Задача A. Обход в глубину

Имя входного файла:	`dfs.in`
Имя выходного файла:	`dfs.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Дан неориентированный невзвешенный граф, в котором выделена вершина. Вам необходимо найти количество вершин, лежащих с ней в одной компоненте связности (включая саму выделенную вершину).

Формат входного файла

В первой строке входного файла содержатся два целых числа N и S ($1 \leqslant S \leqslant N \leqslant 100$), где N — количество вершин графа, а S — выделенная вершина. В следующих N строках записано по N чисел — матрица смежности графа, в которой цифра «0» означает отсутствие ребра между вершинами, а цифра «1» — его наличие. Гарантируется, что на главной диагонали матрицы всегда стоят нули.

Формат выходного файла

Выведите одно целое число — искомое количество вершин.

Примеры

dfs.in	dfs.out
5 1 0 1 1 0 0 1 0 1 0 0 1 1 0 0 0 0 0 0 0 1 0 0 0 1 0	3

Задача B. Компоненты связности

Имя входного файла:	`components.in`
Имя выходного файла:	`components.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Дан неориентированный невзвешенный граф. Необходимо посчитать количество его компонент связности.

Формат входного файла

В первой строке входного файла содержится одно натуральное число N ($N \leqslant 100$) — количество вершин в графе. Далее в N строках по N чисел — матрица смежности графа: в i-й строке на j-м месте стоит «1», если вершины i и j соединены ребром, и «0», если ребра между ними нет. На главной диагонали матрицы стоят нули. Матрица симметрична относительно главной диагонали.

Формат выходного файла

Вывести одно целое число — искомое количество компонент связности графа.

Примеры

components.in	components.out
6 0 1 1 0 0 0 1 0 1 0 0 0 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0	3

Задача C. Долой списывание!

Имя входного файла:	`bipartite.in`
Имя выходного файла:	`bipartite.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Во время контрольной работы профессор Флойд заметил, что некоторые студенты обмениваются записками. Сначала он хотел поставить им всем двойки, но в тот день профессор был добрым, а потому решил разделить студентов на две группы: списывающих и дающих списывать, и поставить двойки только первым.

У профессора записаны все пары студентов, обменявшихся записками. Требуется определить, сможет ли он разделить студентов на две группы так, чтобы любой обмен записками осуществлялся от студента одной группы студенту другой группы.

Формат входного файла

В первой строке находятся два числа N и M — количество студентов и количество пар студентов, обменивающихся записками ($1 \leqslant N \leqslant 100$, $0 \leqslant M \leqslant \frac{N(N-1)}{2}$). Далее в M строках расположены описания пар студентов: два различных числа, соответствующие номерам студентов, обменивающихся записками (нумерация студентов идёт с 1). Каждая пара студентов перечислена не более одного раза.

Формат выходного файла

Необходимо вывести ответ на задачу профессора Флойда. Если возможно разделить студентов на две группы, выведите «YES»; иначе выведите «NO».

Примеры

bipartite.in	bipartite.out
3 2 1 2 2 3	YES

Задача D. Получи дерево

Имя входного файла:	`tree.in`
Имя выходного файла:	`tree.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Дан связный неориентированный граф без петель и кратных рёбер, из которого разрешается удалять ребра. Требуется получить из данного графа дерево.

Формат входного файла

Сначала вводятся два целых числа: N и M ($1 \leqslant N \leqslant 100, 0 \leqslant M \leqslant \frac{N(N-1)}{2}$) — количество вершин и рёбер графа соответственно. Далее идёт M пар чисел, задающих ребра. Гарантируется, что граф связный.

Формат выходного файла

Выведите $N - 1$ пару чисел — рёбра, которые войдут в дерево. Рёбра можно выводить в любом порядке.

Примеры

tree.in	tree.out
4 4 1 2 2 3 3 4 4 1	1 2 2 3 3 4

Задача E. TopSort

Имя входного файла:	`topsort.in`
Имя выходного файла:	`topsort.out`
Ограничение по времени:	3 секунды
Ограничение по памяти:	64 мегабайта

Дан ориентированный невзвешенный граф. Необходимо его топологически отсортировать.

Формат входного файла

В первой строке входного файла даны два натуральных числа N и M ($1 \leqslant N \leqslant 10^5, 1 \leqslant M \leqslant 10^5$) — количество вершин и рёбер в графе соответственно. Далее в M строках перечислены рёбра графа. Каждое ребро задаётся парой чисел — номерами начальной и конечной вершин соответственно.

Формат выходного файла

Вывести любую топологическую сортировку графа в виде последовательности номеров вершин. Если граф невозможно топологически отсортировать, требуется вывести -1.

Примеры

topsort.in	topsort.out
6 6 1 2 3 2 4 2 2 5 6 5 4 6	4 6 3 1 2 5
3 3 1 2 2 3 3 1	-1

Задача F. Есть ли цикл?

Имя входного файла:	`cycle.in`
Имя выходного файла:	`cycle.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Дан ориентированный граф. Требуется определить, есть ли в нем цикл.

Формат входного файла

В первой строке вводится натурально число N ($N \leqslant 50$) — количество вершин. Далее в N строках следуют по N чисел, каждое из которых — «`0`» или «`1`». j-е число в i-й строке равно «`1`» тогда и только тогда, когда существует ребро, идущее из i-й вершины в j-ю. Гарантируется, что на диагонали матрицы будут стоять нули.

Формат выходного файла

Выведите «`0`», если в заданном графе цикла нет, и «`1`», если он есть.

Примеры

cycle.in	cycle.out
3 0 1 0 0 0 1 0 0 0	0