Задача A. Паросочетание

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

Двудольным графом называется неориентированный граф (V, E), $E \subseteq V \times V$ такой, что его множество вершин V можно разбить на два множества A и B, для которых $\forall (e_1, e_2) \in E$ $e_1 \in A$, $e_2 \in B$ и $A \cup B = V$, $A \cap B = \varnothing$.

Паросочетанием в двудольном графе называется любой набор его несмежных рёбер, то есть такой набор $S \subseteq E$, что для любых двух рёбер $e_1 = (u_1, v_1)$, $e_2 = (u_2, v_2)$ из S $u_1 \neq u_2$ и $v_1 \neq v_2$.

Ваша задача — найти максимальное паросочетание в двудольном графе, то есть паросочетание с максимально возможным числом рёбер.

Формат входных данных

В первой строке записаны два целых числа n и m ($1 \leqslant n, m \leqslant 250$), где n — число вершин в множестве A, а m — число вершин в B.

Далее следуют n строк с описаниями рёбер — i-я вершина из A описана в $(i+1)$-й строке файла. Каждая из этих строк содержит номера вершин из B, соединённых с i-й вершиной A. Гарантируется, что в графе нет кратных ребер. Вершины в A и B нумеруются независимо (с единицы). Список завершается числом 0.

Формат выходных данных

Первая строка выходного файла должна содержать одно целое число l — количество рёбер в максимальном паросочетании. Далее следуют l строк, в каждой из которых должны быть два целых числа u_j и v_j — концы рёбер паросочетания в A и B соотвественно.

Пример

pairs.in	pairs.out
2 2 1 2 0 2 0	2 1 1 2 2

Задача B. Минимальное контролирующее множество

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

Требуется построить в двудольном графе минимальное контролирующее множество, если дано максимальное паросочетание.

Формат входных данных

В первой строке файла даны два числа m и n ($1 \leqslant m, n \leqslant 4\,000$) — размеры долей. Каждая из следующих m строк содержит список ребер, выходящих из соответствующей вершины первой доли. Этот список начинается с числа K_i ($0 \leqslant K_i \leqslant n$) — количества ребер, после которого записаны вершины второй доли, соединенные с данной вершиной первой доли, в произвольном порядке. Сумма всех K_i во входном файле не превосходит 500 000. Последняя строка файла содержит некоторое максимальное паросочетание в этом графе — m чисел $0 \leqslant L_i \leqslant n$ — соответствующая i-й вершине первой доли вершина второй доли, или 0, если i-я вершина первой доли не входит в паросочетание.

Формат выходных данных

Первая строка содержит размер минимального контролирующего множества. Вторая строка содержит количество вершин первой доли S, после которого записаны S чисел — номера вершин первой доли, входящих в контролирующее множество, в возрастающем порядке. Третья строка содержит описание вершин второй доли в аналогичном формате.

Примеры

minimal.in	minimal.out
3 2 2 1 2 1 2 1 2 1 2 0	2 1 1 1 2

Задача C. Покрытие путями

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

Задан ориентированный ациклический граф. Требуется определить минимальное количество не пересекающихся по вершинам путей, покрывающих все вершины.

Формат входных данных

Первая строка входного файла содержит целые числа n и m — количества вершин и рёбер графа соответственно ($2 \leqslant n \leqslant 1\,000$, $0 \leqslant m \leqslant 10^5$). В следующих m строках содержатся по два натуральных числа — номера вершин u и v, которые соединяет ребро (u, v).

Формат выходных данных

В первой строке выходного файла выведите натуральное число k — минимальное количество путей, необходимых для покрытия всех вершин.

Пример

paths.in	paths.out
3 3 1 3 3 2 1 2	1

Задача D. День рождения

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

Митя знаком с m юношами и n девушками и хочет пригласить часть из них на свой день рождения. Ему известно, с какими девушками знаком каждый юноша, и с какими юношами знакома каждая девушка. Он хочет добиться того, чтобы каждый приглашённый был знаком со всеми приглашёнными противоположного пола, пригласив при этом максимально возможное число своих знакомых. Помогите ему это сделать!

Формат входных данных

Входной файл состоит из одного или нескольких наборов входных данных. В первой строке входного файла записано число наборов k ($1 \leqslant k \leqslant 20$). В последующих строках записаны сами наборы входных данных.

В первой строке каждого набора задаются числа $0 \leqslant m \leqslant 150$ и $0 \leqslant n \leqslant 150$. Далее следуют m строк, в каждой из которых записано одно или несколько чисел — номера девушек, с которыми знаком i-й юноша (каждый номер встречается не более одного раза). Строка завершается числом 0.

Формат выходных данных

Для каждого набора выведите четыре строки. В первой из них выведите максимальное число знакомых, которых сможет пригласить Митя. В следующей строке выведите количество юношей и количество девушек в максимальном наборе знакомых. Следующие две строки должны содержать номера приглашённых юношей и приглашённых девушек соответственно. Если максимальных наборов несколько, то выведите любой из них.

Примеры

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

Задача E. Паросочетание максимального веса

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

Дан двудольный граф. Количество вершин в левой и правой доле совпадает и равно n. У каждой вершины левой доли есть вес, i-й вершине соответствует вес w_i. Вес просочетания, ребрам которого инцидентны вершины левой доли $a_1, a_2, \ldots, a_k$ есть $\sqrt{\sum_{i=1}^{k} w_{a_i}^2}$. Требуется найти паросочетание максимального веса.

Формат входных данных

Первая строка входного файла содержит натуральное число n — количество вершин в обеих долях ($1 \leqslant n \leqslant 1000$). Вторая строка входного файла содержит n целых чисел $w_1, w_2, \ldots, w_n$ ($1 \leqslant w_i \leqslant 1000$). Следующие n строк содержат описания ребер, инцидентных соответствующей вершине левой доли. Формат описания: количество ребер, затем номера вершин правой доли, разделенные пробелом. Суммарное количество ребер не превосходит 200000.

Формат выходных данных

Выведите n чисел — для каждой вершины левой доли выведите номер вершины правой доли, с которой ее надо взять в паросочетание. Если вершина не входит в паросочетание, выведите 0.

Примеры

стандартный ввод	стандартный вывод
4 1 3 2 4 4 1 2 3 4 2 1 4 2 1 4 2 1 4	2 1 0 4

Задача F. Такси

Имя входного файла:	`taxi.in`
Имя выходного файла:	`taxi.out`
Ограничение по времени:	0.5 секунда
Ограничение по памяти:	256 мегабайт

Управлять службой такси — сосвем не простое дело. Помимо естественной необходимости централизованного управления машинами для того, чтобы обслуживать заказы по мере их поступления и как можно быстрее, нужно также планировать поездки для обслуживания тех клиентов, которые сделали заказы заранее.

В вашем распоряжении находится список заказов такси на следующий день. Вам необходимо минимимизировать число машин такси, необходимых чтобы выполнить все заказы.

Для простоты будем считать, что план города представляет собой квадратную решетку. Адрес в городе будем обозначать парой целых чисел: x-координатой и y-координатой. Время, необходимое для того, чтобы добраться из точки с адресом (a, b) в точку (c, d), равно $|a - c| + |b - d|$ минут. Машина такси может выполнить очередной заказ, либо если это первый ее заказ за день, либо она успевает приехать в начальную точку из предыдущей конечной хотя бы за минуту до указанного срока. Обратите внимание, что выполнение некоторых заказов может окончиться после полуночи.

Формат входных данных

В первой строке входного файла записано число заказов M ($0 < M < 500$). Последующие M строк описывают сами заказы, по одному в строке. Про каждый заказ указано время отправления в формате `hh:mm` (в интервале с `00:00` по `23:59`), координаты (a, b) точки отправления и координаты (c, d) точки назначения. Все координаты во входном файле неотрицательные и не превосходят 200. Заказы записаны упорядоченными по времени отправления.

Формат выходных данных

В выходной файл выведите единственное целое число — минимальное количество машин такси, необходимых для обслуживания всех заказов.

Примеры

taxi.in	taxi.out
2 08:00 10 11 9 16 08:07 9 16 10 11	1
2 08:00 10 11 9 16 08:06 9 16 10 11	2

Задача G. Охота на буйволов

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

Шаман охоты Алкстиминль и его n верных лучников увидели в прерии стадо из n буйволов. Алкстиминль спросил каждого охотника, которого из буйволов тот больше всего хотел бы застрелить.

Каждый дал ответ, после чего коварный Алкстиминль решил отдать приказ, согласно которому каждый лучник стрелял бы в одного буйвола, в каждого буйвола стрелял бы ровно один лучник, и никакой лучник не стрелял бы в выбранного им буйвола.

Предложите вариант приказа для Алкстиминля, либо сообщите шаману охоты, что это сделать невозможно.

Формат входных данных

Первая строка входного файла содержит натуральное число n ($1 \leqslant n \leqslant 100\,000$) — количество лучников в отряде Алкстиминля.

Вторая строка содержит n натуральных чисел от 1 до n — номера буйволов, выбранных 1-м, 2-м, ..., n-м лучником.

Формат выходных данных

Если приказ отдать невозможно, выведите -1. В противном случае выведите любой вариант приказа, который может отдать Алкстиминль — номера буйволов, в которых надлежит стрелять 1-му, 2-му, ..., n-му лучнику.

Примеры

buffalo.in	buffalo.out
3 1 1 2	2 3 1

Задача H. Толстые хоббиты

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

Ни один хоббит не в состоянии в одиночку противостоять полчищам Мордора... В последний поход против Мордора Гэндальф решил отправить N хоббитов из Шира. Но часть хоббитов наотрез отказалась, жалуясь на то, что другие хоббиты наверняка будут дразнить их толстыми. После опроса всех хоббитов оказалось, что любой хоббит отказывается принять участие в походе в том случае, если с ним в поход выступит хотя бы один хоббит с меньшим весом. К счастью для Средиземья, не все хоббиты знают свой точный вес. В Шире были всего одни весы чашечного типа, позволяющие для пары хоббитов определить, какой хоббит тяжелее. Некоторые пары хоббитов взвешивались на этих весах. Всем хоббитам известен результат всех взвешиваний. Гэндальф абсолютно уверен, что в Шире нет двух хоббитов одного веса. Он заинтересован в том, чтобы отряд состоял из наибольшего количества хоббитов. Однако найти наибольшее множество хоббитов, среди которых ни один не считает себя тяжелей другого, оказалось не так-то просто. Подскажите Гэндальфу, на сколько хоббитов он может рассчитывать. Помните при этом, что хоббиты умные существа и знают, что если Сэм тяжелее Пиппина, а Пиппин тяжелее Фродо, то Сэм и подавно будет тяжелее Фродо.

Формат входных данных

В первой строке дано целое число N – количество хоббитов ($2 \leqslant N \leqslant 100$). Все хоббиты пронумерованы целыми числами от 1 до N. В следующих N строках записана матрица размера $N \times N$. Если i-й и j-й хоббит взвешивались на чашечных весах и оказалось, что i-й хоббит тяжелее, то в i-й строке матрицы на j-й позиции стоит единица. Во всех остальных случаях в матрице стоят нули.

Формат выходных данных

В первой строке выведите размер наибольшего множества хоббитов, готового выступить в поход, во второй строке перечислите номера хоббитов из этого множества через пробел.

Примеры

hobbits.in	hobbits.out
2 0 1 0 0	1 2
3 0 0 0 0 0 0 0 0 0	3 1 2 3