Обход в ширину (BFS, Breadth-First-Search) и кратчайшие пути.

A. *Кратчайший путь в невзвешенном графе*

В неориентированном графе требуется найти минимальный путь между двумя вершинами.

Первым на вход поступает число N — количество вершин в графе ($1 \leqslant N \leqslant 100$). Затем записана матрица смежности (0 обозначает отсутствие ребра, 1 — наличие ребра). Далее задаются номера двух вершин — начальной и конечной.

Выведите сначала L — длину кратчайшего пути (количество ребер, которые нужно пройти), а потом сам путь — номера всех вершин в правильном порядке. Если путь имеет длину 0, то его выводить не нужно, достаточно вывести длину. Если пути нет, нужно вывести -1.

Input	Output

`5`
`0 1 0 0 1`
`1 0 1 0 0`
`0 1 0 0 0`
`0 0 0 0 0`
`1 0 0 0 0`
`3 5`
	`3`
	`3 2 1 5`

B. *Два коня*

На стандартной шахматной доске (8×8) живут 2 шахматных коня. Им нужно оказаться на одной клетке. Но наши шахматные кони ходят не по очереди, а одновременно, и если оказываются на одной клетке, никто никого не съедает. Сколько ходов им потребуется, чтобы оказаться в одной клетке?

На вход программы поступают координаты коней, записанные по стандартным шахматным правилам (т.е. двумя символами — маленькая латинская буква (от `a` до `h`) и цифра (от 1 до 8), задающие столбец и строку соответственно).

Требуется вывести наименьшее необходимое количество ходов, либо число -1, если кони не могут встретиться.

Input	Output
`a1 a3`	`1`

C. *Табличка*

Дана таблица, состоящая из N строк и M столбцов. В каждой клетке таблицы записано одно из чисел: 0 или 1. Расстоянием между клетками (x_1, y_1) и (x_2, y_2) назовем сумму $|x_1 - x_2| + |y_1 - y_2|$. Вам необходимо построить таблицу, в клетке (i, j) которой будет записано минимальное расстояние между клеткой (i, j) начальной таблицы и клеткой, в которой записана 1. Гарантируется, что хотя бы одна 1 в таблице есть.

В первой строке вводятся два натуральных числа N и M, не превосходящих 500. Далее идут N строк по M чисел — элементы таблицы.

Требуется вывести N строк по M чисел — элементы искомой таблицы.

Input	Output

`2 3`
`0 0 1`
`1 0 0`
	`1 1 0`
	`0 1 1`

D. *Поиск клада*

Есть лабиринт размера $N \times M$ ($1 \leqslant N, M \leqslant 100, N \times M \leqslant 100$). Каждая клетка лабиринта либо пуста и по ней можно пройти, либо содержит стену. Из клетки можно переходить только в смежную по стене клетку (так, у каждой клетки может быть не более 4 смежных).

В одной из клеток находится клад. Кроме того, в лабиринте есть K входов, из которых можно начать свой путь.

Требуется определить, с какого входа нужно начать свой путь, чтобы пройденное расстояние до клада было наименьшим. Если таких входов несколько, нужно вывести вход с наименьшим номером.

В первой строке записаны 2 числа N и M, задающие размеры лабиринта. Далее следует описание лабиринта: N строк по M символов в каждой.

- `0` означает, что клетка свободна;
- `1` означает, что в клетке находится стена;
- `*` обозначает клетку с сокровищем (такая клетка в лабиринте ровно одна)

В следующей строке записано число K ($1 \leqslant K \leqslant N \times M$) — количество входов в лабиринт. Далее в K строках записаны координаты входов (номер строки и номер столбца, которые нумеруются с 1).

Гарантируется, что координаты входов попарно различны, и то, что все входы расположены в пустых клетках. Ни один из входов не находится в клетке с сокровищем.

Необходимо вывести одно число — искомый номер входа (нумерация начинается с 1). Если до сокровища невозможно добраться, выведите -1.

Input	Output
5 5 00000 00000 10*00 01111 00000 4 1 1 1 5 4 1 5 5	1
3 3 010 1*1 010 4 1 1 1 3 3 1 3 3	-1

E. *Игрушечный лабиринт*

Игрушечный лабиринт представляет собой прозрачную плоскую прямоугольную коробку, внутри которой есть препятствия и перемещается шарик. Лабиринт можно наклонять влево, вправо, к себе или от себя, после каждого наклона шарик перемещается в заданном направлении до ближайшего препятствия или до стенки лабиринта, после чего останавливается. Целью игры является загнать шарик в одно из специальных отверстий — выходов. Шарик проваливается в отверстие, если оно встречается на его пути (шарик не обязан останавливаться в отверстии).

В первой строке входного файла записаны числа N и M — размеры лабиринта (целые положительные числа, не превышающие 100). Затем записаны N строк по M чисел в каждой — описание лабиринта. Число 0 означает свободное место, число 1 — препятствие, число 2 — отверстие.

Выведите единственное число — минимальное количество наклонов, которые необходимо сделать, чтобы шарик покинул лабиринт через одно из отверстий.

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

F. *Только направо*

В прямоугольнике из клеток нарисован лабиринт. Одна из его клеток это выход из лабиринта. Робот, который ходит по лабиринту, умеет поворачивать только направо, но не умеет поворачивать налево и разворачиваться на месте (то есть после поворота направо он должен сделать хотя бы один шаг в новом направлении). Требуется определить длину кратчайшего пути до выхода из лабиринта.

В первой строке через пробел записаны числа R и C ($4 \leqslant R, C \leqslant 20$) — количество строк и столбцов в карте лабиринта. В каждой из следующих R строк записано по C символов, задающих эту карту. Символ S обозначает положение робота, символ F — место выхода из лабиринта, символ X — стенку. Пробелами обозначены проходимые клетки. Гарантируется, что лабиринт окружен стенами. Перед началом движения робот может сориентироваться по любому из 4 направлений (вверх, вниз, влево или направо).

Выведите единственное число — расстояние, которое придется пройти роботу. Гарантируется, что он всегда сможет выйти из лабиринта.

Input	Output
6 6 XXXXXX XF X X X XXX X X S X XXXXXX	9
6 6 XXXXXX XS X X X XXX X X F X XXXXXX	5

G. *Наименьшее кратное*

Дано число X и множество цифр D. Требуется дописать к X минимальное количество цифр из D, чтобы получившееся число делилось на k. При этом получившееся число должно быть минимально возможным.

Первая строка входного файла содержит два натуральных числа X и k ($1 \leqslant X \leqslant 10^{1000}, 2 \leqslant k \leqslant 10^5$). Во второй строке записано количество цифр в множестве D (число от 0 до 10). В третьей строке через пробел записаны эти цифры.

Единственная строка должна содержать минимальное число, полученное из X дописыванием цифр из D и кратное k. Если такого числа не существует, выведите -1.

Input	Output
102 101 3 1 0 3	10201
202 101 3 1 0 3	202

H. *Алгоритм Дейкстры – кратчайшее расстояние*

Дан ориентированный взвешенный граф. Найдите кратчайшее расстояние от одной заданной вершины до другой.

В первой строке содержатся три числа: N, S и F ($1 \leqslant N \leqslant 100, 1 \leqslant S, F \leqslant N$), где N — количество вершин графа, S — начальная вершина, а F — конечная. В следующих N строках вводится по N чисел, не превосходящих 100, — матрица смежности графа, где -1 означает отсутствие ребра между вершинами, а любое неотрицательное число — присутствие ребра данного веса. На главной диагонали матрицы записаны нули.

Требуется вывести искомое расстояние или -1, если пути между указанными вершинами не существует.

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

I. *Алгоритм Дейкстры – кратчайший маршрут*

Дан ориентированный взвешенный граф. Найдите кратчайшее расстояние от одной заданной вершины до другой.

В первой строке содержатся три числа: N, S и F ($1 \leqslant N \leqslant 100, 1 \leqslant S, F \leqslant N$), где N — количество вершин графа, S — начальная вершина, а F — конечная. В следующих N строках вводится по N чисел, не превосходящих 100, — матрица смежности графа, где -1 означает отсутствие ребра между вершинами, а любое неотрицательное число — присутствие ребра данного веса. На главной диагонали матрицы записаны нули.

Требуется вывести последовательно все вершины одного (любого) из кратчайших путей, или одно число -1, если пути между указанными вершинами не существует.

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

J. *Минимальное остовное дерево*

Требуется найти в связном графе остовное дерево минимального веса.

Первая строка входного файла содержит два натуральных числа N и M — количество вершин и рёбер графа соответственно ($1 \leqslant n \leqslant 20000, 0 \leqslant M \leqslant 100000$). Следующие M строк содержат описание рёбер по одному на строке. Ребро номер i описывается тремя натуральными числами b_i, e_i и w_i — номера концов ребра и его вес соответственно ($1 \leqslant b_i, e_i \leqslant n, 0 \leqslant w_i \leqslant 100000$).

Выведите единственное целое число — вес минимального остовного дерева.

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

K. *Авиаперелёты*

Есть ориентированный взвешенный граф, в котором могут быть петли и кратные рёбра, а все веса положительны. Граф хранит информацию об авиарейсах между городами и их цене. Надо найти длину кратчайшего маршрута, состоящего не более чем из K перелётов.

В первой строке находятся числа N (количество городов), M (количество авиарейсов), K (количество оставшихся ночей), S (номер города, в котором живет профессор), F (номер города, в котором проводится конференция).

Ограничения: $2 \leqslant N \leqslant 100, 1 \leqslant M \leqslant 10^5, 1 \leqslant K \leqslant 100, 1 \leqslant S \leqslant N, 1 \leqslant F \leqslant N$.

Далее идет M строк, задающих расписание авиарейсов. i-я строка содержит три натуральных числа: S_i, F_i и P_i, где:

- S_i — номер города, из которого вылетает i-й рейс;
- F_i — номер i-города, в который прилетает i-й рейс;
- P_i — стоимость перелета i-м рейсом.

Ограничения на входные данные следующие: $1 \leqslant S_i \leqslant N, 1 \leqslant F_i \leqslant N, 1 \leqslant P_i \leqslant 10^6$.

Выведите одно число — минимальную стоимость подходящего пути. Если за K или меньше ночей добраться до пунктв назначения нельзя, выведите число -1.

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

L. *Самый длинный оптимальный путь*

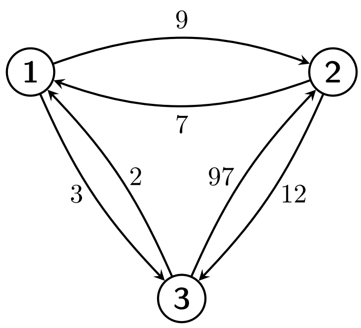
Есть ориентированный взвешенный граф с неотрицательными весами рёбер. Среди всех пар вершин найти две такие вершины, минимальная длина пути между которыми наибольшая.

В первой строке задано число вершин N ($N \leqslant 70$). В следующих N строках записана матрица смежности графа. Число в i-ой строке с номером j задаёт вес ребра, ведущего из вершины с номером i в вершину с номером j. Веса не превосходят 10^6. Петель в графе нет.

Выведите одно число – длину искомого пути.

Input	Output
3 0 9 3 7 0 12 2 97 0	11

Комментарий к тесту:
кратчайший путь из 1 в 2 имеет длину 9
кратчайший путь из 2 в 1 имеет длину 7
кратчайший путь из 2 в 3 имеет длину 10
кратчайший путь из 3 в 2 имеет длину 11 (**это ответ**)
кратчайший путь из 1 в 3 имеет длину 3
кратчайший путь из 3 в 1 имеет длину 2



M. *Самый короткий путь*

Дан ориентированный взвешенный полный граф. Нужно вычислить минимум длин всех возможных путей между всеми парами различных вершин этого графа.

В первой строке задано число вершин N ($N \leqslant 70$). В следующих N строках записана матрица смежности графа. Число в i-ой строке с номером j задаёт вес w_{ij} ребра, ведущего из вершины с номером i в вершину с номером j, ($|w_{ij}| \leqslant 10^6$).

Если такой минимум существует, вывести его.

Если минимума не существует (т.е. если в графе можно найти путь отрицательной длины, сколь угодно большой по модулю) — вывести число -1.

Input	Output
3 0 42 18468 6335 0 26501 19170 15725 0	42
3 0 -7 3 -2 0 10 2 215 0	-1

N. *Транзитивное замыкание*

Невзвешенный ориентированный граф задан своей матрицей смежности. Требуется построить его транзитивное замыкание, то есть матрицу, в которой в i-й строке и j-м столбце находится 1, если от вершины i можно добраться до вершины j, и 0 — иначе.

В первой строке входных данных записано число N $(1 \leqslant V \leqslant 100)$ — количество вершин графа. В следующих N строках записана матрица смежности графа — в каждой строке N чисел 0 (нет ребра) или 1 (есть ребро). На диагонали матрицы стоят единицы (т.е. из вершины можно «добраться» в саму себя).

Необходимо вывести матрицу транзитивного замыкания графа в формате, аналогичном формату матрицы смежности.

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

O. *Длины кратчайших путей*

Дан ориентированный взвешенный граф, в котором могут быть кратные рёбра и петли. Каждое ребро имеет вес, выражающийся целым числом (возможно, отрицательным). Циклы отрицательного веса отсутствуют.

Надо посчитать длины кратчайших путей от вершины номер 1 до всех остальных вершин.

В первой строке входных данных записаны числа V $(1 \leqslant V \leqslant 100)$ — количество вершин графа и число E $(0 \leqslant V \leqslant 10^4)$ — количество рёбер. В следующих строках записано E троек чисел, описывающих ребра: начало ребра, конец ребра и его вес (целое число от -100 до 100).

Программа должна вывести V чисел — расстояния от вершины номер 1 до всех вершин графа. Если пути до соответствующей вершины не существует, вместо длины пути выведите число 30000.

Input	Output
6 4 1 2 10 2 3 10 1 3 100 4 5 -10	0 10 20 30000 30000 30000