Лабораторная работа №1

«Реализация элементов линейной алгебры на Python»

(трудоемкость 4 часа[1])

Цель работы: познакомиться с возможностями языка Python по обработке матриц и их элементов на инструментов библиотеки NumPy[2].

Порядок выполнения работы:

1) Ознакомиться с примерами реализации различных операций, производимых с матрицами на Python (см. стр. 16-30[3])

2) Выполнить самостоятельные задания согласно варианту:

номер варианта	**Задача 1** (см. стр. 30-31)	**Задача 2** (см. стр. 32)	**Задача 3** (см. стр. 33)	**Задача 4** (см. стр. 34)
1	21	28	36	44
2	22	29	37	45
3	23	30	38	46
4	24	31	39	47
5	25	32	40	48
6	26	33	41	44
7	27	28	36	45
8	21	29	37	46
9	22	30	38	47
10	23	31	39	48
11	24	32	40	44
12	25	33	41	45
13	26	28	36	46
14	27	29	37	47
15	21	30	38	48
16	22	31	39	44
17	23	32	40	45
18	24	33	41	46
19	25	28	36	47
20	26	29	37	48
21	27	30	38	44
22	21	31	39	45
23	22	32	40	46
24	23	33	41	47
25	24	28	36	48

1 4 часа = 2 пары

2 Эта лабораторная работа основана на примерах из книги:

Балджы, А. С. Математика на Python : учебно-методическое пособие : [16+] / А. С. Балджы, М. Б. Хрипунова, И. А. Александрова. – Москва : Прометей, 2018. – Часть 1. Элементы линейной алгебры и аналитической геометрии. – 76 с. : табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=494849 (дата обращения: 24.01.2023). – Библиогр. в кн. – ISBN 978-5-907003-86-6. – Текст : электронный.

Внимание: доступ к указанной книге осуществляется посредством электронной библиотеки biblioclub.ru и может быть ограничен, поэтому в конце документа размещены скриншоты книги.

3 Здесь и далее указаны оригинальные номера страниц книги (видны на скриншотах).

3) Оформить отчет, содержащий:
   - титульный лист;
   - цель работы;
   - задание для своего варианта;
   - листинг программного кода на языке программирования Python с реализацией решения задач 1-4 и распечаткой результатов. Программный код снабдить смысловыми комментариями;
   - вывод по итогам выполнения лабораторной работы.

Федеральное государственное образовательное бюджетное учреждение высшего образования
«ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ
РОССИЙСКОЙ ФЕДЕРАЦИИ»

А. С. БАЛДЖЫ М. Б. ХРИПУНОВА
И. А. АЛЕКСАНДРОВА

МАТЕМАТИКА НА PYTHON. ЧАСТЬ I

ЭЛЕМЕНТЫ ЛИНЕЙНОЙ АЛГЕБРЫ И АНАЛИТИЧЕСКОЙ ГЕОМЕТРИИ

Учебно-методическое пособие

Москва 2018

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ 4

ТЕМА1. Установка Python в составе Anaconda для Windows.
- Знакомство с Python. Основные команды. Синтаксис 6
- Операторы Python 9
- Именные функции, инструкции def и return, pass, import . . . 9
- Аргументы функции 10
- Input, print, условные конструкции, циклы 10
- Библиотеки Python 14
  - Библиотека Math. 14
  - Библиотека Mathplotlib 16
  - Библиотека Sympy 16
- Обработка матриц в Python. Библиотека NumPy 16
- Создание матриц 17

ТЕМА 2. Элементы линейной алгебры. Примеры решения задач 21
- Операции над матрицами и их свойства 21
  - Ввод и вывод матрицы (пакет Numpy). 21
  - Умножение матриц. 22
  - Возведение матрицы в степень. 23
  - Транспонирование матрицы. 24
- Задания для самостоятельной работы 30
- Системы линейных уравнений (СЛУ) 37
- Задания для самостоятельной работы 44

ТЕМА 3. Элементы аналитической геометрии. 47
- Линейные пространства. Векторы на плоскости и в пространстве. Операции над векторами. 47
- Прямые на плоскости в пространстве 53
- Решить задачи самостоятельно 56
- Линейные операторы 58
- Квадратичные формы 60
- Кривые второго порядка 62
- Плоскости и прямые в пространстве 69
- Задания для самостоятельной работы 73

ЛИТЕРАТУРА 74

Существуют две возможные реализации цикла в Python: while и for. Мы будем использовать цикл for. Если нужно указать некоторый диапазон чисел, удобно воспользоваться функцией range(). Рассмотрим пример

```
In [15]:  for i in range(5):
              print(i)
```

Данный фрагмент кода выводит на экран числа в диапазоне от 0 до 4. Отметим, что на Python нумерация начинается с нуля, соответственно 5ым значением в нашем примере будет 4. диапазоне. Переменной i в цикле for последовательно передаются значения из заданного диапазона, и они выводятся на экран:

```
In [15]:  for i in range(5):
              print(i)

0
1
2
3
4
```

БИБЛИОТЕКИ PYTHON

Библиотека Math

Для вычислений с действительными числами применяется библиотека Math, которая содержит много полезных функций. Ниже приведены основные функции, которые включены в эту библиотеку.

math.ceil(X)	— округление до ближайшего большего числа.
math.fabs(X)	— модуль X.
math.factorial(X)	— факториал числа X.
math.floor(X)	— округление вниз.
math.fmod(X, Y)	— остаток от деления X на Y.

math.fsum	— сумма всех членов последовательности
math.isfinite(X)	— является ли X числом.
math.isinf(X)	— является ли X бесконечностью.
math.modf(X)	— возвращает дробную и целую часть числа X. Оба числа имеют тот же знак, что и X.
math.trunc(X)	— усекает значение X до целого.
math.exp(X)	— e^X.
math.expm1(X)	— e^X — 1
math.log(X, [base])	— логарифм X по основанию base. Если base не указан, вычисляется натуральный логарифм.
math.log10(X)	— логарифм X по основанию 10.
math.log2(X)	— логарифм X по основанию 2.
math.pow(X, Y)	— X^Y.
math.sqrt(X)	— квадратный корень из X.
math.acos(X)	— арккосинус X. В радианах.
math.asin(X)	— арксинус X. В радианах.
math.atan(X)	— арктангенс X. В радианах.
math.cos(X)	— косинус X (X указывается в радианах
math.sin(X)	— синус X (X указывается в радианах
math.tan(X)	— тангенс X (X указывается в радианах
math.hypot(X, Y)	— вычисляет гипотенузу треугольника с катетами X и Y (math.sqrt(x * x + y * y
math.degrees(X)	— конвертирует радианы в градусы.
math.radians(X)	— конвертирует градусы в радианы.
math.pi — pi = 3,1415926...	Число ϖ
math.e — e = 2,718281...	Число e

Для подключения библиотеки используем **import**. Загрузим библиотеку **math** и выведем число e.

```
import math
math.e
```

```
2.718281828459045
```

Можно это сделать, используя псевдоним библиотеки. Например, так

```
import math as m
m.e
```

```
2.718281828459045
```

*Библиотека **Mathplotlib***

Библиотека matplotlib — это набор методов для создания двумерной графики для языка программирования python. Эта библиотека включает множество различных методов, мы остановимся только на некоторых из них. Графика в Mathplotlib представляет собой отдельную тему, подробно рассмотренную, например, в пособии [12]. Остановимся подробней на ней при рассмотрении кривых II порядка.

*Библиотека **Sympy***

Библиотека Sympy предназначена для символьных вычислений. В нее входят библиотеки для работы с матрицами, с векторами, прямыми и плоскостями, для решения статистических и теоретико-вероятностных задач.

ОБРАБОТКА МАТРИЦ В PYTHON. БИБЛИОТЕКА NUMPY

Ниже мы рассмотрим основные операции над матрицами на языке Python.

Напомним, что матрицей размера m x n называется таблица чисел, содержащая m строк и n столбцов. Числа, составляющие матрицу, называются элементами матрицы и в общем виде обозначаются где i — номер строки, j — номер столбца. Определения и операции матричной алгебры подробно описаны, например, в [1], глава 1.

Для работы с матричной алгеброй в Python разработано специальное расширение языка Python — это библиотека NumPy. В программировании матрицы принято называть двумерными массивами, векторы одномерными массивами, возможна также работа с массивами размерности больше двух. Наиболее важным объектом NumPy является ndarray (однородный массив).

СОЗДАНИЕ МАТРИЦ

Рассмотрим один из способов создания матрицы в NumPy. Используем функцию numpy.array() пита numpy.ndarray . При этом в квадратных скобках, через запятую указываются строки матрицы. Вся последовательность строк также заключается в квадратные скобки, а аргумент функции указывается в круглых скобках. Например, в результате выполнения следующего кода будет создан вектор (одномерный массив) из трех элементов 1,2,3.

```
In [13]: import numpy as np
         a = np.array([[1, 2, 3]])
         a

Out[13]: array([[1, 2, 3]])
```

Ниже приведен пример создания матрицы b размером 2x3.

```
In [11]: b=np.array([[1.5,2,3],[4,5,6]])
         b

Out[11]: array([[1.5, 2. , 3. ],
                [4. , 5. , 6. ]])
```

Чтобы посмотреть какого типа объект можно использовать функцию type.

```
type(a)
Out[14]: numpy.ndarray
```

Элементами матрицы могут быть величины различных типов. В этом случае тип элементов должен быть указан как параметр после определения всех элементов матрицы. Так, ниже рассмотрен пример матрицы, состоящей из комплексных чисел, тип данных np.complex.

```
In [17]: b=np.array([[1.5,2,3],[4,5,6]],dtype=np.complex)
         b
Out[17]: array([[1.5+0.j, 2. +0.j, 3. +0.j],
                [4. +0.j, 5. +0.j, 6. +0.j]])
```

Функция array() не единственная функция для создания массивов.

Обычно на практике элементы массива вначале неизвестны, а массив, в котором они будут храниться, уже нужен. Поэтому имеется несколько функций для того, чтобы создавать массивы с каким-то исходным содержимым (по умолчанию тип создаваемого массива — float64).

Функция zeros() создает массив из нулей, а функция ones() — массив из единиц. Обе функции принимают в качестве аргументов размеры матрицы, и необязательный аргумент dtype — тип элементов матрицы:

```
In [21]: np.zeros((3,5))

Out[21]: array([[0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0.],
                [0., 0., 0., 0., 0.]])
```

```
In [22]: np.ones((2,2,2))
Out[22]: array([[[1., 1.],
                 [1., 1.]],

                [[1., 1.],
                 [1., 1.]]])
```

Функция eye() создаёт единичную матрицу (двумерный массив)

```
In [23]: np.eye(5)

Out[23]: array([[1., 0., 0., 0., 0.],
               [0., 1., 0., 0., 0.],
               [0., 0., 1., 0., 0.],
               [0., 0., 0., 1., 0.],
               [0., 0., 0., 0., 1.]])
```

Функция empty() создает матрицу нужного размера, исходное содержимое формируется случайно. Например, при создании матрицы размером 3Х3, получаемые значения могут быть такими:

```
In [24]: np.empty((3,3))

Out[24]: array([[2.37663529e-312, 2.14321575e-312, 2.37663529e-312],
               [2.56761491e-312, 8.48798317e-313, 9.33678148e-313],
               [8.70018275e-313, 2.02566915e-322, 2.47032823e-323]])
```

Для создания последовательностей чисел, в NumPy имеется функция arange(). Она формирует последовательность чисел от начального до конечного значения, включая первое и исключая последнее, с некоторым шагом. Например, последовательность чисел от 10 до 30 с шагом 5 будет иметь следующий вид:

```
In [27]: np.arange(10,30,5)

Out[27]: array([10, 15, 20, 25])
```

А последовательность чисел от 0 до 1 с шагом 0,1 будет иметь следующий вид:

```
In [26]: np.arange(0,1,0.1)

Out[26]: array([0. , 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9])
```

Обратим внимание, что числа имеющие дробную часть записываются через точку, например, 0.1.

С официальной документацией по NumPy можно познакомиться по ссылке (на английском языке) https://docs.scipy.org/doc/numpy/reference/.

Далее рассмотрим решение задач линейной алгебры на Python.

План изложения следующий: формулировка задачи, решение ее в Python, примеры задач для самостоятельного решения.

ТЕМА 2. ЭЛЕМЕНТЫ ЛИНЕЙНОЙ АЛГЕБРЫ. ПРИМЕРЫ РЕШЕНИЯ ЗАДАЧ

ОПЕРАЦИИ НАД МАТРИЦАМИ И ИХ СВОЙСТВА

Ввод и вывод матрицы (пакет Numpy)

1. Ввести матрицу $\underset{2\times 3}{A}=\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$

Решение

```
import numpy as np
A=np.array([[1,2,3],[4,5,6]])
print (A)
[[1 2 3]
[4 5 6]]
```

2. Ввести матрицу $\underset{2\times 3}{B}=\begin{pmatrix} 10 & 20 & 30 \\ 40 & 50 & 60 \end{pmatrix}$

Решение

```
B=np.array([[10,20,30],[40,50,60]])
print (B)
[[10 20 30]
[40 50 60]]
```

3. Найти сумму A+B, заданных в 1. и 2.

Решение

```
print (A+B)
[[11 22 33]
[44 55 66]]
```

4. Найти 5А, заданной в 1.

Решение

```
print (5*A)
[[ 5 10 15]
[20 25 30]]
```

5. Найти А-В, заданных в 1. и 2.

Решение

```
print (A–B)
[[ -9 -18 -27]
[-36 -45 -54]]
```

Найти В-А, заданных в 1. и 2.

Решение

```
print (B–A)
[[ 9 18 27]
[36 45 54]]
```

6. Ввести матрицу $\underset{3\times 4}{B}=\begin{pmatrix} 10 & 20 & 30 & 40 \\ 50 & 60 & 70 & 80 \\ 90 & 100 & 110 & 120 \end{pmatrix}$

Решение

```
matrix2=np.array([[10,20,30,40],[50,60,70,80],
[90,100,110,120]])
print(matrix2)
[[ 10 20 30 40]
[ 50 60 70 80]
[ 90 100 110 120]]
```

Умножение матриц

Для вычисления произведения двух матриц А и В применяется функция np.dot(A,B)

7. Умножить
$\underset{2\times 3}{A}=\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$ на $\underset{3\times 4}{B}=\begin{pmatrix} 10 & 20 & 30 & 40 \\ 50 & 60 & 70 & 80 \\ 90 & 100 & 110 & 120 \end{pmatrix}$

Решение

```
A=np.array([[1,2,3],[4,5,6]])
B=np.array([[10,20,30,40],
[50,60,70,80],[90,100,110,120]])
print (np.dot(A,B))
[[ 380 440 500 560]
[ 830 980 1130 1280]]
```

8. Проверить свойство некоммутативности умножения матриц

$$\underset{2\times 2}{A}=\begin{pmatrix}1 & 2\\ 3 & 4\end{pmatrix}, \underset{2\times 2}{B}=\begin{pmatrix}5 & 1\\ -1 & 0\end{pmatrix}$$

Решение

```
A=np.array([[1,2],[3,4]])
B=np.array([[5,1],[-1,0]])
print(np.dot(A,B))
print (np.dot(B,A))
[[ 3 1]
[11 3]]
[[ 8 14]
[-1 -2]]
```

Видим, что в общем случае умножение матриц не коммутативно, то есть A*B B*A.

Возведение матрицы в степень

Используем модуль linalg из библиотеки numpy, который позволяет осуществлять операции из линейной алгебры для матриц. Для этого импортируем его из библиотеки numpy, заменив, linalg на псевдоним ln.

9. Возвести $\underset{2\times 2}{A}=\begin{pmatrix}1 & 2\\ 3 & 4\end{pmatrix}$ **в 3 степень**

Решение

```
A=np.array([[1,2],[3,4]])
from numpy import linalg as ln
```

```
print (ln.matrix_power(A,3))
[[ 37 54]
[ 81 118]]
```

Транспонирование матрицы

A.transpose() — один из операторов для получения транспонированной матрицы.

10. Транспонировать матрицы $\underset{2\times 2}{A}=\begin{pmatrix}1 & 2\\ 3 & 4\end{pmatrix}$,

$$\underset{3\times 4}{B}=\begin{pmatrix}10 & 20 & 30 & 40\\ 50 & 60 & 70 & 80\\ 90 & 100 & 110 & 120\end{pmatrix}$$

Решение

```
A=np.array([[1,2,3],[4,5,6]])
B=np.array([[10,20,30,40],
[50,60,70,80],[90,100,110,120]])

print (A.transpose())
print (matrix2.transpose())
[[1 3]
[2 4]]
[[ 10 50 90]
[ 20 60 100]
[ 30 70 110]
[ 40 80 120]]
```

11. Вычислить

$$\left(A^T+2B\right)\cdot(2C-5D),\ A=\begin{pmatrix}1 & 0 & -1\\ 2 & 3 & -2\end{pmatrix},\ B=\begin{pmatrix}5 & 1\\ 3 & 2\\ 4 & -3\end{pmatrix},$$

$$C=\begin{pmatrix}2 & 0 & -1 & 3\\ 5 & 1 & 2 & 5\end{pmatrix},$$

$$D=\begin{pmatrix}7 & 1 & 1 & -3\\ 4 & 1 & -2 & 0\end{pmatrix}.$$

Решение

```
A=np.array([[1,0,-1],[2,3,-2]])
B=np.array([[5,1],[3,2],[4,-3]])
C=np.array([[2,0,-1,3],[5,1,2,5]])
D=np.array([[7,1,1,-3],[4,1,-2,0]])
Print(np.dot(A.transpose()+B,2*C-5*D))
[[-216 -39 0 156]
[-143 -30 49 113]
[ -43 0 -91 13]]
```

12. Вычислить $G = (A^T - 2B)\cdot(2C - 5D)$

$A=\begin{pmatrix} 1 & 0 & -1 \\ 2 & 3 & -2 \end{pmatrix}$, $B=\begin{pmatrix} 5 & 1 \\ 3 & 2 \\ 4 & -3 \end{pmatrix}$, $C=\begin{pmatrix} 2 & 0 & -1 & 3 \\ 5 & 1 & 2 & 5 \end{pmatrix}$,

$D=\begin{pmatrix} 7 & 1 & 1 & -3 \\ 4 & 1 & -2 & 0 \end{pmatrix}$.

Решение

```
import numpy as np
A=np.array([[1,0,-1],[2,3,-2]])
print (A)
[[ 1 0 -1]
[ 2 3 -2]]
B=np.array([[5,1],[3,2],[4,-3]])
print (B)
[[ 5 1]
[ 3 2]
[ 4 -3]]
C=np.array([[2,0,-1,3],[5,1,2,5]])
print (C)
[[ 2 0 -1 3]
[ 5 1 2 5]]
D=np.array([[7,1,1,-3],[4,1,-2,0]])
print (D)
[[ 7 1 1 -3]
```

```
[ 4 1 -2 0]]
print (np.dot(A.transpose()-2*B,2*C-5*D))
[[ 279 45 63 -189]
[ 196 33 28 -136]
[ 239 33 119 -149]]
```

13. Вычислить $(A-2B^T)\cdot(C+3D)$

$$A=\begin{pmatrix}-3 & 0\\ 2 & -1\\ 1 & 1\\ 0 & 3\end{pmatrix},\ B=\begin{pmatrix}3 & 2 & 1 & -4\\ 1 & 1 & -3 & 0\end{pmatrix},\ C=\begin{pmatrix}4 & 0 & -3\\ 2 & 5 & -1\end{pmatrix},$$

$$D=\begin{pmatrix}10 & 5 & -1\\ -2 & 3 & -3\end{pmatrix}.$$

Решение

```
import numpy as np
A=np.array([[-3,0],[2,-1],[1,1],[0,3]])
Print(A)
[[-3 0]
[ 2 -1]
[ 1 1]
[ 0 3]]
B=np.array([[3,2,1,-4],[1,1,-3,0]])
print (B)
[[ 3 2 1 -4]
[ 1 1 -3 0]]
C=np.array([[4,0,-3],[2,5,-1]])
print ©
[[ 4 0 -3]
[ 2 5 -1]]
D=np.array([[10,5,-1],[-2,3,-3]])
print (D)
[[10 5 -1]
```

```
[-2 3 -3]]
G=np.dot(A-2*B.transpose(),C+3*D)
print (G)
[[-298 -163 74]
[ -56 -72 42]
[ -62 83 -64]
[ 260 162 -78]]
```

14. Вычислить значение многочлена $F(x) = 2x^3 - x + 5$ **от матрицы** $A = \begin{pmatrix} -1 & 0 \\ 2 & 2 \end{pmatrix}$.

Решение

Здесь мы вводим функции, а не выполняем действия с матрицами для того, чтобы еще раз обратить внимание на удобство и универсальность подпрограмм в Python. В качестве параметра (аргумента) x можно поставить любую матрицу того же размера, и программа для решения задачи не изменится. Отметим, что eye(2) — это единичная матрица размера 2, а eye(n) — единичная матрица размера n.

```
def f(x):
return 2*ln.matrix_power(x,3)-x+5*
A=np.array([[-1,0],[2,2]])
Print( f(A))
[[ 4. 0.]
[ 10. 19.]]
```

15. Вычислите определители матриц

Для вычисления определителя матрицы A используем ln.det(A).

$$A = \begin{vmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 3 & 4 \\ 3 & 1 & 2 & 4 \\ 4 & 3 & 2 & 1 \end{vmatrix}$$

Решение

```
import numpy as np
from numpy import linalg as ln
A=np.array([[1,2,3,4],[2,1,3,4],[3,1,2,4],[4,3,2,1]])
Det_A=ln.det(A)
print (Det_A)
```

-30.0

И еще два примера.

16.

$$B=\begin{vmatrix} 4 & 6 & -2 & 4 \\ 1 & 2 & -3 & 1 \\ 4 & -2 & 1 & 0 \\ 6 & 4 & 4 & 6 \end{vmatrix}$$

Решение

```
B=np.array([[4,6,-2,4],[1,2,-3,1],[4,-2,1,0],[6,4,4,6]])
Det_B=ln.det(B)
print (Det_B)
```

-144.0

17.

$$C=\begin{vmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 3 & 4 \\ 3 & 1 & 2 & 4 \\ 4 & 3 & 2 & 1 \end{vmatrix}$$

Решение

```
C=np.array([[1,2,3,4],[2,1,3,4],[3,1,2,4],[4,3,2,1]])
Det_C=ln.det(C)
print (Det_C)
```

-30.0

18. Вычислить обратную матрицу

Для вычисления обратной матрицы A используем ln.inv(A)

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

Решение

```
A=np.array([[1,-1,1],[2,1,1],[1,1,2]])
A_inv=ln.inv(A)
Print(A_inv)
[[ 0.2 0.6 -0.4]
[-0.6 0.2 0.2]
[ 0.2 -0.4 0.6]]
```

19. Решить матричное уравнение $A \cdot X = B$

$A=\begin{pmatrix}3 & 1\\ -3 & 1\end{pmatrix}$, $B=\begin{pmatrix}9 & 5\\ -3 & -1\end{pmatrix}$

Используем def matrix_eq(A,B):

Поясним, что матрицу X мы находим, программируя произведение (np.dot) обратной к матрице A (ln.inv(A)) и матрицы B, то есть $X = A^{-1} \cdot B$.

Решение

```
A=np.array([[3,1],[-3,1]])
B=np.array([[9,5],[-3,-1]])
def matrix_eq(A,B):
return np.dot(ln.inv(A),B)
print (matrix_eq(A,B))
[[ 2. 1.]
[ 3. 2.]]
```

20. Найдите ранг матрицы

$$\begin{pmatrix} 0 & -1 & 3 & 0 & 2 \\ 2 & -4 & 1 & 5 & 3 \\ -4 & 5 & 7 & -10 & 0 \\ -2 & 1 & 8 & -5 & 3 \end{pmatrix}$$

Ранг матрицы находит функция ln.matrix_rank(A).

Решение

```
A=[[0,-1,3,0,2],[2,-4,1,5,3],[-4,5,7,-10,0],[-2,1,8,-5,3]]
A=np.array([[0,-1,3,0,2],[2,-4,1,5,3],
[-4,5,7,-10,0],[-2,1,8,-5,3]])
print (ln.matrix_rank(A))
2
```

ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

Вычислить *G*, если даны *A*, *B*, *C*, *D*.

21. $G=\left(A-2B^T\right)\cdot(C+3D)$, $A=\begin{pmatrix} 2 & -1 \\ 3 & 2 \\ 5 & -2 \end{pmatrix}$, $B=\begin{pmatrix} 3 & 2 & 1 & -4 \\ 1 & 1 & -3 & 0 \end{pmatrix}$, $C=\begin{pmatrix} 4 & 0 & -3 \\ 2 & 5 & -1 \end{pmatrix}$,

$D=\begin{pmatrix} 10 & 5 & -1 \\ -2 & 3 & -3 \end{pmatrix}$.

22. $G=(2A+B)^T\cdot(3C-4D)$, $A=\begin{pmatrix} 2 & -1 \\ 3 & 2 \\ 5 & -2 \end{pmatrix}$, $B=\begin{pmatrix} -2 & -1 \\ 11 & -2 \\ 5 & 7 \end{pmatrix}$, $C=\begin{pmatrix} 1 & 3 & 0 & 4 \\ 0 & 2 & -3 & 5 \\ 5 & 3 & 10 & -2 \end{pmatrix}$,

$D=\begin{pmatrix} -2 & 5 & 1 & -3 \\ 1 & 1 & -5 & -4 \\ 2 & 6 & 11 & -1 \end{pmatrix}$.

23. $G=(3A-B)\cdot(2C+3D)^T$, $A=\begin{pmatrix}0&1&1&-1\\1&1&-2&-5\\2&2&-1&-1\end{pmatrix}$, $B=\begin{pmatrix}-1&2&1&-1\\2&4&-3&-2\\1&1&10&-1\end{pmatrix}$, $C=$

$$\begin{pmatrix}4&2&1&-5\\3&1&-2&-1\\5&4&0&-4\end{pmatrix},\ D=\begin{pmatrix}-2&1&3&-3\\1&1&-1&-2\\1&2&-3&-1\end{pmatrix}.$$

24. $G=(2A+3B)\cdot\left(C-2D^T\right)$,

$$A=\begin{pmatrix}2&1&-1\\0&3&2\\1&4&-3\\0&5&2\end{pmatrix},\ B=\begin{pmatrix}0&1&-2\\1&1&1\\3&-2&2\\2&0&4\end{pmatrix},\ C=\begin{pmatrix}3&-2\\1&4\\2&-3\end{pmatrix},\ D=\begin{pmatrix}-3&2&-2\\-4&0&-1\end{pmatrix}.$$

25. $G=2\cdot(A-B)\cdot(3C+2D)^T$,

$$A=\begin{pmatrix}-2&1\\0&-3\\-1&4\\2&-1\end{pmatrix},\ B=\begin{pmatrix}3&-2\\4&-1\\1&4\\5&-3\end{pmatrix},\ C=\begin{pmatrix}0&-1\\3&-4\\5&-1\end{pmatrix},\ D=\begin{pmatrix}-3&4\\1&2\\-2&3\end{pmatrix}.$$

26. $G=3(A+B)^T\cdot(2C-D)$, $A=\begin{pmatrix}-2&3&-4\\-3&5&-2\end{pmatrix}$, $B=\begin{pmatrix}0&-3&-1\\2&0&-4\end{pmatrix}$,

$$C=\begin{pmatrix}0&4&1&-3\\2&1&-1&5\end{pmatrix},\ D=\begin{pmatrix}3&0&2&-2\\0&3&-3&0\end{pmatrix}.$$

27. $G=5\left(A^T-2B\right)\cdot(C+2D)$, $A=\begin{pmatrix}0&1&-2\\3&2&1\\2&0&-2\\1&4&-1\end{pmatrix}$, $B=\begin{pmatrix}2&1&2&-2\\0&3&-1&-1\\4&1&0&-3\end{pmatrix}$, $C=\begin{pmatrix}-3&4\\-2&1\\-1&4\\-2&3\end{pmatrix}$, D

$$=\begin{pmatrix}2&-2\\3&-4\\4&-3\\5&-4\end{pmatrix}$$

Вычислить значение многочлена $f(x)$ от матрицы A.

28. $f(x)=2x^3+x+5$, $A=\begin{pmatrix}-1 & 0\\ 2 & 2\end{pmatrix}$.

29. $f(x)=x^3+2x-4$, $A=\begin{pmatrix}3 & 1\\ 0 & 1\end{pmatrix}$.

30. $f(x)=x^3+x^2+x+1$, $A=\begin{pmatrix}2 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 3\end{pmatrix}$.

31. $f(x)=2x^2+x-3$, $A=\begin{pmatrix}-1 & 0 & 0\\ -2 & 1 & 0\\ -3 & 0 & 1\end{pmatrix}$.

32. $f(x)=x^3-x^2-1$, $A=\begin{pmatrix}-2 & 1\\ 1 & 0\end{pmatrix}$.

33. $f(x)=x^2-x^3+1-x$, $A=\begin{pmatrix}0 & 0 & -1\\ 0 & 1 & 0\\ 1 & 0 & 0\end{pmatrix}$.

34. $f(x)=5x^2-x-5$, $A=\begin{pmatrix}1 & 0 & 0\\ 2 & 1 & 0\\ 3 & 2 & 1\end{pmatrix}$.

35. $f(x)=3x^4-x^2+3$, $A=\begin{pmatrix}-5 & -1 & 4 & 2 & -9 & 11\\ -1 & 5 & 4 & -6 & 8 & -6\\ 4 & 4 & -4 & 2 & -7 & -3\\ 2 & -6 & 2 & -8 & 5 & -2\\ -9 & 8 & -7 & 5 & -6 & 7\\ 11 & -6 & -3 & -2 & 7 & -8\end{pmatrix}$

Вычислить определитель.

36. $\begin{vmatrix} 3 & 5 & 7 & 2 \\ 7 & 6 & 3 & 7 \\ 5 & 4 & 3 & 5 \\ -5 & -6 & -5 & -4 \end{vmatrix}$.

37. $\begin{vmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{vmatrix}$.

38. $\begin{vmatrix} 2 & 4 & 8 & 0 \\ 4 & 8 & 0 & 27 \\ 8 & 0 & 27 & 9 \\ 0 & 27 & 9 & 3 \end{vmatrix}$.

39. $\begin{vmatrix} -3 & 2 & 4 & 0 \\ -2 & 3 & 6 & 9 \\ -4 & 0 & 3 & 9 \\ -6 & 2 & 3 & 0 \end{vmatrix}$.

40. $\begin{vmatrix} 2 & -3 & 4 & 3 \\ 4 & -2 & 3 & 2 \\ 5 & -5 & 2 & 3 \\ 3 & -4 & 4 & 3 \end{vmatrix}$.

41. $\begin{vmatrix} 1 & 1 & 1 & 1 \\ 4 & 3 & 2 & 1 \\ 16 & 9 & 4 & 1 \\ 64 & 27 & 8 & 1 \end{vmatrix}$.

42. $\begin{vmatrix} 0 & 3 & 5 & 7 \\ -3 & 0 & 3 & 5 \\ -5 & -3 & 0 & 3 \\ -7 & -5 & -3 & 0 \end{vmatrix}$

Найти матрицу обратную к данной.

$$44.\ A=\begin{pmatrix} -1 & -4 & 8 & 15 & -1 & 5 \\ -8 & 4 & -5 & 7 & -5 & 8 \\ 7 & 3 & -14 & 7 & 8 & -2 \\ 10 & -9 & 0 & -4 & -2 & -6 \\ -6 & -3 & -4 & -2 & 12 & 2 \\ -7 & 8 & 12 & -4 & -9 & 9 \end{pmatrix}$$

$$45.\ B=\begin{pmatrix} 3 & -5 & 9 & 3 & -1 & 0 \\ 5 & -3 & -5 & 0 & -5 & 13 \\ -6 & 10 & -4 & 15 & -2 & 0 \\ 15 & -6 & 9 & -9 & -2 & -6 \\ 8 & -3 & 5 & -5 & 8 & -1 \\ 0 & 8 & 6 & -4 & 12 & -7 \end{pmatrix}$$

$$46.\ A=\begin{pmatrix} -5 & -1 & 4 & 2 & -9 & 11 \\ -1 & 5 & 4 & -6 & 8 & -6 \\ 4 & 4 & -4 & 2 & -7 & -3 \\ 2 & -6 & 2 & -8 & 5 & -2 \\ -9 & 8 & -7 & 5 & -6 & 7 \\ 11 & -6 & -3 & -2 & 7 & -8 \end{pmatrix}$$

$$47.\ A=\begin{pmatrix} -1 & 0 & 8 & 6 & -1 & 5 \\ 8 & 4 & -5 & 4 & -5 & 11 \\ 3 & 6 & -14 & 7 & 8 & -2 \\ 10 & 4 & 0 & -4 & -2 & -6 \\ -6 & -3 & -4 & -2 & 2 & 13 \\ -7 & 8 & 12 & -4 & 9 & -5 \end{pmatrix}$$

$$48.\ B=\begin{pmatrix} 8 & -5 & 12 & -5 & -1 & 0 \\ 5 & -3 & 8 & 13 & -5 & 13 \\ -6 & 13 & -4 & 15 & -3 & 3 \\ 15 & -11 & 9 & -9 & -2 & -6 \\ 8 & -3 & 9 & -5 & 8 & -1 \\ 0 & 8 & 6 & -4 & 12 & 7 \end{pmatrix}$$