

Practice work 8 Working with two-dimensional arrays.

Objective: the study two-dimensional arrays in Python.

know- a way to describe a two-dimensional array, ways to enter elements of a two-dimensional array;

be able to- enter arrays, get lists by assigning specific values, apply functions;

own- basic skills in creating programs for processing two-dimensional arrays.

Matrices are called arrays of elements, presented in the form of rectangular tables, for which the rules of mathematical operations are defined. Matrix elements can be numbers, algebraic symbols, or mathematical functions.

Python also uses lists to work with matrices. Each element of the matrix list contains a nested list.

Thus, a structure of nested lists is obtained, the number of which determines the number of columns of the matrix, and the number of elements inside each nested list indicates the number of rows in the original matrix.

1. Create a list

Let two numbers be given: the number of rows n and the number of columns m . It is necessary to create a list of size $n \times m$ filled with zeros.

The obvious solution turns out to be wrong:

$A = [[0] * m] * n$

It is easy to see this if you assign the value 1 to the element $A[0][0]$ and then display the value of another element $A[1][0]$ — it will also be equal to 1! The point is that $[0] * m$ returns a reference to a list of m zeros. But subsequent iteration of that element creates a list of n elements that are references to the same list (just like doing $B = A$ on lists doesn't create a new list), so all rows of the resulting list are actually the same and the same line.

Thus, a two-dimensional list cannot be created using a single-line repetition operation.

First way.

First, let's create a list of n elements (just n zeros to start with). Then we make each element of the list a reference to another one-dimensional list of m elements:

```
A = [0] * n
for i in
range(n): A[i] =
[0] * m
```

```
n=3
m=3
A=[0]*n
for i in range(n):
    A[i]=[0]*m
print('A:',A)
```

```
A: [[0, 0, 0], [0, 0, 0], [0, 0, 0]]
>>> |
```

The second way.

Create an empty list, then add a new element to it n times, which is a string-list:

```
A=[]
for i in range(n):
A.append([0] * m)
```

```
n=3
m=4
A = []
for i in range(n):
    A.append([0]*m)
print(A)
```

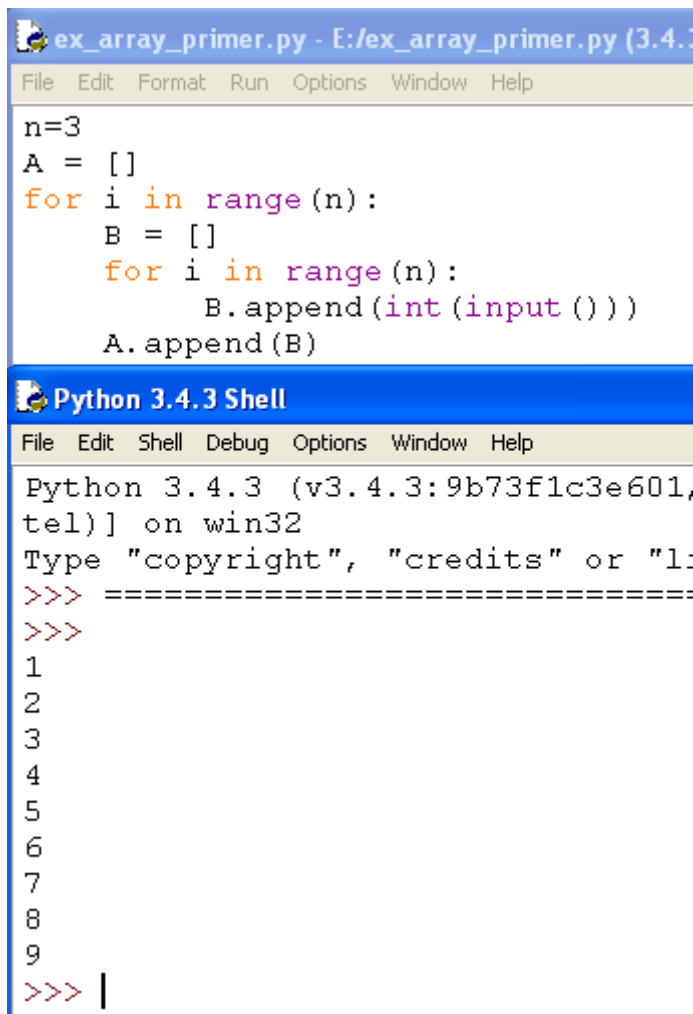
```
A: [[0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
>>> |
```

2. Nested list input (two-dimensional array)

Example:

```
n
=5 A =
[]
    for i in
range(n): b =
input()
        for i in
range(len(row)): row[i] =
int(row[i])
```

```
A.append(row)
```



The screenshot shows two windows from a Python IDE. The top window, titled 'ex_array_primer.py - E:/ex_array_primer.py (3.4.3)', contains the following Python code:

```
n=3
A = []
for i in range(n):
    B = []
    for i in range(n):
        B.append(int(input()))
    A.append(B)
```

The bottom window, titled 'Python 3.4.3 Shell', shows the execution of the script. It displays the Python version and architecture, followed by a prompt to enter input. The user has entered the numbers 1 through 9, which are displayed line by line. The prompt '>>> |' is shown at the bottom, indicating the shell is waiting for more input.

3. Output nested list (two-dimensional array)

As a rule, two nested loops are used to process and display the list. The first loop is on the line number, the second loop is on the elements inside the line. For example, to display a two-dimensional numeric list on the screen line by line, separating the numbers with spaces within one line, you can do this:

```
for i in
range(n): for j in
range(n):
    print(A[i][j], end = ' ')
print()
```

```

n=3
A = []
#Ввод массива
for i in range(n):
    B = []
    for i in range(n):
        B.append(int(input()))
    A.append(B)
#Вывод массива
for i in range(n):
    for j in range(n):
        print(A[i][j], end=' ')
    print()

```

```

1
2
3
4
5
6
7
8
9
1 2 3
4 5 6
7 8 9

```

The same, but loops not by index, but by list values:

```

for row in A:
    for elem in row:
        print(elem, end = ' ')
    print()

```

```
A=[[1,2,3,4],[5,6,7,8]]
for row in A:
    for elem in row:
        print(elem, end = ' ')
    print()

Python 3.7.1 Shell
File Edit Shell Debug Options Window Help
Python 3.7.1 (v3.7.1:260ec2c36a,
1)] on win32
Type "help", "copyright", "cred:
>>>
===== RESTART: K
1 2 3 4
5 6 7 8
>>> |
```

To output a single line, you can use the join method.
Using this method in a for loop, you can for row

in A:

```
print(' '.join(list(map(str, row))))
```

4. Handling and Displaying Nested Lists

Often in tasks it is necessary to store rectangular tables with data. Such tables are called matrices or two-dimensional arrays. In the Python programming language, a table can be represented as a list of strings, each element of which is in turn a list of, for example, numbers. For example, you can create a numerical table with two rows and three columns like this:

```
A = [ [1, 2, 3], [4, 5, 6] ]
```

Here the first line of the list A[0] is a list of numbers [1, 2, 3]. That is

```
A[0][0]= 1,
```

```
A[0][1]= 2,
```

```
A[0][2]= 3,
```

```
A[1][0]=4,
```

```
A[1][1]=5,
```

```
A[1][2]=6.
```

We use two nested loops to calculate the sum of all the numbers in the list:

```
S=0
for i in range(len(A)):
    for j in
range(len(A[i])): S +=
A[i][j]
```

Or the same with a loop not by index, but by row values:

```
S=0
for row in A:
    for
element in row:
        S +=
element
```

```
A=[[1, 2, 3,4],[ 5, 6,7,8]]
#вывод при помощи цикла for и метода join
print('Массив A:')
for i in A:
    print(' '.join(list(map(str, i))))
#Пример 1. Подсчёт суммы всех элементов
s = 0
for i in range(len(A)):
    for j in range(len(A[i])):
        s += A[i][j]
print('Пример 1. Сумма элементов:', s)
#Пример 2. Подсчёт суммы всех элементов
s = 0
for row in A:
    for elem in row:
        s += elem
print('Пример 2. Сумма элементов:', s)
```

```
Массив A:
1 2 3 4
5 6 7 8
Пример 1. Сумма элементов: 36
Пример 2. Сумма элементов: 36
```

5. An example of complex array processing

Let a square matrix with n rows and n columns be given. It is necessary to assign the value 0 to the elements located on the main diagonal passing from the upper left corner to the lower right (that is, those elements $A[i][j]$ for which $i=j$) to assign the value 0, to the elements located above the main diagonal - the value 1, elements below the main diagonal - the value 2. That is, get an array like this (example for $n=3$):

```
0 1 1
2 0 1
2 2 0
```

Let's consider several ways to solve this problem.

First way.

The elements that lie above the main diagonal are the elements of $A[i][j]$, for which $i < j$, and for elements below the main diagonal $i > j$. Thus, we can compare the values of i and j and use them to determine the value of $A[i][j]$. We get the following algorithm:

```
for i in
range(n):
    for j in
range(n):
        if i < j:
            A[i][j] = 0
        elif i > j:
            A[i][j] = 2
        else:
            A[i][j] = 1
```

The following is an example program in which a 3x3 square matrix is filled with elements with a value of 9, and then elements located on the main diagonal passing from the upper left corner to the lower right (that is, those elements $A[i][j]$ for which $i == j$) are assigned the value 0, elements located above the main diagonal - the value 1, elements , located below the main diagonal - the value 2.

```

n=3
A=[]
#заполняем массив 9-ми
for i in range(n):
    A.append([9]*n)
#вывод исходного массива
for i in range(n):
    for j in range(n):
        print(A[i][j], end = ' ')
    print()
#заменяем элементы главной диагонали, выше и ниже неё
for i in range(n):
    for j in range(n):
        if i < j:
            A[i][j] = 1
        elif i > j:
            A[i][j] = 2
        else:
            A[i][j] = 0
#вывод изменённого массива
print()
for i in range(n):
    for j in range(n):
        print(A[i][j], end = ' ')
    print()

```

```

9 9 9
9 9 9
9 9 9

0 1 1
2 0 1
2 2 0
>>> |

```

The second way.

This algorithm is bad because it executes one or two if statements to process each element. If we complicate the algorithm, then we can do without conditional instructions at all.

First we fill in the main diagonal, for which we need one cycle:

```

for i in
range(n): A[i][i]
= 1

```

Then we fill with the value 0 all the elements above the main diagonal, for which we need to assign in each of the rows with the number i

value to elements $A[i][j]$ for $j=i+1, \dots, n-1$. Here we need nested loops:

```
for i in range(n):
    for j in range(i +
1, n): A[i][j] = 0
```

Similarly, we assign a value to 2 elements of $A[i][j]$ for $j=0, \dots, i-1$: for i in $\text{range}(n)$:

```
    for j in
range(0, i): A[i][j] =
2
```

You can also combine the outer loops into one and get another, more compact solution:

```
for i in range(n):
    for j in
range(0, i): A[i][j] =
2
    A[i][i] = 1
    for j in range(i +
1, n): A[i][j] = 0
```

```
n=3
A=[]
#заполняем массив 9-ми
for i in range(n):
    A.append([9]*n)
#вывод исходного массива
for i in range(n):
    for j in range(n):
        print(A[i][j], end = ' ')
    print()
#заменяем элементы главной диагонали, выше и ниже неё
for i in range(n):
    for j in range(0, i):
        A[i][j] = 2
    A[i][i] = 0
    for j in range(i + 1, n):
        A[i][j] = 1
#вывод изменённого массива
print()
for i in range(n):
    for j in range(n):
        print(A[i][j], end = ' ')
    print()
```

```

9 9 9
9 9 9
9 9 9

0 1 1
2 0 1
2 2 0
>>> |

```

The third way.

But this solution uses the operation of repeating lists to build the next line of the list. The i -th line of the list consists of i numbers 2, followed by a single number 1, followed by $n-i-1$ number 0:

```

for i in range(n):
    A[i] = [2] * i + [1] + [0] * (n - i - 1)

```

```

n=3
A=[]
#заполняем массив 9-ми
for i in range(n):
    A.append([9]*n)
#вывод исходного массива
for i in range(n):
    for j in range(n):
        print(A[i][j], end = ' ')
    print()
#заменяем элементы главной диагонали, выше и ниже неё
for i in range(n):
    A[i] = [2] * i + [0] + [1] * (n - i - 1)
#вывод изменённого массива
print()
for i in range(n):
    for j in range(n):
        print(A[i][j], end = ' ')
    print()

```

```

9 9 9
9 9 9
9 9 9

0 1 1
2 0 1
2 2 0
>>> |

```

Task 0

1. Given a two-dimensional array of size 3x3. Determine the maximum value among the elements of the third column of the array; the maximum value among the elements of the second row of the array. Output the received values.

Solution:

```
n=3
a=[]
for i in range(n):
    b = []
    for j in range(n):
        print('Введите [',i,',',j,','] элемент')
        b.append(int(input()))
    a.append(b)
#вывод массива
for i in range(n):
    for j in range(n):
        print(a[i][j], end=' ')
    print()

#максимальное значение среди элементов третьего столбца
maximum=a[0][2]
for i in range(n):
    for j in range(n):
        if maximum<a[i][2]:
            maximum=a[i][2]
print('Максимальный в 3 столбце:',maximum)

#максимальное значение среди элементов второй строки
maximum=a[1][0]
for i in range(n):
    for j in range(n):
        if maximum<a[1][j]:
            maximum=a[1][j]
print('Максимальный во второй строке:',maximum)
```

```

Введите [ 0 , 0 ] элемент
1
Введите [ 0 , 1 ] элемент
2
Введите [ 0 , 2 ] элемент
3
Введите [ 1 , 0 ] элемент
4
Введите [ 1 , 1 ] элемент
5
Введите [ 1 , 2 ] элемент
6
Введите [ 2 , 0 ] элемент
7
Введите [ 2 , 1 ] элемент
8
Введите [ 2 , 2 ] элемент
9
1 2 3
4 5 6
7 8 9
Максимальный в 3 столбце: 9
Максимальный во второй строке: 6

```

2. Given a two-dimensional array of size $m \times n$. Form a new array by replacing positive elements with ones and negative elements with zeros. Output both arrays.

Solution:

```
m=int(input('Введите количество строк'))
n=int(input('Введите количество столбцов'))
a=[]
for i in range(m):
    b = []
    for j in range(n):
        print('Введите [' ,i ,',' ,j ,'] элемент')
        b.append(int(input()))
    a.append(b)
#вывод массива
print('Исходный массив:')
for i in range(m):
    for j in range(n):
        print(a[i][j], end=' ')
    print()

for i in range(m):
    for j in range(n):
        if a[i][j]<0: a[i][j]=0
        elif a[i][j]>0: a[i][j]=1

#вывод массива
print('Изменённый массив:')
for i in range(m):
    for j in range(n):
        print(a[i][j], end=' ')
    print()
```

```

Введите количество строк:3
Введите количество столбцов:4
Введите [ 0 , 0 ] элемент
-1
Введите [ 0 , 1 ] элемент
5
Введите [ 0 , 2 ] элемент
4
Введите [ 0 , 3 ] элемент
-5
Введите [ 1 , 0 ] элемент
-2
Введите [ 1 , 1 ] элемент
-1
Введите [ 1 , 2 ] элемент
0
Введите [ 1 , 3 ] элемент
4
Введите [ 2 , 0 ] элемент
-5
Введите [ 2 , 1 ] элемент
4
Введите [ 2 , 2 ] элемент
5
Введите [ 2 , 3 ] элемент
-5
Исходный массив:
-1 5 4 -5
-2 -1 0 4
-5 4 5 -5
Полученный массив:
0 1 1 0
0 0 0 1
0 1 1 0

```

Task 1.

1. Calculate the sum and the number of positive elements of the matrix $A[N, N]$ located above the main diagonal.
2. Given a matrix $B[N, M]$. Find the maximum and minimum elements in each row of the matrix and swap them with the first and last elements of the row, respectively.

Task 2.

1. An integer square matrix of order n is given. Determine if it is a magic square, that is, a matrix in which the sums of elements in all rows and columns are the same.
2. Given a rectangular matrix $A[N, N]$. Swap first and

the last columns are swapped and displayed.

Task 3.

1. Determine if the given integer square matrix of the n th order is symmetric (with respect to the main diagonal).
2. Given a real matrix of size $n \times m$. By rearranging its rows and columns, ensure that the largest element (or one of them) is in the upper left corner.

Task 4.

1. Given a rectangular matrix. Find the row with the largest and the row with the smallest sum of elements. Print the found strings and the sums of their elements.
2. Given a square matrix $A[N, N]$, Write zeros instead of negative elements of the matrix, and ones instead of positive ones. Print the lower triangular matrix in the conventional form.

Task 5.

1. Sort in ascending order the elements of each row of an $n \times m$ matrix.
2. Given a real matrix of size $n \times m$, all elements of which are different. In each row, the element with the smallest value is selected. If the number is even, then it is replaced by zero, odd - by one. Display the new matrix.

Task 6.

1. You are given an integer square matrix. Find the largest element in each row and the smallest element in each column. Display on screen.
2. Given a real square matrix of order N (N is odd), all elements of which are different. Find the largest element among those on the main and secondary diagonals and swap it with the element on the intersection of these diagonals.

Task 7.

1. A square matrix, symmetric about the main diagonal, is given by the upper triangle as a one-dimensional array. Restore the original matrix and print row by row.

2. For a given square matrix, form a one-dimensional array of its diagonal elements. Find the trace of a matrix by summing the elements of a one-dimensional array. Transform the original matrix according to the rule: divide even rows by the resulting value, leave odd rows unchanged.

Task 8.

1. Given a matrix of order n and a number k . Divide the elements of the k th row by the diagonal element located in this row.

2. Given a square matrix. Get the transposed matrix (inverted with respect to the main diagonal) and display it on the screen.

Task 9.

1. For an integer square matrix, find the number of elements that are multiples of k and the largest of these elements.

2. In a given real square matrix of order n , find the element with the greatest modulus. Get a square matrix of order $n - 1$ by discarding from the original matrix the row and column at the intersection of which the element with the found value is located.

Task 10.

1. Find the maximum among all elements of those rows of a given matrix that are ordered (either ascending or descending).

2. Arrange the columns of the matrix $D[M, N]$ in ascending order of the elements of the k th row ($1 \leq k \leq M$).

Task 11.

1. Given a real square matrix of order n , find the sum of the elements of the row containing the element with the smallest value. It is assumed that there is only one such element.

2. Among the columns of a given integer matrix containing only such elements that modulo is not greater than 10, find the column with the minimum product of the elements and swap with the neighboring one.

Task 12.

1. For a given square matrix, find k such that the k th row matrix coincides with the k th column.

1. Given a real matrix of size $n \times m$. It is required to transform the matrix: element by element subtract the last row from all rows except the last one.

Task 13.

1. Determine the smallest element of each even row of the matrix $A[M, N]$.
2. Find the largest and smallest elements of a rectangular matrix and swap them.

Task 14.

1. Given a square matrix. Swap the row with the maximum element on the main diagonal with the row with the given number m .
2. Write a program that fills a square matrix of order n with natural numbers $1, 2, 3, \dots, n^2$, writing them into it "in a spiral".

For example, for $n = 5$ we get the following matrix: 1 2 3 4 5

16 17 18 19 6

15 24 25 20 7

14 23 22 21 8

14 12 11 10 9

Task 15.

1. Determine the numbers of rows of the matrix $R[M, N]$, at least one element of which is equal to c , and multiply the elements of these rows by d .
2. Among those rows of an integer matrix that contain only odd elements, find the row with the maximum sum of the modules of the elements.