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 σ, 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.

Обратите внимание на строку, выделенную жирным. Элемент массива, ранее получивший числовое значение инициируется строковым значением. При этом Python выполняет эту операцию без выдачи предупреждения (элементы списка могут быть разного типа). Здесь это сделано только для примера.

Листинг программы

```
from random import *
n=int(input("Введите размер кв. матрицы (N): "))
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) # случайное
                                  # 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'              # А 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]]
[[3, 3, 3], [3, 3, 3], [3, 3, 3]]

    58 75 20
    17 99 45
    26 49 73

    58 75 20
    17 99 AB
    26 49 73
```

Отметим следующие моменты, касающиеся такой реализации кода:

- элементом списка может быть объект любого типа, это снижает эффективность кода;
- в случае обработки массивов (набор 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 функии, которые скрывают алгоритм решения, например, такие, как поиск максимума или минимума.

Для решения нашего задания, см. постановку задачи выше, 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()`. Когда генерацию псевдослучайных чисел необходимо выполнить в диапазоне `[a, b)`, где `b > a`, следует использовать следующее выражение:

```
y= a + (b - a)*Rand,
```

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

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

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

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 структуры. В терминах информатики такую структуру принято называть подпрограммой.

Выделяют два вида подпрограмм – это процедуры и функции. Процедуры и функции могут возвращать результат через параметры. Кроме этого функция всегда возвращает результат через собственное имя и, в отличие от процедур, может участвовать в выражениях.

В Python используется только понятие "функция". Объект функция создаётся инструкцией `def`, за которой следует имя функции и, при необходимости, параметры, которые указываются в круглых скобках. Заголовок функции завершается двоеточием. Тело функции образуется блоком инструкций, которые следуют далее и имеют отступ в четыре пробела, по умолчанию. При отсутствии инструкций в теле функции, например в процессе тестирования, записывают инструкцию `pass`, которая ничего не делает. Выход из функции выполняется после выполнения последней инструкции тела функции. Такой инструкцией может быть, но необязательно, и инструкция `return`. В этом случае функция, через свое имя, возвращает объект, 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.

Функции могут быть вложенными в другие функции, но делать это не рекомендуется, см., например, М. Лутц, Изучаем Python, 2011.

Замечание: Переменные, используемые при описании функции, обычно 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.

Замечание о последовательности описания функций в Python:

При вызове функции (`f2()`) из другой функции (`f1()`) сама функция `f2()` может быть описана после `f1()`. Здесь важно только то, что вызов функции `f1()` выполнен после описания `f2()`. Это вызвано тем обстоятельством, что в Python вначале формируется байт код (это интерпретатор) и только затем работает виртуальная машина Python. Таким образом, к моменту выполнения кода (вызов `f1()`) размещение функции `f2()` известно и код может быть обработан без проблем.

Пример:

```
def f1(y):
    f2(y)  # Опережающая ссылка

def f2(x): # Вызываемая функция
    print(x)

f1(56)     # Вызов функции
```

Описание алгоритма

Рассмотрим алгоритм решения нашей задачи.

1. Запросить размер массива N. Поскольку форма массива не определена, решим задачу для квадратной матрицы (NxN).
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 псевдослучайными числами в диапазоне `[a,b)`. Размер матрицы `n*n` элементов.

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)

Эта функция выводит массив 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, avr, 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]) - avr) > sigma:
                Matr[Row][Col] = avr
    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))))
# Корректировка
NMatr = CorrectMatr(MyMatr, mMidl, sqrt(mDisp))
PrintMatr(NMatr)
```

Результат работы программы

```
Введите размер матрицы (NxN): 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 x 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 **A** назовем такие элементы a_{kl}, для которых выполняются условия: $i-1<=k<=i+1$, $j-1<=m<=j+1$, $(k,m)\neq(i,j)$.

Это все элементы, которые окружают центральный элемент по горизонтали, вертикали и диагоналям. Для крайних левых элементов соседей слева нет, так же и для правых крайних – соседей справа нет.

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

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

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 10x10.

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. Понятие соседей дано в варианте 9.

Подсчитать количество локальных минимумов заданной матрицы размером 10 x 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

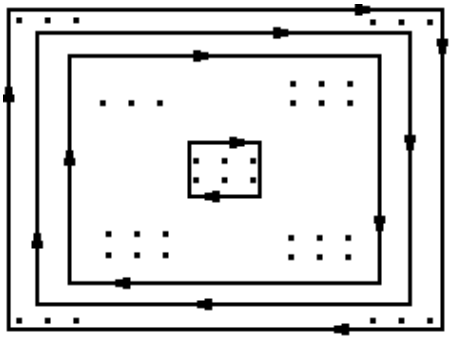
Осуществить циклический сдвиг элементов прямоугольной матрицы на n элементов вправо или вниз (в зависимости от введенного режима), n может быть больше количества элементов в строке или столбце.

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

1, 2, 3, …,n Исходное состояние	-> n, 1, 2, 3, …,n-1 Циклический сдвиг вправо на 1 элем.	-> n-1, n, 1, …,n-2 на 2 элем.	-> n-2, n-1, n, 1, 2, …,n-3 на 3 элем.

Вариант 13

Осуществить циклический сдвиг элементов матрицы размером m * n (m строк x 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 14

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

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

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 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 20

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

Вариант 21

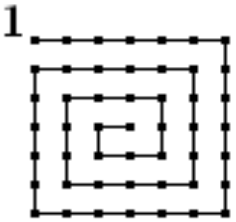
Дана целочисленная квадратная матрица. Определить:

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

Вариант 22

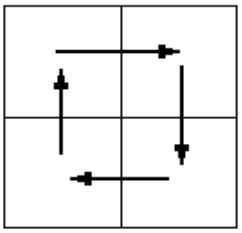
Напишите программу, формирующую квадратную матрицу, элементы которой являются натуральными числами, расположенными в порядке возрастания от 1 до n^2 (n – порядок матрицы) согласно схеме, приведённой на рисунке.

Вычислить сумму элементов, расположенных на главной диагонали полученной матрицы.



Вариант 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

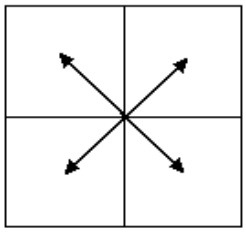
Даны две матрицы одного порядка m * n (m строк x n столбцов). Написать программу сложения, вычитания и транспонирования матриц.

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

Вариант 26

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

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



Вариант 27

Написать программу, которая заполняет матрицу размерности 4x13 числами от 1 до 52 случайным образом: повторение чисел не допускается. Полученный результат вывести на экран монитора.

Вариант 28

Произведением двух матриц **A** на **B** называется такая матрица **C**, для которой: $c_{ik} = a_{i1} \cdot b_{1k} + a_{i2} \cdot b_{2k} + ...a_{in} \cdot b_{nk} = \sum_{j=1}^{n} a_{ij} \cdot b_{jk}$.

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:

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

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

Сумма матриц **A** и **B** одинакового размера есть матрица **C** того же размера с элементами $c_{ik} = a_{ik} + b_{ik}$ при всех i и k. Таким образом, сложение матриц одинакового размера выполняется поэлементно.

Произведение матрицы на действительное число λ будет матрица, у которой все элементы умножены на λ: $\mathbf{C} = \lambda \cdot \mathbf{A} = |\lambda \cdot a_{ik}|$.

Произведением двух матриц **A** на **B** называется такая матрица **C**, для которой: $c_{ik} = a_{i1} \cdot b_{1k} + a_{i2} \cdot b_{2k} + ...a_{in} \cdot b_{nk} = \sum_{j=1}^{n} a_{ij} \cdot b_{jk}$.

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 вычисляющую экспоненту от матрицы с суммированием первых n элементов ряда.

Пример представления единичной матрицы, матрицы размерностью 2x2 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:

a) случайным образом заполняет элементы двухмерной матрицы размерности 6x6. Повторение символов не допускается;

b) 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) формирую слово.