 print('Случайная матрица :\n', matrix)  
 return matrix  
  
  
def transpose(first\_matrix):  
 a = first\_matrix.transpose()  
 print('Транспонированная матрица \n', a)  
  
  
def multiplication(first\_matrix, second\_matrix):  
 ch = input('1 - Если нужно умножить первую матрицу на вторую, 2 - если наоборот')  
 if int(ch) == 1:  
 rez = first\_matrix.dot(second\_matrix)  
 print(rez)  
 elif int(ch) == 2:  
 rez = second\_matrix.dot(first\_matrix)  
 print(rez)  
  
  
def division(first\_matrix, second\_matrix):  
 ch = input('1 - Если нужно разделить первую матрицу на вторую, 2 - если наоборот')  
 if int(ch) == 1:  
 rez = first\_matrix/second\_matrix  
 print(rez)  
 elif int(ch) == 2:  
 rez = second\_matrix/first\_matrix  
 print(rez)  
  
  
def rod(first\_matrix, second\_matrix):  
 ch = input('1 - Если нужно разделить первую матрицу на вторую, 2 - если наоборот')  
 if int(ch) == 1:  
 rez = first\_matrix % second\_matrix  
 print(rez)  
 elif int(ch) == 2:  
 rez = second\_matrix % first\_matrix  
 print(rez)  
  
  
def min\_max(first\_matrix, second\_matrix):  
 ch = input('1 - Если нужно найти максимальный элемент, 2 - если минимальный')  
 ch1 = input('1 - Для первого массива, 2 - для второго')  
 if int(ch) == 1:  
 if int(ch1) == 1:  
 rez = np.max(first\_matrix)  
 print('Максимальный элемент первого массива =', rez)  
 elif int(ch1) == 2:  
 rez = np.max(second\_matrix)  
 print('Максимальный элемент второго массива =', rez)  
 elif int(ch) == 2:  
 if int(ch1) == 1:  
 rez = np.min(first\_matrix)  
 print('Минимальный элемент первого массива =', rez)  
 elif int(ch1) == 2:  
 rez = np.min(second\_matrix)  
 print('Минимальный элемент второго массива =', rez)  
  
  
def exp(first\_matrix, second\_matrix):  
 ch = input('1 - Если нужно возвести первую матрицу во вторую, 2 - если наоборот')  
 if int(ch) == 1:  
 rez = first\_matrix \*\* second\_matrix  
 print(rez)  
 elif int(ch) == 2:  
 rez = second\_matrix \*\* first\_matrix  
 print(rez)  
  
  
def chol(first\_matrix, second\_matrix):  
 ch = input('1 - Если нужно разложить первую матрицу, 2 - если вторую')  
 if int(ch) == 1:  
 rez = np.linalg.cholesky(first\_matrix)  
 print(rez)  
 elif int(ch) == 2:  
 rez = np.linalg.cholesky(second\_matrix)  
 print(rez)  
  
  
def det(first\_matrix, second\_matrix):  
 ch = input('1 - Если нужен определитель первой матрицы, 2 - если второй')  
 if int(ch) == 1:  
 rez = np.linalg.det(first\_matrix)  
 print('Определитель первой матрицы =', rez)  
 elif int(ch) == 2:  
 rez = np.linalg.det(second\_matrix)  
 print('Определитель второй матрицы =', rez)  
  
  
def inv(first\_matrix, second\_matrix):  
 ch = input('1 - Если нужена обратная первой матрица, 2 - если второй')  
 if int(ch) == 1:  
 rez = np.linalg.inv(first\_matrix)  
 print('Обратная первой матрица =\n', rez)  
 elif int(ch) == 2:  
 rez = np.linalg.inv(second\_matrix)  
 print('Обратная второй матрица =\n', rez)  
  
  
def main():  
 print('Какое действие вам необходимо совершить?')  
 print('1 - Транспонировать матрицу; 2 - Умножение матриц; 3 - Деление; 4 - Остаток от деления;')  
 print('5 - Поиск минимального или максимального значения; 6 - Возвести в степень; ')  
 print('7 - Разложение Холецкого; 8 - Определитель; 9 - Обратная матрица;')  
 action = input('Отправьте ноль для выхода')  
 action = int(action)  
 if action == 1:  
 first\_matrix = read(1)  
 print('Исходная матрица\n', first\_matrix)  
 transpose(first\_matrix)  
 main()  
 if action == 2:  
 first\_matrix = read(1)  
 second\_matrix = read(2)  
 multiplication(first\_matrix, second\_matrix)  
 main()  
 if action == 3:  
 first\_matrix = read(1)  
 second\_matrix = read(2)  
 division(first\_matrix, second\_matrix)  
 main()  
 if action == 4:  
 first\_matrix = read(1)  
 second\_matrix = read(2)  
 rod(first\_matrix, second\_matrix)  
 main()  
 if action == 5:  
 first\_matrix = read(1)  
 second\_matrix = read(2)  
 min\_max(first\_matrix, second\_matrix)  
 main()  
 if action == 6:  
 first\_matrix = read(1)  
 second\_matrix = read(2)  
 exp(first\_matrix, second\_matrix)  
 main()  
 if action == 7:  
 first\_matrix = read(1)  
 second\_matrix = read(2)  
 chol(first\_matrix, second\_matrix)  
 main()  
 if action == 8:  
 first\_matrix = read(1)  
 second\_matrix = read(2)  
 det(first\_matrix, second\_matrix)  
 main()  
 if action == 9:  
 first\_matrix = read(1)  
 second\_matrix = read(2)  
 inv(first\_matrix, second\_matrix)  
 main()  
 if action == 0:  
 exit('Выход')  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()