Задача A. От списков смежности к матрице смежности

Имя входного файла:	`altom.in`
Имя выходного файла:	`altom.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Формат входных данных

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. 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.

Пример

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

Задача B. Подсчет количества ребер неориентированного графа

Имя входного файла:	`count.in`
Имя выходного файла:	`count.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

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

Формат входных данных

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 выходных данных

Выведите количество ребер заданного графа.

Пример

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

Задача C. Проверка на неориентированность

Имя входного файла:	`check.in`
Имя выходного файла:	`check.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

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 ($1 \leqslant N \leqslant 100$), 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 D. 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:	256 мегабайт

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

Формат входных данных

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 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 E. Истоки и стоки

Имя входного файла:	`source.in`
Имя выходного файла:	`source.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

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 ($1 \leqslant N \leqslant 100$), 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.

Пример

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 F. Компоненты связности

Имя входного файла:	`components.in`
Имя выходного файла:	`components.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Дан неориентированный граф. Требуется выделить компоненты связности в нем.

Формат входных данных

Первая строка входного файла содержит два натуральных числа n и m — количество вершин и ребер графа соответственно ($1 \leqslant n \leqslant 100\,000$, $0 \leqslant m \leqslant 200\,000$).

Следующие m строк содержат описание ребер по одному на строке. Ребро номер i описывается двумя натуральными числами b_i, e_i — номерами концов ребра ($1 \leqslant b_i, e_i \leqslant n$). Допускаются петли и параллельные ребра.

Формат выходных данных

В первой строке выходного файла выведите целое число k — количество компонент связности графа. Во второй строке выведите n натуральных чисел $a_1, a_1, \ldots, a_n$, не превосходящих k, где a_i — номер компоненты связности, которой принадлежит i-я вершина.

Примеры

components.in	components.out
3 1 1 2	2 1 1 2
4 2 1 3 2 4	2 1 2 1 2

Задача G. Поиск цикла

Имя входного файла:	`cycle.in`
Имя выходного файла:	`cycle.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Дан ориентированный невзвешенный граф. Необходимо определить есть ли в нём циклы, и если есть, то вывести любой из них.

Формат входных данных

В первой строке входного файла находятся два натуральных числа n и m ($1 \leqslant n \leqslant 100\,000, m \leqslant 100\,000$) — 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 выходных данных

Если в графе нет цикла, то вывести `NO`, иначе — `YES` и затем перечислить все вершины в порядке обхода цикла.

Пример

cycle.in	cycle.out
3 3 1 2 2 3 3 1	YES 2 3 1

Задача H. Лесопосадки

Имя входного файла:	`tree.in`
Имя выходного файла:	`tree.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Дан неориентированный невзвешенный граф. Необходимо определить, является ли он деревом.

Формат входных данных

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-ой строке на j-ом 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 выходных данных

Вывести «`YES`», если граф является деревом, «`NO`» иначе.

Примеры

tree.in	tree.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	NO
3 0 1 0 1 0 1 0 1 0	YES

Задача I. Топологическая сортировка

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Дан ориентированный невзвешенный граф. Необходимо его топологически отсортировать.

Формат входных данных

В первой строке входного файла даны два натуральных числа N и M ($1 \leqslant N \leqslant 100\,000$, $0 \leqslant M \leqslant 100\,000$) — 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 выходных данных

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

Пример

стандартный ввод	стандартный вывод
6 6 1 2 3 2 4 2 2 5 6 5 4 6	4 6 3 1 2 5

Задача J. Кратчайшие расстояния

Имя входного файла:	`mindist.in`
Имя выходного файла:	`mindist.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Дан ориентированный граф. Найдите расстояния от вершины x до всех остальных вершин графа.

Формат входных данных

В первой строке входного файла содержатся два натуральных числа N и x ($1 \leqslant N \leqslant 1\,000$, $1 \leqslant x \leqslant N$) — количество вершин в графе и стартовая вершина соответственно. 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.

Формат выходных данных

Выведите через пробел числа $d_1, d_2, \ldots d_n$, где d_i — это -1, если путей между x и i нет, и минимальное расстояние между x и i в противном случае.

Пример

mindist.in	mindist.out
6 5 0 1 1 0 0 0 1 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1 0 0 0 1 1 0 0 0 1 0 0 0 0	2 2 1 1 0 -1

Задача K. Обход в ширину

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

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

Формат входных данных

В первой строке входного файла содержатся 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 направленное 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

стандартный ввод	стандартный вывод
5 4 2 0 1 0 1 1 1 0 1 0 0 0 1 0 1 0 1 0 1 0 0 1 0 0 0 0	2

Задача L. Приключения шахматного коня

Имя входного файла:	`knight.in`
Имя выходного файла:	`knight.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

На шахматной доске $N \times N$ в клетке (x_1, y_1) стоит голодный шахматный конь. Он хочет попасть в клетку (x_2, y_2), где растет вкусная шахматная трава. Какое наименьшее количество ходов он должен для этого сделать?

Формат входных данных

На вход программы поступает пять чисел: N, x_1, y_1, x_2, y_2 ($5 \leqslant N \leqslant 20, 1 \leqslant x_1, y_1, x_2, y_2 \leqslant N$). Левая верхняя клетка доски имеет координаты $(1, 1)$, правая нижняя — (N, N).

Формат выходных данных

В первой строке выведите единственное число K - наименьшее необходимое число ходов коня. В каждой из следующих $K + 1$ строк должно быть записано 2 числа - координаты очередной клетки в пути коня.

Пример

knight.in	knight.out
5 1 1 3 2	2 1 1 3 2

Задача M. Числа

Имя входного файла:	`numbers.in`
Имя выходного файла:	`numbers.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Витя хочет придумать новую игру с числами. В этой игре от игроков требуется преобразовывать четырехзначные числа не содержащие нулей при помощи следующего разрешенного набора действий:

1. Можно увеличить первую цифру числа на 1, если она не равна 9.
2. Можно уменьшить последнюю цифру на 1, если она не равна 1.
3. Можно циклически сдвинуть все цифры на одну вправо.
4. Можно циклически сдвинуть все цифры на одну влево.

Например, применяя эти правила к числу 1234 можно получить числа 2234, 1233, 4123 и 2341 соответственно. Точные правила игры Витя пока не придумал, но пока его интересует вопрос, как получить из одного числа другое за минимальное количество операций.

Формат входных данных

Во входном файле содержится два различных четырехзначных числа, каждое из которых не содержит нулей.

Формат выходных данных

Программа должна вывести последовательность четырехзначных чисел, не содержащих нулей. Последовательность должна начинаться первым из данных чисел и заканчиваться вторым из данных чисел, каждое последующее число в последовательности должно быть получено из предыдущего числа применением одного из правил. Количество чисел в последовательности должно быть минимально возможным.

Пример

numbers.in	numbers.out
1234 4321	1234 2234 3234 4323 4322 4321

Задача N. Эвакуация

Имя входного файла:	`evacuation.in`
Имя выходного файла:	`evacuation.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Одна из Сверхсекретных организаций, чье название мы не имеем право разглашать, представляет собой сеть из N подземных бункеров, соединенных равными по длине туннелями, по которым из любого бункера можно добраться до любого другого (не обязательно напрямую). Связь с внешним миром осуществляется через специальные засекреченные выходы, которые расположены в некоторых из бункеров. Организации понадобилось составить план эвакуации персонала на случай экстренной ситуации. Для этого для каждого из бункеров необходимо узнать, сколько времени потребуется для того, чтобы добраться до ближайшего из выходов. Вам, как специалисту по таким задачам, поручено рассчитать необходимое время для каждого из бункеров по заданному описанию помещения Сверхсекретной организации. Для вашего же удобства бункеры занумерованы числами от 1 до N.

Формат входных данных

В первой строке записано число N, во второй — число K ($1 \leqslant N \leqslant 100\,000$, $1 \leqslant K \leqslant N$) — количество бункеров и количество выходов соответственно. Далее через пробел записаны K различных чисел от 1 до N, обозначающих номера бункеров, в которых расположены выходы. Потом идёт целое число M ($1 \leqslant M \leqslant 100\,000$) — количество туннелей. Далее вводятся M пар чисел — номера бункеров, соединенных туннелем. По каждому из туннелей можно двигаться в обе стороны. В организации не существует туннелей, ведущих из бункера в самого себя, зато может существовать более одного туннеля между парой бункеров.

Формат выходных данных

В первой строке выведите N чисел, разделённых пробелом — для каждого из бункеров минимальное время, необходимое чтобы добраться до выхода. Считайте, что время перемещения по одному туннелю равно 1. Во второй строке выведите N чисел — для каждого бункера номер ближайшего бункера с выходом, если таких несколько выведите бункер с наименьшим номером.

Пример

evacuation.in	evacuation.out
3 1 2 3 1 2 3 1 2 3	1 0 1 2 2 2