Задача A. Обход в ширину

Ограничение по времени: 2 секунды
Ограничение по памяти: 64 мегабайта

Дан неориентированный граф. В нём необходимо найти расстояние от одной заданной вершины до другой.

Формат входных данных

В первой строке входного файла содержится три натуральных числа N, S и F ($1 \leqslant S, F \leqslant N \leqslant 100$) — количество вершин в графе и номера начальной и конечной вершин соответственно. Далее в N строках задана матрица смежности графа. Если значение в j-м элементе i-й строки равно 1, то в графе есть ребро из вершины i в вершину j.

Формат выходных данных

В единственной строке должно находиться минимальное расстояние от начальной вершины до конечной. Если пути не существует, выведите 0.

Пример

bfs.in	bfs.out
4 4 3 0 1 1 1 1 0 1 0 1 1 0 0 1 0 0 0	2

Задача B. Путь

Ограничение по времени: 2 секунды
Ограничение по памяти: 64 мегабайта

В неориентированном графе требуется найти минимальный путь между двумя вершинами.

Формат входных данных

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.

Формат выходных данных

Выведите сначала L — длину кратчайшего пути (количество ребер, которые нужно пройти), а затем $L+1$ число — путь от одной вершины до другой, заданный своими вершинами. Если пути не существует, выведите одно число -1.

Пример

path.in	path.out
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

Задача C. Про коня

Ограничение по времени: 2 секунды
Ограничение по памяти: 64 мегабайта

На шахматной доске 8×8 указаны две несовпадающие клетки. Найдите кратчайший маршрут коня из первой клетки во вторую.

Формат входных данных

Во входном файле записаны координаты двух клеток. Каждая координата представлена двумя символами, где сначала указана одна строчная буква от `a` до `h`, а после буквы (без пробела) цифра от `1` до `8`, например `h8`. Каждая клетка записана в отдельной строке.

Формат выходных данных

Программа должна вывести последовательность клеток, первая из которых совпадает с первой данной, а последняя совпадает со второй данной. Две соседние клетки должны быть соединены ходом коня, при этом количество клеток в последовательности должно быть минимально возможным.

Пример

knight.in	knight.out
a1 b1	a1 b3 d2 b1

Задача D. Максимум по минимуму

Ограничение по времени: 2 секунды
Ограничение по памяти: 64 мегабайта

Кракозябрик Пушистик очень часто ездит на гастроли со своими авторскими выступлениями игры на доремифасольчиках.

Но сегодня всё по-другому. Пушистик организует выступление для своих друзей у себя дома. Чтобы никто не опоздал и не пропустил ни одного из шедевров нашего героя, Пушистик решил посчитать, сколько будет добираться тот гость, который будет ехать до него дольше всех.

Автобусы в родном городе музыканта Финденляндии ходят очень странно — если автобус ходит от остановки «Музыкальная школа» до остановки «Консерватория», то это не значит, что автобус с таким же номером ходит от «Консерватории» до «Музыкальной школы».

Пушистик долго наблюдал за автобусами в своём городе и выяснил, что время, за которое любой кракозябрик может добраться до дома музканта, равно минимальному количеству автобусов, на котором ему придётся проехать.

Формат входных данных

В первой строке входного файла содержится три натуральных числа N, M и S ($1 \leqslant S \leqslant N \leqslant 5000$, $1 \leqslant M \leqslant 20000$) — количество остановок автобусов в Финденляндии, количество автобусов и номер остановки, рядом с которой живёт Пушистик. Каждый автобус имеет ровно две остановки — ту, от которой он едет, и ту, до которой он едет. Далее в M строках перечислены маршруты автобусов. Каждый маршрут задаётся парой чисел — номерами начальной и конечной остановок соответственно.

Формат выходных данных

Вывести одно целое число — искомое минимальное количество автобусов, на котором придётся проехать гостю, который будет добираться дольше всех.

Пример

maxmin.in	maxmin.out
3 5 3 1 2 2 1 3 1 2 3 3 3	2