Задача A. Ярый коллекционер бабочек

Имя входного файла:	`collect.in`
Имя выходного файла:	`collect.out`
Ограничение по времени:	4 секунды
Ограничение по памяти:	64 мегабайта

Как известно, Андрей Сергеевич — ярый коллекционер бабочек. Он имеет огромную коллекцию, экспонаты которой собраны со всего мира. Будем считать, что в мире существует 2 000 000 000 видов бабочек.

Чтобы не запутаться, Андрей Сергеевич присвоил каждому виду уникальный номер. Нумерация видов бабочек начинается с единицы.

Теперь он хочет знать, есть ли бабочка с видом K в его коллекции, или же её придётся добывать, затрачивая уйму сил и денег.

Формат входного файла

В первой строке входного файла содержится единственное число N ($1 \leqslant N \leqslant 100\,000$) — количество видов бабочек в коллекции Андрея Сергеевича.

В следующей строке через пробел находятся N упорядоченных по возрастанию чисел — номера видов бабочек в коллекции.

Все виды бабочки в коллекции имеют различные номера.

В третьей строке файла записано число M ($1 \leqslant M \leqslant 100\,000$), количество видов бабочек, про которых Андрей Сергеевич хочет узнать, есть ли они у него в коллекции или же нет. В последней строке входного файла содержатся через пробел M чисел — номера видов бабочек, наличие которых необходимо проверить.

Формат выходного файла

Выходной файл должен содержать M строчек. Для каждого запроса выведите число "`YES`", если бабочка с данным номером содержится в коллекции, и "`NO`" — в противном случае.

Примеры

collect.in	collect.out
7 10 47 50 63 89 90 99 4 84 33 10 82	NO NO YES NO

Задача B. Коровы - в стойла

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

На прямой расположены стойла, в которые необходимо расставить коров так, чтобы минимальное растояние между коровами было как можно больше.

Формат входного файла

В первой строке вводятся числа N ($2 < N < 10\,001$) — количество стойл и K ($1 < K < N$) — количество коров. Во второй строке задаются N натуральных чисел в порядке возрастания — координаты стойл (координаты не превосходят 10^9).

Формат выходного файла

Выведите одно число — наибольшее возможное допустимое расстояние.

Примеры

cows.in	cows.out
5 3 1 2 3 100 1000	99

Задача C. 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 входного файла

Во входном файле записано два числа N и M ($0 < N \leqslant 100$, $0 \leqslant M \leqslant \frac{N(N-1)}{2}$). В следующих M строках записаны по два числа i и j ($1 \leqslant i, j \leqslant N$), которые означают, что перекрестки i и j соединены тоннелем.

Формат выходного файла

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 D. 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 входного файла

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

Формат выходного файла

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 E. 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 входного файла

В первой строке файла — число N — количество вершин графа ($1 \leqslant N \leqslant 100$). Во второй строке и далее — матрица смежности.

Формат выходного файла

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

Задача F. 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 входного файла

В первой строке входного файла содержится число N — количество вершин ($1 \leqslant N \leqslant 100$). 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.

Формат выходного файла

Выведите матрицу смежности ориентированного графа.

Примеры

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

Задача G. Транзитивность графа

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

Напомним, что граф называется транзитивным, если всегда из того, что вершины u и v соединены ребром и вершины v и w соединены ребром следует, что вершины u и w соединены ребром.

Проверьте, что заданный неориентированный граф является транзитивным.

Формат входного файла

В первой строке — числа N и M ($1 \leqslant N \leqslant 100$, $0 \leqslant M \leqslant \frac{N(N-1)}{2}$). Далее идет M строк — список ребер.

Формат выходного файла

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

Примеры

transitive.in	transitive.out
3 3 1 2 2 3 1 3	YES
3 2 1 2 1 3	NO

Задача H. 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 входного файла

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

Формат выходного файла

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	YES

Задача I. 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 входного файла

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

Формат выходного файла

Выведите количество ребер заданного графа.

Примеры

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

Задача J. 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 входного файла

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

Формат выходного файла

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

Примеры

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

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

Формат входного файла

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

Формат выходного файла

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 L. 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 входного файла

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

Формат выходного файла

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 порядке.

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 M. 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 ребром. Для заданного списком ребер графа проверьте, является ли он полным.

Формат входного файла

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

Формат выходного файла

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

Примеры

complete.in	complete.out
3 3 1 2 1 3 2 3	YES