

## Лабораторная работа №5: "Двумерные массивы и функции"

### Цель работы

Дать студентам практический навык в написании программ обработки двумерных массивов с использованием функций.

### Постановка задачи

Дан двумерный массив вещественных чисел.

1. Инициировать элементы случайными числами в диапазоне (-10 -: 10).
2. Вычислить среднее и дисперсию ( $D$ ) по всем элементам массива.
3. Заменить, на среднее значение, те элементы массива, отклонение которых от среднего превышает  $\sigma$ , где  $\sigma = \sqrt{D}$ .
4. Пункты задания оформить в виде функций.

### Теоретическое введение

#### Организация двумерных массивов

В Python реализовать массив можно через вложенные списки. В следующем листинге программы показан прием формирования двумерного списка и способы инициализации его элементов. В этой программе следует обратить внимание на то, что элемент списка, ранее инициированный числом, получает значение строкового типа и преобразование делается по умолчанию.

#### Листинг программы

```
from random import *
n=int(input("Введите размер матрицы (NxN): "))
A=[] # Пустой список
B=[]
for i in range(n):
    A.append([0]*n) # Вложенный список
                    # инициированный 0
    B.append([3]*n) # Вложенный список
                    # инициированный 3
print(A) # Вывод списка
print(B)
print() # Просто "Enter"
for Row in range(n): # По строкам и
    for Col in range(n): # по столбцам
        A[Row][Col]=randint(0,99) # псевдослучайное число
for Row in range(n): # Вывод результата
    for Col in range(n):
        print("{0:02}".format(A[Row][Col]), end=" ")
    print()
print()
#
A[1][2] = 'AB' # A это "сюрприз"
for i in range(n): # Вывод результата
    for j in range(n):
        if ((i==1) and (j==2)):
            print(A[1][2], end=" ")
        else:
            print("{0:02}".format(A[i][j]), end=" ")
```

```
print()
```

### Результат работы программы

```
Введите размер матрицы (NxN): 4
[[0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
[[3, 3, 3, 3], [3, 3, 3, 3], [3, 3, 3, 3], [3, 3, 3, 3]]
```

```
58 75 20 06
17 99 45 72
26 49 73 46
74 20 35 03
```

```
58 75 20 06
17 99 AB 72
26 49 73 46
74 20 35 03
```

Отметим следующие моменты, касающиеся такой реализации кода:

- элементом списка может быть объект любого типа, это снижает эффективность кода;
- в случае обработки массивов (совокупность однотипных элементов) следует использовать более эффективный код.

В научных вычислениях массивы используются часто. В качестве примеров можно рассмотреть и временные ряды метеонаблюдений, и матрицы, используемые при решении систем уравнений.

Как уже отмечалось, Python является объектно-ориентированным языком. В этом языке массивы – это объекты со своими атрибутами и методами.

Так, для получения размерности массива можно воспользоваться атрибутом `ndim`:

```
dim = Mymas.ndim
```

Количество элементов по каждой из координат возвращает атрибут `shape`:

```
a, b = Mymas.shape
```

Атрибут `size` возвращает количество элементов в массиве:

```
n = Mymas.size
```

Функция `len()` позволяет получить количество элементов в строке:

```
m = len(Mymas)
```

С целью повышения эффективности кода и реализации дополнительных методов и функций, применяемых в обработке массивов, в Python разработаны различные модули. Например, большие возможности для работы с многомерными массивами предоставляет модуль **NumPy (numpy)**, который и предлагается использовать для решения задач нашего пособия.

Следующий листинг программы демонстрирует способы генерации матриц и вывода матриц на экран с использованием модуля **numpy**.

### Листинг программы

```
import numpy as np

n=int(input("Введите размер матрицы (NxN): "))
# Нулевая матрица
Z=np.zeros((n, n), int) # Row, Colum, type
print(Z)
print()
# Диагональная единичная матрица
```

```

E=np.eye(n)
print(E)
print()
# Матрица со случайным набором значений
A = 20*np.random.random(size=(n,n)) - 10
for Row in range(n):
    for Col in range(n):
        print("{0:>5.2f}".format(A[Row][Col]), end=" ")
    print()
print()
# Инициализация матрицы
for Row in range(n):
    for Col in range(n):
        A[Row][Col]=Row*10+Col
for Row in range(n):
    for Col in range(n):
        print("{0:>05.2f}".format(A[Row][Col]), end=" ")
    print()

```

### Результат работы программы

Введите размер матрицы (NxN): 4

```

[[0 0 0 0]
 [0 0 0 0]
 [0 0 0 0]
 [0 0 0 0]]

```

```

[[ 1.  0.  0.  0.]
 [ 0.  1.  0.  0.]
 [ 0.  0.  1.  0.]
 [ 0.  0.  0.  1.]]

```

```

-0.09 -1.68  5.82  0.47
-5.72 -5.65 -5.96 -2.20
 6.20  1.72 -7.72 -5.56
 8.60  0.18  8.39  4.11

```

```

00.00 01.00 02.00 03.00
10.00 11.00 12.00 13.00
20.00 21.00 22.00 23.00
30.00 31.00 32.00 33.00

```

В решении заданий следует автоматизировать процесс формирования значений для двумерных массивов, а не вводить их руками. Не следует использовать готовые методы и функции, которые скрывают алгоритм решения, например, такие, как поиск максимума или минимума.

Для решения нашего задания инициализацию элементов матрицы выполним функцией, которая генерирует псевдослучайные числа вещественного типа из состава модуля NumPy.

Некоторые функции модуля NumPy, формируют псевдослучайное число в диапазоне  $[0, 1)$  (включая ноль и исключая 1), например, `random()`, `ranf()`, `sample()`. В случае, когда генерацию псевдослучайных чисел необходимо выполнить в диапазоне  $[a, b)$ , где  $b > a$ , следует использовать следующее выражение:

$$y = a + (b - a) * \text{Rand},$$

где Rand – псевдослучайное число, генерируемое одной из перечисленных выше функций. Пример представлен в программе, приведенной выше.

Для расчёта среднего, дисперсии и среднеквадратичного отклонения воспользуемся следующими формулами:

$$\text{Среднее: } A = \frac{\sum_{i=1}^n \sum_{j=1}^n a_{ij}}{n^2}; \quad \text{Дисперсия: } D = \frac{\sum_{i=1}^n \sum_{j=1}^n (a_{ij} - A)^2}{n^2 - 1}; \quad \sigma = \sqrt{D}.$$

Инициализацию матрицы, вычисление среднего и дисперсии, а так же корректировку матрицы оформим в виде функций.

### Функции

В программе часто повторяющийся фрагмент кода может быть оформлен в виде отдельной именованной сруктуры. В терминах информатики такую структуру принято называть подпрограммой.

Существуют два вида подпрограмм – это процедуры и функции. Функции, в отличие от процедур, могут участвовать в выражениях. Функция возвращает результат через собственное имя. Кроме этого часть результата может быть возвращена через один из параметров.

В некоторых языках это разделение существует явно, а в некоторых – неявно. Так, например, в языке Паскаль при описании подпрограмм используются слова `procedure` и `function`, в Basic – Sub (от subroutine) и `function`.

Пример из Паскаля:

```
readln(x, y);  
y := 2*y+sin(x);
```

В этом примере `readln()` – процедура, а `sin(x)` – функция, с аргументом `x`.

В языке Си и Python такое разделение сделано неявно. В этих языках функция может не возвращать или возвращать значение.

В языке Си подпрограммам предшествует описатель типа. Для не возвращающих значение (подпрограмм), используется тип `void`. Для возвращающих значение (функций) – тип, который соответствует типу возвращаемого значения, например, `int`, `float` или `bool`.

В Python используется только понятие "функция".

В тех случаях, когда используется инструкция `return`, функция возвращает объект через свое имя, который указан за инструкцией `return`. Такая функция может участвовать в правой части выражений. Если инструкция `return` не используется, то функция возвращает значение `None`, которое так же может быть проанализировано в условном выражении.

**ЗАМЕЧАНИЕ:** Переменные, используемые при описании функции, называют параметрами, а переменные, которые используются при вызове процедуры – аргументами.

Существует несколько стилей оформления программы с подпрограммами. В одних из них описание подпрограмм предшествует описанию самой программы, в других – описание подпрограмм следует за описание программы (перед программой помещается только заголовок подпрограммы), а в третьих – подпрограммы помещаются в отдельные модули, которые подключаются к программе с помощью специальных инструкций: `uses` (Паскаль), `include` (Си), `import` (Python). Во всех стилях принимаются меры к тому, чтобы к моменту вызова подпрограммы в программе, ее имя (имя подпрограммы) было известно.

С этой целью в Python, рекомендуется все функции описывать в голове программы. Описание функции выполняется по следующей схеме:

```
def <Имя_функции> ([Параметры]):
    '''Строка документирования'''
    <Тело_функции>
    [return <Возвращаемый_результат>]
```

Тут <Имя\_функции> - это имя, которое будет использовано при вызове функции в программе. После имени функции, в круглых скобках, могут присутствовать параметры. По своей сути, описание параметров – это описание переменных, которые используются в теле функции для выполнения различных инструкций с их участием. С другой стороны, параметры – это переменные, через которые функция получает данные из программы или возвращает новые данные в программу.

Имя функции должно быть уникальным и соответствовать требованиям к именованию переменных.

Инструкция `return` необязательна, но может быть использована для возвращения данных из функции через имя этой функции.

Опишем алгоритм решения нашей задачи.

#### Описание алгоритма

1. Запросить размер массива  $N$ . Поскольку форма массива неопределена, решим задачу для квадратной матрицы ( $N \times N$ ).
2. Изготовить массив – инициировать набором псевдослучайных вещественных чисел (функция `MakeMatr()`).
3. Вывести полученную матрицу (функция `PrintMatr()`).
4. Вычислить среднее значение и дисперсию (функция `MidlDisp()`).
5. Выполнить корректировку элементов (функция `CorrectMatr()`).
6. Вывести откорректированную матрицу

#### Описание входных и выходных данных

Программа запрашивает размер массива. Результат работы программы визуализируется на экране монитора. Тип элементов матрицы задан как вещественный (**float**) в соответствии с заданием.

#### Описание подпрограмм

##### **Функция *MakeMatr(n, a, b)***

Функция инициализирует квадратную матрицу размером  $n \times n$  псевдослучайными числами в диапазоне  $[a, b)$ .

Возвращается объект типа `array`.

##### **Функция *MidlDisp( Matr)***

Функция вычисляет среднее значение по элементам массива *Matr* и дисперсию.

Функция возвращает объект типа кортеж, в котором первый элемент – среднее значение, а второй - дисперсия.

##### **Функция *CorrectMatr( Matr)***

Функция выполняет корректировку массива *Matr*, заменяя значения элементов, средним, если модуль значения в массиве превышает среднеквадратичное отклонение.

Функция возвращает значение типа `array`.

##### **Функция *PrintMatr( Matr)***

Служит для вывода массива на экран монитора.

#### Листинг программы

```
import numpy as np
from math import *
```

```

def MakeMatr(n, a, b):
    """ Инициализация квадратной матрицы
    размером NxN псевдослучайными величинами
    в диапазоне [a,b) """
    Matr = (b-a)*np.random.random(size=(n,n)) + a
    return Matr

def MidlDisp(Matr):
    """ Вычисление среднего значения
    Вычисление дисперсии """
    sum = 0
    (nRow, nCol) = Matr.shape # Размеры матрицы
    for Row in range(nRow):
        for Col in range(nCol):
            sum += Matr[Row][Col]
    Midl = sum / Matr.size # Среднее
    Disp = 0
    for Row in range(nRow):
        for Col in range(nCol):
            Disp += (Matr[Row][Col] - Midl)**2
    return (Midl, Disp / (Matr.size - 1)) # Кортеж

def CorrectMatr(Matr, aver, sigma):
    """ Замена "отскочивших" значений """
    (nRow, nCol) = Matr.shape # Размеры матрицы
    for Row in range(nRow):
        for Col in range(nCol):
            if abs(abs(Matr[Row][Col]) - aver) > sigma:
                Matr[Row][Col] = aver
    return (Matr)

def PrintMatr(Matr):
    """ Печать матрицы """
    (nRow, nCol) = Matr.shape
    for Row in range(nRow):
        for Col in range(nCol):
            print("{0: 7.3f}".format(Matr[Row][Col]), end=" ")
        print()
    print()

n=int(input("Введите размер матрицы (NxN): "))
MyMatr=MakeMatr(n, -5, 5) # Изготовить матрицу
PrintMatr(MyMatr)

(mMidl, mDisp) = MidlDisp(MyMatr) # Среднее и дисперсия
print("Midl = {0:7.3f}".format(mMidl))
print("Disp = {0:7.3f} Sig = {1:7.3f}"\
      .format(mDisp, sqrt(mDisp)))

NMatr = CorrectMatr(MyMatr, mMidl, sqrt(mDisp)) # Корректировка
PrintMatr(NMatr)

```

## Результат работы программы

Введите размер матрицы (NхN) : 7

-3.775	-3.782	-1.127	4.921	-3.295	1.567	-0.084
-1.091	-4.529	3.789	-4.245	3.748	4.199	-0.593
-4.977	4.779	1.184	-0.448	-1.605	3.030	-4.608
0.076	-1.509	-3.713	-3.791	-4.139	4.570	3.234
2.997	-3.789	1.384	-2.880	-2.572	-2.550	-3.275
-2.468	-1.418	-0.306	2.019	4.532	2.518	1.570
-2.188	4.122	-0.271	2.883	-3.256	4.610	-1.648

Midl = -0.249

Disp = 9.902 Sig = 3.147

-0.249	-0.249	-1.127	-0.249	-0.249	1.567	-0.084
-1.091	-0.249	-0.249	-0.249	-0.249	-0.249	-0.593
-0.249	-0.249	1.184	-0.448	-1.605	-0.249	-0.249
0.076	-1.509	-0.249	-0.249	-0.249	-0.249	-0.249
-0.249	-0.249	1.384	-2.880	-2.572	-2.550	-0.249
-2.468	-1.418	-0.306	2.019	-0.249	2.518	1.570
-2.188	-0.249	-0.271	2.883	-0.249	-0.249	-1.648

## Задание к лабораторной работе №5 "Двумерные массивы и функции"

Размерности двумерных массивов следует запрашивать у пользователя. Все необходимые данные должны передаваться в функции в качестве параметров; Все переменные, используемые только внутри функции, должны быть описаны как локальные.

Использование глобальных переменных в функциях не допускается. Обеспечить вывод, как исходного массива, так и массива, полученного в результате работы программы, там, где это возможно по условию задачи.

Пункты задания оформить в виде функций.

### Вариант 1

Дана целочисленная прямоугольная матрица. Определить:

1. Количество строк, не содержащих ни одного нулевого элемента.
2. Максимальное значение из чисел, встречающихся в заданной матрице более одного раза.

### Вариант 2

Дана целочисленная прямоугольная матрица.

1. Определить количество столбцов, не содержащих ни одного нулевого элемента.
2. Переставляя строки заданной матрицы, расположить их в соответствии с ростом характеристик.

*ПРИМЕЧАНИЕ:* Характеристикой строки целочисленной матрицы назовем сумму ее положительных четных элементов.

### Вариант 3

Дана целочисленная прямоугольная матрица. Определить:

1. Количество столбцов, содержащих хотя бы один нулевой элемент.
2. Номер строки, в которой находится самая длинная серия одинаковых элементов.

### Вариант 4

Дана целочисленная квадратная матрица. Определить:

1. Произведение элементов в тех строках, которые не содержат отрицательных элементов.
2. Максимум среди сумм элементов диагоналей, параллельных главной диагонали матрицы.

### Вариант 5

Дана целочисленная квадратная матрица. Определить:

1. Сумму элементов в тех столбцах, которые не содержат отрицательных элементов.
2. Минимум среди сумм модулей элементов диагоналей, параллельных побочной диагонали матрицы.

### Вариант 6

Дана целочисленная прямоугольная матрица. Определить:

1. Сумму элементов в тех строках, которые содержат хотя бы один отрицательный элемент.
2. Номера строк и столбцов всех седловых точек матрицы.

*ПРИМЕЧАНИЕ:* Матрица  $A$  имеет седловую точку  $A_{ij}$ , если  $A_{ij}$  является минимальным элементом в  $i$ -й строке и максимальным в  $j$ -м столбце.

### Вариант 7

Для заданной матрицы размером  $8 \times 8$  найти такие  $k$ , что элементы  $k$ -й строки матрицы совпадают с элементами  $k$ -ого столбца.

Найти сумму элементов в тех строках, которые содержат хотя бы один отрицательный элемент.



### Вариант 8

Переставляя столбцы заданной матрицы, расположить их в соответствии с ростом характеристик.

Найти сумму элементов в тех столбцах, которые содержат хотя бы один отрицательный элемент.

**ПРИМЕЧАНИЕ:** Характеристикой столбца целочисленной матрицы назовем сумму модулей его отрицательных нечетных элементов.

### Вариант 9

Соседями элемента  $A_{ij}$  в матрице назовем элементы  $A_{kl}$ , где  $i-1 \leq k \leq i+1$ ,  $j-1 \leq l \leq j+1$ ,  $(k, l) \neq (i, j)$ .

$$\begin{pmatrix} a_{1,1} & \mathbf{a_{1,2}} & \dots & a_{1,j-1} & a_{1,j} & a_{1,j+1} & \dots \\ \mathbf{a_{2,1}} & \mathbf{a_{2,2}} & \dots & a_{2,j-1} & a_{2,j} & a_{2,j+1} & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ a_{i-1,1} & a_{i-1,2} & \dots & \mathbf{a_{i-1,j-1}} & \mathbf{a_{i-1,j}} & \mathbf{a_{i-1,j+1}} & \dots \\ a_{i,1} & a_{i,2} & \dots & \mathbf{a_{i,j-1}} & \mathbf{a_{i,j}} & \mathbf{a_{i,j+1}} & \dots \\ a_{i+1,1} & a_{i+1,2} & \dots & \mathbf{a_{i+1,j-1}} & \mathbf{a_{i+1,j}} & \mathbf{a_{i+1,j+1}} & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{pmatrix}$$

В этой матрице соседи углового элемента  $a_{11}$  и элемента  $a_{ij}$  выделены полужирным шрифтом.

Операция сглаживания матрицы дает новую матрицу того же размера, каждый элемент которой получается как среднее арифметическое имеющихся соседей соответствующего элемента исходной матрицы.

Построить результат сглаживания заданной вещественной матрицы размером  $10 \times 10$ .

В сглаженной матрице найти сумму модулей элементов, расположенных ниже главной диагонали.

### Вариант 10

Элемент матрицы называется локальным минимумом, если он строго меньше всех имеющихся у него соседей.

Соседями элемента  $A_{ij}$  в матрице назовем элементы  $A_{kl}$ , где  $i-1 \leq k \leq i+1$ ,  $j-1 \leq l \leq j+1$ ,  $(k, l) \neq (i, j)$ .

$$\begin{pmatrix} \mathbf{a_{1,1}} & \mathbf{a_{1,2}} & \dots & a_{1,j-1} & a_{1,j} & a_{1,j+1} & \dots \\ \mathbf{a_{2,1}} & \mathbf{a_{2,2}} & \dots & a_{2,j-1} & a_{2,j} & a_{2,j+1} & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ a_{i-1,1} & a_{i-1,2} & \dots & \mathbf{a_{i-1,j-1}} & \mathbf{a_{i-1,j}} & \mathbf{a_{i-1,j+1}} & \dots \\ a_{i,1} & a_{i,2} & \dots & \mathbf{a_{i,j-1}} & \mathbf{a_{i,j}} & \mathbf{a_{i,j+1}} & \dots \\ a_{i+1,1} & a_{i+1,2} & \dots & \mathbf{a_{i+1,j-1}} & \mathbf{a_{i+1,j}} & \mathbf{a_{i+1,j+1}} & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{pmatrix}$$

В этой матрице соседи углового элемента  $a_{11}$  и элемента  $a_{ij}$  выделены полужирным шрифтом.

Подсчитать количество локальных минимумов заданной матрицы размером  $10 \times 10$ .

Найти сумму модулей элементов, расположенных выше главной диагонали.

### Вариант 11

Коэффициенты системы линейных уравнений заданы в виде прямоугольной матрицы. С помощью допустимых преобразований привести систему к треугольному виду.

Найти количество строк, среднее арифметическое элементов которых меньше заданной величины.

### Вариант 12

Уплотнить заданную матрицу, удаляя из нее строки и столбцы, заполненные нулями.

Найти номер первой из строк, содержащих хотя бы один положительный элемент.

### Вариант 13

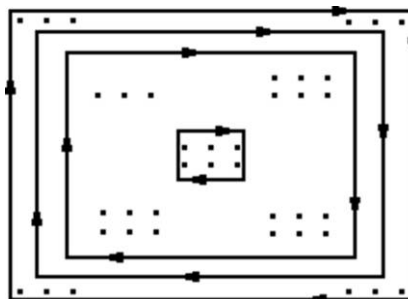
Осуществить циклический сдвиг элементов прямоугольной матрицы на  $n$  элементов вправо или вниз (в зависимости от введенного режима),  $n$  может быть больше количества элементов в строке или столбце.

**ПРИМЕЧАНИЕ:** Под циклическим сдвигом элементов понимается перестановка элементов строки или столбца по следующему правилу:

$1, 2, 3, \dots, n$	$\rightarrow$	$n, 1, 2, 3, \dots, n-1$	$\rightarrow$	$n-1, n, 1, \dots, n-2$	$\rightarrow$	$n-2, n-1, n, 1, 2, \dots, n-3$
Исходное		Циклический		на 2 элем.		на 3 элем.
состояние		сдвиг вправо на				
		1 элем.				

### Вариант 14

Осуществить циклический сдвиг элементов матрицы размером  $M \times N$  ( $M$  строк  $\times$   $N$  столбцов). Сдвиг выполнить вправо на  $k$  элементов таким образом: элементы первой строки сдвигаются в последний столбец сверху вниз, из него - в последнюю строку справа налево, из нее - в первый столбец снизу вверх, из него - в первую строку; для остальных элементов - аналогично.



### Вариант 15

Дана целочисленная прямоугольная матрица. Определить номер первого из столбцов, содержащих хотя бы один нулевой элемент.

Переставляя строки заданной матрицы, расположить их в соответствии с убыванием характеристик.

**ПРИМЕЧАНИЕ:** Характеристикой строки целочисленной матрицы назовем сумму ее отрицательных четных элементов.

### Вариант 16

Упорядочить строки целочисленной прямоугольной матрицы по возрастанию количества одинаковых элементов в каждой строке.

Найти номер первого из столбцов, не содержащих ни одного отрицательного элемента.

### Вариант 17

Путем перестановки элементов квадратной вещественной матрицы добиться того, чтобы ее максимальный элемент находился в левом верхнем углу (1,1), следующий по величине - в позиции (2, 2), следующий по величине - в позиции (3, 3) и т. д., заполнив, таким образом, всю главную диагональ.

Найти номер первой из строк, не содержащих ни одного положительного элемента.

### Вариант 18

Дана целочисленная прямоугольная матрица. Определить:

1. Количество строк, содержащих хотя бы один нулевой элемент.
2. Номер столбца, в котором находится самая длинная серия одинаковых элементов.

### Вариант 19

Дана целочисленная квадратная матрица. Определить:

1. Сумму элементов в тех строках, которые не содержат отрицательных элементов.
2. Минимум среди сумм элементов диагоналей, параллельных главной диагонали матрицы.

### Вариант 20

Дана целочисленная прямоугольная матрица. Определить:

1. Количество отрицательных элементов в тех строках, которые содержат хотя бы один нулевой элемент.
2. Номера строк и столбцов всех седловых точек матрицы.

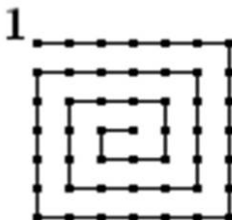
*ПРИМЕЧАНИЕ:*

Матрица  $A$  имеет седловую точку  $A_{ij}$ , если  $A_{ij}$  является минимальным элементом в  $i$ -й строке и максимальным в  $j$ -м столбце.

### Вариант 21

Напишите программу, формирующую квадратную матрицу, элементы которой являются натуральными числами, расположенными в порядке возрастания от 1 до  $n^2$  ( $n$  - порядок матрицы) согласно схеме, приведённой на рисунке.

Вычислить сумму элементов, расположенных на главной диагонали полученной матрицы.

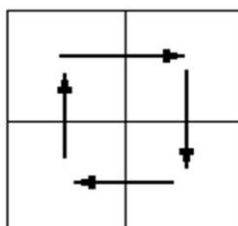


### Вариант 22

Написать программу, которая меняет местами столбцы квадратной матрицы, содержащие наибольший и наименьший элементы и вычисляет сумму элементов главной диагонали.

### Вариант 23

Квадратная матрица порядка  $2n$  состоит из 4-х блоков. Написать программу, которая формирует новую матрицу, переставляя блоки исходной матрицы согласно схеме, и печатает сумму элементов каждого блока.



#### Вариант 24

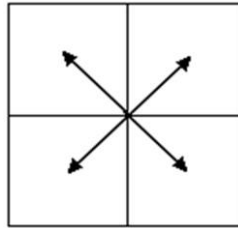
Имеется квадратная матрица с целочисленными элементами. Написать программу, которая столбцы заданной матрицы делает строками новой матрицы.

Для каждого значения элемента матрицы подсчитать количество элементов, которые принимают такое же значение. Подсчёт проводить лишь для тех значений, которые представлены в матрице.

#### Вариант 25

Квадратная матрица порядка  $2 \times n$  состоит из 4-х блоков. Написать программу, которая:

- формирует новую матрицу, переставляя блоки исходной матрицы согласно схеме;
- печатает произведение элементов каждого блока.



#### Вариант 26

Написать программу, которая заполняет матрицу размерности  $4 \times 13$  числами от 1 до 52 случайным образом: повторение чисел не допускается. Полученный результат вывести на экран монитора.

#### Вариант 27

Даны две матрицы одного порядка  $M \times N$  ( $M$  строк  $\times$   $N$  столбцов). Написать программу сложения, вычитания и транспонирования матриц.

1. Сложение и вычитание:  $c_{ij} = a_{ij} \pm b_{ij}$
2. Транспонирование  $b_{ij} = a_{ji}$

#### Вариант 28

Произведением двух матриц  $A_{m \times n}$  на  $B_{n \times l}$  называется такая матрица  $C_{m \times l}$ , для которой:

$$c_{ik} = a_{i1} \cdot b_{1k} + a_{i2} \cdot b_{2k} + \dots + a_{in} \cdot b_{nk} = \sum_{j=1}^n a_{ij} \cdot b_{jk}.$$

Т.е. элемент  $c_{ik}$  матрицы  $C$  равен сумме произведений элементов  $i$ -й строки матрицы  $A$  на соответствующие элементы  $k$ -го столбца матрицы  $B$ .

Написать программу вычисления произведения двух матриц.

Программа должна по заданным размерностям матриц сообщать о возможности получения такого произведения.

### Вариант 29

Экспонента от (квадратной) матрицы  $A$  может быть определена следующим образом:

$$e^{\mathbf{A}} = \mathbf{I} + \mathbf{A} + \frac{1}{2!}\mathbf{A}^2 + \frac{1}{3!}\mathbf{A}^3 + \dots,$$

где  $\mathbf{I}$  - единичная матрица, у которой элементы главной диагонали равны единице, а остальные - нулю:  $i_{mn} = \begin{cases} 1 & m = n \\ 0 & m \neq n \end{cases}$ ,  $\mathbf{A}^2 = \mathbf{A} \times \mathbf{A}$ .

Произведением двух матриц  $A_{mn}$  на  $B_{nl}$  называется такая матрица  $C_{ml}$ , для которой:

$$c_{ik} = a_{i1} \cdot b_{1k} + a_{i2} \cdot b_{2k} + \dots + a_{in} \cdot b_{nk} = \sum_{j=1}^n a_{ij} \cdot b_{jk}.$$

Т.е. элемент  $c_{ik}$  матрицы  $C$  равен сумме произведений элементов  $i$ -й строки матрицы  $A$  на соответствующие элементы  $k$ -го столбца матрицы  $B$ .

Написать программу для вычисления матричной суммы первых  $n$  элементов ряда.

Примеры представления единичной матрицы и матрицы размерностью  $2 \times 2$  и вычисления квадрата матрицы:

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}, \quad A^2 = \begin{pmatrix} 1 \cdot 1 + 2 \cdot 3 & 1 \cdot 2 + 2 \cdot 4 \\ 3 \cdot 1 + 4 \cdot 3 & 3 \cdot 2 + 4 \cdot 4 \end{pmatrix} = \begin{pmatrix} 7 & 10 \\ 15 & 22 \end{pmatrix}$$

### Вариант 30

Имеется набор символов 0..9 и A..Z (всего 36 символов). Написать программу, которая:

- случайным образом заполняет элементы двухмерной матрицы размерности  $6 \times 6$ . Повторение символов не допускается.

- выводит предложение, которое содержит не более 10 слов, составленных по следующему принципу:

- случайным образом задаётся число, которое является номером столбца или строки;
- элементы строки (столбца) формируют слово.