

Задача А. Светофоры

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

В подземелье M тоннелей и N перекрестков, каждый тоннель соединяет какие-то два перекрестка. Мышиный король решил поставить по светофору в каждом тоннеле перед каждым перекрестком. Напишите программу, которая посчитает, сколько светофоров должно быть установлено на каждом из перекрестков. Перекрестки пронумерованы числами от 1 до N .

Формат входных данных

Во входном файле записано два числа N и M ($0 < N \leq 100$, $0 \leq M \leq \frac{N(N-1)}{2}$). В следующих M строках записаны по два числа i и j ($1 \leq i, j \leq N$), которые означают, что перекрестки i и j соединены тоннелем. Гарантируется, что никакой тоннель не соединяет перекресток сам с собой, и не существует двух различных тоннелей, соединяющих одну и ту же пару вершин.

Формат выходных данных

В выходной файл вывести N чисел: k -е число означает количество светофоров на k -м перекрестке.

Примеры

lights.in	lights.out
7 10 5 1 3 2 7 1 5 2 7 4 6 5 6 4 7 5 2 1 5 3	3 3 2 2 5 2 3

Задача В. Цветной дождь

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

В Банановой республике очень много холмов, соединенных мостами. На химическом заводе произошла авария, в результате чего испарилось экспериментальное удобрение «зован». На следующий день выпал цветной дождь, причем он прошел только над холмами. В

некоторых местах падали красные капли, в некоторых — синие, а в остальных — зеленые, в результате чего холмы стали соответствующего цвета. Президенту Банановой республики это понравилось, но ему захотелось покрасить мосты между вершинами холмов так, чтобы мосты были покрашены в цвет холмов, которые они соединяют. К сожалению, если холмы разного цвета, то покрасить мост таким образом не удастся. Посчитайте количество таких «плохих» мостов.

Формат входных данных

В первой строке файла записано число N — количество холмов ($1 \leq N \leq 100$). Во второй и далее — матрица смежности, описывающая наличие мостов между холмами. В последней строке написаны N чисел k_1, k_2, \dots, k_N , которые обозначают цвет соответствующего холма: 1 — красный, 2 — синий, 3 — зеленый.

Гарантируется, что матрица смежности симметрична относительно главной диагонали, а элементы на диагонали содержат нули.

Формат выходных данных

Выведите количество мостов, соединяющих холмы разных цветов.

Примеры

rain.in	rain.out
1 0 1	0
7 0 1 0 0 0 1 1 1 0 1 0 0 0 0 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0 1 0 1 0 0 1 0 0 0 0 0 0 1 1 1 1 1 3 3	4

Задача С. От матрицы смежности к спискам смежности

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

Простой ориентированный граф задан матрицей смежности. Выведите его представление в виде списков смежности.

Формат входных данных

В первой строке файла находится число N — количество вершин графа ($1 \leq N \leq 100$). Во второй строке и далее — матрица смежности. Гарантируется, что граф не содержит

петель.

Формат выходных данных

Выведите N строк — списки смежности графа. В i -й строке сначала выведите количество исходящих из i -й вершины рёбер, а затем — номера вершин, в которые эти рёбра идут, упорядоченные по возрастанию.

Примеры

mtoal.in	mtoal.out
5	1 3
0 0 1 0 0	2 1 3
1 0 1 0 0	1 5
0 0 0 0 1	2 1 2
1 1 0 0 0	2 1 2
1 1 0 0 0	

Задача D. От списков смежности к матрице смежности

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

Формат входных данных

В первой строке входного файла содержится число N — количество вершин ($1 \leq N \leq 100$). Далее идут N строк. В i -й строке содержится описание всех рёбер, исходящих из i -й вершины. Описание начинается количеством исходящих рёбер. Далее следуют номера вершин, в которые эти рёбра идут. Все вершины нумеруются натуральными числами от 1 до N . Гарантируется, что i -й список смежности не содержит числа i , а также все списки не содержат повторяющихся чисел.

Формат выходных данных

Выведите матрицу смежности ориентированного графа.

Примеры

altom.in	altom.out
3	0 1 1
2 2 3	0 0 0
0	0 1 0
1 2	

Задача E. Проверка на неориентированность

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

По матрице $N \times N$ из нулей и единиц определите, может ли данная матрица быть матрицей смежности простого неориентированного графа.

Формат входных данных

В первой строке число N ($1 \leq N \leq 100$), далее матрица — N строк по N чисел, каждое из которых равно 0 или 1.

Формат выходных данных

Выведите YES, если приведенная матрица может быть матрицей смежности простого неориентированного графа, иначе выведите NO.

Примеры

check.in	check.out
3 0 1 1 1 0 1 1 1 0	YES
3 0 1 0 1 0 1 1 1 0	NO
3 0 1 0 1 1 1 0 1 0	NO

Задача F. Подсчет количества ребер неориентированного графа

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

Простой неориентированный граф задан матрицей смежности. Найдите количество ребер в графе.

Формат входных данных

В первой строке число N — число вершин в графе ($1 \leq N \leq 100$), затем матрица смежности — N строк по N чисел, каждое из которых равно 0 или 1.

Формат выходных данных

Выведите количество ребер заданного графа.

Примеры

count.in	count.out
3 0 1 1 1 0 1 1 1 0	3

Задача G. Проверка на наличие кратных ребер, ориентированный вариант

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

Ориентированный граф задан списком ребер. Проверьте, содержит ли он кратные ребра.

Формат входных данных

N — число вершин и M — число ребер ($1 \leq N \leq 100$, $1 \leq M \leq 10\,000$), затем M пар чисел — ребра графа.

Формат выходных данных

Выведите YES, если граф содержит параллельные ребра, иначе NO.

Примеры

check.in	check.out
5 3 2 5 3 1 3 2	NO
3 5 1 2 2 3 3 1 2 3 2 1	YES

Задача H. Полустепени вершин

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

Ориентированный граф задан матрицей смежности. Найдите полустепени захода и по-

лустепени исхода всех вершин графа (т. е. количество входящих в нее и исходящих из нее ребер соответственно для каждой вершины).

Формат входных данных

N — число вершин в графе ($1 \leq N \leq 100$), затем матрица смежности: N строк по N чисел, каждое из которых равно 0 или 1.

Формат выходных данных

Выведите N пар чисел: для каждой вершины сначала полустепень захода и затем полустепень исхода.

Примеры

half-degree.in	half-degree.out
4 0 1 0 1 1 0 1 1 0 1 0 0 1 1 1 1	2 2 3 3 2 1 3 4

Задача I. Истоки и стоки

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

Вершина ориентированного графа называется истоком, если в нее не входит ни одно ребро, и стоком, если из нее не выходит ни одного ребра.

Ориентированный граф задан матрицей смежности. Найдите все его вершины-истоки и все вершины-стоки.

Формат входных данных

N — число вершин в графе ($1 \leq N \leq 100$), затем матрица смежности — N строк по N чисел, каждое из которых равно 0 или 1.

Формат выходных данных

В первой строке выведите K — число истоков в графе, затем номера вершин, являющихся истоками в порядке возрастания. Во второй строке выведите информацию о стоках в том же формате.

Примеры

source.in	source.out
5 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0	2 3 4 3 1 4 5

Задача J. Полный граф

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

Неориентированный граф называется полным, если любая пара его различных вершин соединена хотя бы одним ребром. Для заданного списком ребер графа проверьте, является ли он полным.

Формат входных данных

Программе на вход даются числа N и M , где N — число вершин ($1 \leq N \leq 100$) и M — число ребер ($1 \leq M \leq 10000$), а затем M пар чисел — ребра графа.

Формат выходных данных

Выведите YES, если граф является полным, и NO в противном случае.

Примеры

<code>complete.in</code>	<code>complete.out</code>
3 3 1 2 1 3 2 3	YES

Задача K. Транзитивность графа

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

Напомним, что неориентированный граф без петель и кратных рёбер называется транзитивным, если всегда из того, что вершины u и v соединены ребром, вершины v и w соединены ребром и все три вершины u , v и w различны, следует, что вершины u и w соединены ребром.

Проверьте, что заданный неориентированный граф является транзитивным.

Формат входных данных

В первой строке — числа N и M — количество вершин и рёбер графа ($3 \leq N \leq 100$, $0 \leq M \leq \frac{N(N-1)}{2}$). Далее идет M строк — список ребер.

Формат выходных данных

Выведите YES или NO, как ответ на вопрос о транзитивности графа.

Примеры

<code>transitive.in</code>	<code>transitive.out</code>
3 3 1 2 2 3 1 3	YES
3 2 1 2 1 3	NO

Задача L. Переселение сыщика

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

Ниро Вульф решил переехать в другой город. Одна седьмая тонны веса мешает ему перемещаться быстро. В работе детектива необходимо быстро оказываться на месте преступления, а впоследствии настигать преступника, пока он не успел сбежать. Поэтому Вульф ищет такой город, в котором он мог бы с одной площади попасть на другую, проехав не более чем по двум улицам.

Напишите для него программу, которая по карте города сообщала, обладает ли город нужным свойством.

Формат входных данных

В первой строке заданы два числа: n — количество площадей ($n < 100$) и m — количество улиц между площадями.

В последующих m строках содержится пара чисел от 1 до n — начало и конец улицы.

Формат выходных данных

Выведите «Yes», если город пригоден для жизни, и «No» в противном случае.

Примеры

<code>two.in</code>	<code>two.out</code>
3 2 1 2 2 3	YES
4 3 1 2 2 3 3 4	NO