

## Листок 14: Двумерные списки (матрицы)

Задачи на 5: A-Q, S, U-Y.

Чтение входных данных во всех задачах — из файла `input.txt`.

Вывод результатов работы программы — в файл `output.txt`.

Пример создания прямоугольного массива ( $N$  строк,  $M$  столбцов), содержащего нули:

```
x = [[0 for i in range(M)] for j in range(N)]
```

Чтение прямоугольного массива, содержащего  $N$  строк (и неважно сколько столбцов):

```
x = []
```

```
for i in range(N):
```

```
    x.append(list(map(int, input().split())))
```

Вывод прямоугольного массива, содержащего  $N$  строк (и неважно сколько столбцов):

```
for i in range(N):
```

```
    print(' '.join(map(str, x[i])))
```

В задачах, в которых требуется заполнить массив, предполагается сначала создать двумерный массив указанного размера, заполненный нулями, затем заполнить в соответствии с условием задачи, затем вывести.

В задачах A-E, G-I количество действий должно быть пропорционально количеству единиц в таблице. В частности, запрещаются решения, обходящие всю таблицу и расставляющие единицы в нужные клетки и/или использующие оператор `if`.

Во всех задачах листочка запрещается использование вспомогательных двумерных массивов.

### Задачи

A. *Заполнение массива 0-1: ступеньки*

Дано натуральное число  $N \leq 20$ .

Напишите программу, которая создаёт таблицу размера  $N \times N$ , заполненную нулями и расставляет в данной таблице единицы в соответствии с примером (первый шаг из левого верхнего угла вправо на одну клетку, далее по три, если возможно).

В решении запрещается использовать условный оператор.

Input	Output
7	1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 1 1

B. *Заполнение массива 0-1: побочная диагональ и соседние*

Дано натуральное число  $N \leq 20$ .

Напишите программу, которая создаёт таблицу размера  $N \times N$ , заполненную нулями и расставляет в данной таблице единицы в соответствии с примером: побочная диагональ и соседние с ней (сверху и снизу) диагонали, если они есть заполняются единицами, остальные нулями.

В решении запрещается использовать условный оператор.

Input	Output
4	0 0 1 1 0 1 1 1 1 1 1 0 1 1 0 0

C. *Заполнение массива 0-1: уголки*

Дано натуральное число  $N \leq 20$ .

Напишите программу, которая создаёт таблицу размера  $N \times N$ , заполненную нулями и расставляет в данной таблице единицы в соответствии с примером: чередуются уголки с из нулей и единиц — сначала уголок, образованный крайним левым столбцом и верхней строкой, затем уголки через один.

В решении запрещается использовать условный оператор.

Input	Output
5	1 1 1 1 1 1 0 0 0 0 1 0 1 1 1 1 0 1 0 0 1 0 1 0 1

D. *Заполнение массива 0-1: концентрические квадраты*

Дано натуральное число  $N \leq 20$ .

Напишите программу, которая создаёт таблицу размера  $N \times N$ , заполненную нулями и расставляет в данной таблице единицы в соответствии с примером: концентрические квадраты, начиная с внешнего.

В решении запрещается использовать условный оператор.

Input	Output
5	1 1 1 1 1 1 0 0 0 1 1 0 1 0 1 1 0 0 0 1 1 1 1 1 1

E. *Заполнение массива 0-1: шахматный порядок*

Дано натуральное число  $N \leq 20$ .

Напишите программу, которая создаёт таблицу размера  $N \times N$ , заполненную нулями и расставляет в данной таблице единицы в шахматном порядке, причём в левом нижнем углу должна стоять единица.

В решении запрещается использовать условный оператор.

Input	Output
5	1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1

F\* *Заполнение массива 0-1: спираль*

Дано натуральное число  $N \leq 20$ .

Напишите программу, которая в заполненной нулями квадратной матрице  $N \times N$  проводит спираль из единиц. Спираль начинается в левом верхнем углу и закручивается по часовой стрелке. Каждая единица, кроме начальной и конечной, граничит по стороне ровно с двумя единицами.

Input	Output
5	1 1 1 1 1 0 0 0 0 1 1 1 1 0 1 1 0 0 0 1 1 1 1 1 1

■ В задачах G-I заполнить массив  $N \times M$  ( $N$  строк,  $M$  столбцов) в соответствии с условиями. Обходить весь массив нельзя, использовать `if` нельзя. Использовать `max`, `min`, `abs` — можно.

G. *Заполнение массива 0-1: сумма индексов*

Для данных  $N, M$  и  $k$  ( $0 \leq k \leq N + M - 2$ ) заполнить единицами те и только те клетки  $[i, j]$  таблицы  $N \times M$ , для которых  $i + j = k$ .

В первой строке вводится натуральное число  $N$ , во второй натуральное число  $M$ , в третьей — натуральное число  $k$ .

Программа должна вывести в выходной файл массив размера  $N \times M$ , заполненный указанным образом.

Input	Output
3	0 0 0 1
4	0 0 1 0
3	0 1 0 0

Н. *Заполнение массива 0-1: разность индексов*

Для данных  $N, M$  и  $k$  ( $1 - N \leq k \leq N - 1$ ) заполнить единицами те и только те клетки  $[i, j]$  таблицы  $N \times M$ , для которых  $i - j = k$ .

В первой строке вводится натуральное число  $N$ , во второй натуральное число  $M$ , в третьей — целое число  $k$ .

Программа должна вывести в выходной файл массив размера  $N \times M$ , заполненный указанным образом.

Input	Output
5	0 0 0
3	1 0 0
1	0 1 0
	0 0 1
	0 0 0

И. *Заполнение массива 0-1: модуль разности индексов*

Для данных  $N, M$  и  $k$  ( $0 \leq k \leq N - 1$ ) заполнить единицами те и только те клетки  $[i, j]$  таблицы  $N \times M$ , для которых  $|i - j| = k$ .

В первой строке вводится натуральное число  $N$ , во второй натуральное число  $M$ , в третьей — натуральное число  $k$ .

Программа должна вывести в выходной файл массив размера  $N \times M$ , заполненный указанным образом.

Input	Output
4	0 0 1 0 0 0
6	0 0 0 1 0 0
2	1 0 0 0 1 0
	0 1 0 0 0 1

Ж. *Заполнение массива: по диагоналям*

Для данных  $N$  и  $M$  заполнить матрицу неотрицательными числами по диагоналям.

В первой строке вводится натуральное число  $N$ , во второй натуральное число  $M$ .

Программа должна вывести в выходной файл массив размера  $N \times M$ , заполненный указанным образом.

Input	Output
4	0 1 2 3 4
5	1 2 3 4 5
	2 3 4 5 6
	3 4 5 6 7

К. *Заполнение массива: от диагонали*

Для данных  $N$  и  $M$  заполнить матрицу неотрицательными числами по диагоналям.

В первой строке вводится натуральное число  $N$ , во второй натуральное число  $M$ .

Программа должна вывести в выходной файл массив размера  $N \times M$ , заполненный указанным образом.

Input	Output
4	0 1 2 3 4
5	1 0 1 2 3
	2 1 0 1 2
	3 2 1 0 1

Л. *Заполнение массива: змейкой (1)*

Для данных  $N$  и  $M$  заполнить матрицу натуральными числами змейкой по диагонали, как показано в примере.

В первой строке вводится натуральное число  $N$ , во второй натуральное число  $M$ .

Программа должна вывести в выходной файл массив размера  $N \times M$ , заполненный указанным образом.

Input	Output
4	1 3 4 10 11
5	2 5 9 12 17
	6 8 13 16 18
	7 14 15 19 20

М. *Заполнение массива: по спирали*

Для данных  $N$  и  $M$  заполнить матрицу натуральными числами по спирали, как показано в примере. Спираль закручивается по часовой стрелке.

В первой строке вводится натуральное число  $N$ , во второй натуральное число  $M$ .

Программа должна вывести в выходной файл массив размера  $N \times M$ , заполненный указанным образом.

Input	Output
4	1 2 3 4 5
5	14 15 16 17 6
	13 20 19 18 7
	12 11 10 9 8

Н. *Заполнение массива: шахматный порядок*

Заполните массив размером  $N \times M$  в шахматном порядке: клетки одного цвета заполнены нулями, а другого цвета — заполнены числами натурального ряда сверху вниз, слева направо. В левом верхнем углу записано число 1.

В первой строке вводится натуральное число  $N$ , во второй натуральное число  $M$ .

Программа должна вывести в выходной файл массив размера  $N \times M$ , заполненный указанным образом, отводя на вывод каждого числа ровно 4 символа.

*Примечание:* выводить одно целое число с нужным количеством пробелов можно, например, так:

```
print('{0:4d}'.format(a))
```

Подробнее про форматированный вывод можно посмотреть в справочнике (стр. 8, параграф 2.3).

Кроме того, может пригодиться метод `rjust` (там же, стр. 39, параграф 8.2.3)

Input	Output
3	1 0 2 0 3
5	0 4 0 5 0
	6 0 7 0 8

О. *Строка с минимальной суммой*

Напишите программу, которая находит в матрице строку с минимальной суммой.

В первой строке записаны через пробел размеры матрицы: количество строк  $N$  и количество столбцов  $M$  ( $1 \leq N, M \leq 100$ ). В следующих  $N$  строках записаны строки матрицы, в каждой — по  $M$  натуральных чисел, разделённых пробелами.

Программа должна вывести все элементы найденной строки с минимальной суммой, разделив их пробелами.

Input	Output
4 5	13 26 56 9 12
1 3 2 54 234	
75 12 3 46 9	
13 26 56 9 12	
14 90 897 6 34	

■ В задачах P-R необходимо сначала прочитать двумерный массив из файла, потом *изменить* его, после чего вывести полученный результат. Вспомогательные массивы использовать нельзя.

Р. *Отражение матрицы по горизонтали*

Напишите программу, которая выполняет зеркальное отражение матрицы по горизонтали относительно середины (слева направо).

В первой строке записаны через пробел размеры матрицы: количество строк  $N$  и количество столбцов  $M$  ( $1 \leq N, M \leq 100$ ). В следующих  $N$  строках записаны строки матрицы, в каждой — по  $M$  натуральных чисел, разделённых пробелами.

Программа должна вывести матрицу, полученную в результате зеркального отражения исходной матрицы по горизонтали (слева направо).

Input	Output
4 5	15 14 13 12 11
11 12 13 14 15	30 29 28 27 26
26 27 28 29 30	45 44 43 42 41
41 42 43 44 45	60 59 58 57 56
56 57 58 59 60	

Q. *Поворот матрицы по часовой стрелке на 90 градусов*

Напишите программу, которая выполняет вращение квадратной матрицы вправо (на  $90^\circ$  по часовой стрелке).

В первой строке записан размер матрицы — количество строк и столбцов  $N$  ( $1 \leq N \leq 100$ ). В следующих  $N$  строках записаны строки матрицы, в каждой — по  $N$  натуральных чисел, разделённых пробелами.

Программа должна вывести матрицу, полученную из исходной вращением вправо (по часовой стрелке).

Input	Output
5	31 26 21 16 11
11 12 13 14 15	32 27 22 17 12
16 17 18 19 20	33 28 23 18 13
21 22 23 24 25	34 29 24 19 14
26 27 28 29 30	35 30 25 20 15
31 32 33 34 35	

R. *Поворот матрицы против часовой стрелки на 90 градусов*

Напишите программу, которая выполняет вращение квадратной матрицы влево (на  $90^\circ$  против часовой стрелке).

В первой строке записан размер матрицы — количество строк и столбцов  $N$  ( $1 \leq N \leq 100$ ). В следующих  $N$  строках записаны строки матрицы, в каждой — по  $N$  натуральных чисел, разделённых пробелами.

Программа должна вывести матрицу, полученную из исходной вращением влево (против часовой стрелки).

Input	Output
5	15 20 25 30 35
11 12 13 14 15	14 19 24 29 34
16 17 18 19 20	13 18 23 28 33
21 22 23 24 25	12 17 22 27 32
26 27 28 29 30	11 16 21 26 31
31 32 33 34 35	

S. *Треугольник Паскаля*

Даны два числа  $N$  и  $M$ . Создайте массив размера  $N \times M$  и заполните его по следующим правилам:

Числа, стоящие в строке 0 или в столбце 0 равны 1 ( $A[0][j] = 1$ ,  $A[i][0] = 1$ ). Для всех остальных элементов массива  $A[i][j] = A[i-1][j] + A[i][j-1]$ , то есть каждый элемент равен сумме двух элементов, стоящих слева и сверху от него.

В первой строке вводится натуральное число  $N$ , во второй натуральное число  $M$ .

Выведите данный массив на экран, отводя на вывод каждого элемента массива ровно 6 символов.

Input	Output					
4	1	1	1	1	1	1
6	1	2	3	4	5	6
	1	3	6	10	15	21
	1	4	10	20	35	56

T. *Треугольник Паскаля - 2*

Треугольник Паскаля состоит из чисел, где каждое число равно двум числам, стоящим над ним. Если перенумеровать строки треугольника Паскаля с нуля, то  $i$ -я строка содержит  $i + 1$  число, которые равны  $C_j^i$ , где  $j[0, i] = 1$ .

По данному числу  $n$  создайте список из  $n$  строк, где  $i$ -й элемент списка должен быть списком, содержащим  $i + 1$  число — элементы  $i$ -й строки треугольника Паскаля.

Заполните этот массив числами треугольника Паскаля.

Input	Output				
5	1				
	1	1			
	1	2	1		
	1	3	3	1	
	1	4	6	4	1

U. *Покупка билетов в кинотеатр*

В кинотеатре  $N$  рядов по  $M$  мест в каждом. В двумерном массиве хранится информация о проданных билетах, число 1 означает, что билет на данное место уже продано, число 0 означает, что место свободно. Ряды нумеруются, как и строки в таблице, сверху вниз.

Поступил запрос на продажу  $k$  билетов на соседние места в одном ряду. Определите, можно ли выполнить такой запрос.

Программа получает на вход числа  $N$  и  $M$ . Далее идет  $N$  строк, содержащих  $M$  чисел (0 или 1), разделенных пробелами. Затем дано число  $k$ .

Программа должна вывести номер ряда, в котором есть  $k$  подряд идущих свободных мест. Если таких рядов несколько, то выведите номер наименьшего подходящего ряда. Если подходящего ряда нет, выведите число 0.

Input	Output
3 4 0 1 0 1 1 0 0 1 1 1 1 1 2	2

V. *Дважды упорядоченный массив*

Таблица  $x$  размером  $N \times M$  упорядочена по строкам и столбцам, т.е.:

$$x[i][0] \leq x[i][1] \leq \dots \leq x[i][M-1], \forall i$$

$$x[0][j] \leq x[1][j] \leq \dots \leq x[N-1][j], \forall j$$

Если найдётся элемент массива, равный заданному числу, напечатать YES, иначе напечатать NO. В первой строке через пробел указаны три натуральных числа  $N, M, K$  ( $1 \leq N, M \leq 500, K \leq 10^4$ ) — количество строк, количество столбцов и количество запросов. Затем в  $N$  строках записаны по  $M$  целых чисел в каждой, разделённых пробелами. Затем следуют  $K$  строк, в каждой из которых записано одно целое число.

Требуется вывести  $K$  строк — ответы на запросы. Если число найдено в таблице, то следует вывести слово YES; если числа нет, вывести слово NO.

**Подсказка:** у этой задачи есть очень простое решение, сложность одного запроса  $O(N + M)$ .

Input	Output
3 3 2 1 2 3 4 5 6 7 8 9 6 97	YES NO

W. *Седловые точки массива*

Найти все седловые точки в прямоугольной таблице. Седловой точкой называется элемент таблицы, равный минимуму в своей строке и максимуму в своём столбце.

Программа получает на вход числа  $N$  и  $M$ . Далее идет  $N$  строк, содержащих  $M$  чисел, разделенных пробелами.

Программа должна вывести индексы всех седловых точек матрицы в порядке обхода по строкам (сверху вниз, слева направо). Номер строки и номер столбца каждой седловой точки разделяются пробелами. Нумерация строк и столбцов начинается с единицы. Если в матрице нет ни одной седловой точки, нужно вывести число 0.

Input	Output
4 5 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 9 17 18 19 20	3 1

■ В задачах X-Z разрешается использовать дополнительный одномерный массив длины не более  $N$  (наибольшая сторона).

### X. Обнуление массива

В заданной квадратной таблице заменить нулями те и только те элементы, стоящие в строках или столбцах, где имеются нули.

В первой строчке записано одно натуральное число  $N$  ( $N < 400$ ). Затем в  $N$  строках перечисляются элементы массива, по  $N$  в каждой строке, через пробел.

Программа должна вывести квадратный массив  $N \times N$ , заполненный в соответствии с условием задачи.

Input	Output
4	0 2 3 0
1 2 3 4	0 0 0 0
0 3 4 5	0 0 0 0
1 2 3 0	0 6 6 0
6 6 6 6	

### Y. Общий элемент в строках

Дана квадратная таблица  $x$  размером  $N \times N$ . Каждая её строка упорядочена по неубыванию.

Если есть число, встречающееся во всех строках, вывести YES. Если такого числа не окажется — вывести NO.

Сначала на вход программы подаётся натуральное число  $N$ . Затем в  $N$  строках записаны через пробел по  $N$  чисел в каждой строке, причём числа в каждой строке упорядочены по неубыванию. Программа должна вывести одну строку, где написано YES, если хотя бы одно такое число существует и NO, если такого числа нет.

Input	Output
3	YES
1 2 3	
2 3 3	
3 3 3	

### Z. Все общие элементы в строках

В условиях предыдущей задачи вывести в порядке возрастания все числа, встречающиеся в каждой строке таблицы. Если ни одного такого числа не окажется, вывести NO.

Сначала на вход программы подаётся натуральное число  $N$ . Затем в  $N$  строках записаны через пробел по  $N$  чисел в каждой строке, причём числа в каждой строке упорядочены по неубыванию. Программа должна вывести одну строку, где через пробел должны быть перечислены в порядке возрастания все числа, которые встречаются в каждой строке данной таблицы. Если таких чисел нет — вывести NO.

Input	Output
4	1 5
0 1 5 6	
1 4 5 6	
1 3 5 5	
1 2 5 5	

Input	Output
7	NO
1 3 4 4 4 6 6	
1 2 2 2 3 5 6	
1 1 3 3 3 4 4	
3 3 4 4 5 6 6	
1 2 4 5 6 6 6	
1 2 3 4 5 5 6	
1 1 1 2 3 5 6	

### ЗА. Острова

Дана квадратная таблица  $x$  размером  $N \times N$ , заполненная нулями и единицами. Суша обозначается единицей, вода нулём. Участки суши (клетки) граничат друг с другом, если они имеют общую сторону, т.е. с клеткой с координатами  $[i][j]$  будут граничить клетки с координатами  $[i-1][j]$ ,  $[i+1][j]$ ,  $[i][j-1]$  и  $[i][j+1]$ .

Напишите программу, подсчитывающую количество островов.

Сначала на вход программы подаётся натуральное число  $N$  ( $N \leq 100$ ). Затем в  $N$  строках записаны через пробел по  $N$  чисел в каждой строке.

Программа должна вывести одно число — ответ на вопрос задачи.

Input	Output
3 1 1 1 0 0 0 1 1 1	2

**Указание:** в этой задаче исходный массив можно изменять.

Для изменения ограничения максимальной глубины рекурсии в программе следует использовать функцию `setrecursionlimit(N)`, где  $N$  — требуемая глубина рекурсии. Функцию `setrecursionlimit` надо импортировать из модуля `sys`.

### ZB. Простой квадрат

У Пети имеется игровое поле размером  $3 \times 3$ , заполненное числами от 1 до 9, все числа разные. В начале игры он может поставить фишку в любую клетку поля. На каждом шаге игры разрешается перемещать фишку в любую соседнюю по стороне клетку, но не разрешается посещать одну и ту же клетку дважды.

Петя внимательно ведёт протокол игры, записывая в него цифры в том порядке, в котором фишка посещала клетки. Пете стало интересно, какое максимальное число он может получить в протоколе. Помогите ему ответить на этот вопрос.

Входной файл содержит описание поля — 3 строки по 3 целых числа, разделенных пробелами. Гарантируется, что все девять чисел различны и лежат в диапазоне от 1 до 9

Выведите одно целое число — максимальное число, которое могло получиться в протоколе при игре на данном поле.

Ответ можно выводить не в виде числа, а в виде строки или в виде последовательности отдельных цифр (но не разделяя их пробелами).

Input	Output
1 2 3 4 5 6 7 8 9	987456321

Если вам потребуется передавать в функцию двумерный массив по значению (т.е. изменения массива внутри функции не отражаются на переданном массиве вне её), это делается так:

```
from copy import deepcopy
...
def f(array, ...)
...
# x - двумерный массив
f(deepcopy(x), ...)
```