Задача A. Конденсация графа. Light.

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	0.5 секунд
Ограничение по памяти:	256 мегабайт

Требуется найти количество ребер в конденсации ориентированного графа. Примечание: конденсация графа не содержит кратных ребер и петель.

Формат входных данных

Первая строка входного файла содержит два натуральных числа n и m – количество вершин и ребер графа соответственно ($n \leqslant 10\,000, m \leqslant 100\,000$). Следующие m строк содержат описание ребер, по одному на строке. Ребро номер i описывается двумя натуральными числами b_i, e_i – началом и концом ребра соответственно ($1 \leqslant b_i, e_i \leqslant n$). В графе могут присутствовать кратные ребра и петли.

Формат выходных данных

Первая строка выходного файла должна содержать одно число – количество ребер в конденсации графа.

Пример

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

Задача B. Конденсация графа

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Вам задан ориентированный граф с N вершинами и M ребрами ($1 \leqslant N \leqslant 200\,000$, $1 \leqslant M \leqslant 200\,000$). Найдите компоненты сильной связности заданного графа и топологически отсортируйте его конденсацию.

Формат входных данных

Граф задан во входном файле следующим образом: первая строка содержит числа N и M. Каждая из следующих M строк содержит описание ребра – два целых числа из диапазона от 1 до N – номера начала и конца ребра.

Формат выходных данных

На первой строке выведите число K – количество компонент сильной связности в заданном графе. На следующей строке выведите N чисел – для каждой вершины выведите номер компоненты сильной связности, которой принадлежит эта вершина. Компоненты сильной связности должны быть занумерованы таким образом, чтобы для любого ребра номер компоненты сильной связности его начала не превышал номера компоненты сильной связности его конца.

Пример

стандартный ввод	стандартный вывод
10 19 1 4 7 8 5 10 8 9 9 6 2 6 6 2 3 8 9 2 7 2 9 7 4 5 3 6 7 3 6 7 10 8 10 1 2 9 2 7	2 1 2 2 1 1 2 2 2 2 1

Задача C. Правильная скорая помощь

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

В связи с напряженной эпидемиологической обстановкой было решено пристраивать к уже существующим домам станции скорой помощи. Город задан в виде графа, в котором ребра — это односторонние дороги, а вершины — дома. Разумеется, во время вызова скорая может игнорировать ПДД (и даже направление движения), но вот возвращаться обратно по встречке уже не получится: больной уже под контролем врачей, да и рискованно это слишком.

Экономическая обстановка тоже не самая спокойная, поэтому требуется определить минимальное количество станций, которое нужно построить, чтобы скорая могла доехать обратно до станции от любого дома города.

Формат входных данных

В первой строке входного файла задано число n ($1 \leqslant n \leqslant 3000$) — количество домов. Во второй строке записано количество дорог m ($1 \leqslant m \leqslant 10^5$). Далее следует описание дорог в формате $a_i\, b_i$, означающее, что по i-й дороге разрешается движение от дома a_i к дому b_i ($1 \leqslant a_i, b_i \leqslant n$).

Формат выходных данных

Выведите одно число — минимальное количество станций, которое нужно построить, чтобы скорая могла доехать от любого дома до станции, соблюдая ПДД. Если к дому пристроена станция, то от этого дома до станции, очевидно, можно доехать.

Пример

стандартный ввод	стандартный вывод
3 3 1 2 2 3 1 3	1

Задача D. 2-SAT

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

Формулировка `2-SAT`: нужно подобрать значения n булевых переменных так, чтобы все m утверждений вида $(x_{i_1} = e_1) \vee (x_{i_2} = e_2)$ обратились в истину. В данной задаче вам гарантируется, что решение существует.

Формат входных данных

Входной файл состоит из одного или нескольких тестов.

Каждый тест описывается следующим образом. На первой строке число переменных n и число утверждений m. Каждая из следующих m строк содержит числа i_1, e_1, i_2, e_2, задает утверждение $(x_{i_1} = e_1) \vee (x_{i_2} = e_2)$ $(0 \leqslant i_j < n,\ 0 \leqslant e_j \leqslant 1)$. Ограничения: сумма всех n не больше 100 000, сумма всех m не больше 300 000.

Формат выходных данных

Для каждого теста выведите строку из n нулей и единиц — значения переменных. Если у данной задачи `2-SAT` есть несколько решений, выведите любое.

Пример

стандартный ввод	стандартный вывод
1 0 2 2 0 0 1 0 0 1 1 1 3 4 0 1 1 0 0 0 2 1 1 1 2 0 0 0 0 1	0 01 000

Задача E. Эйлеров путь

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

Дан неориентированный связный граф, не более трех вершин имеет нечетную степень. Требуется определить, существует ли в нем путь, проходящий по всем ребрам.

Если такой путь существует, необходимо его вывести.

Формат входных данных

Первая строка входного файла содержит натуральное число n — количество вершин графа ($1 \leqslant n \leqslant 100\,000$). Далее следуют n строк, задающих ребра. В i-й из этих строк находится число m_i — количество ребер, инцидентных вершине i. Далее следуют m_i натуральных чисел — номера вершин, в которые ведут ребро из i-й вершины.

Граф может содержать кратные ребра, но не содержит петель.

Граф содержит не более 300 000 ребер.

Формат выходных данных

Если решение существует, то в первую строку выходного файла выведите одно число k — количество ребер в искомом маршруте, а во вторую $k+1$ число — номера вершин в порядке их посещения.

Если решений нет, выведите в выходной файл одно число `-1`.

Если решений несколько, выведите любое.

Пример

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

Задача F. Студентам — бесплатно!

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Зал Большого галактического театра состоит из S рядов, по S мест в каждом ряду. Продажа билетов на каждый спектакль происходит по следующему принципу: первые $S^2 - N$ ценителей прекрасного приобретают билеты на любые места по их вкусу, а оставшиеся N кресел администрация бесплатно выделяет студентам, отдавая дань сложившимся традициям.

Во избежание обвинений в дискриминации по половому признаку, рассаживать студентов по этим N местам необходимо таким образом, чтобы:

- в каждом ряду количество девушек-студенток и количество юношей-студентов различалось бы не более чем на 1;
- на каждой «вертикали мест» (т. е. местах, имеющих один и тот же номер, но расположенных в разных рядах) количество девушек-студенток и количество юношей-студентов также различалось бы не более чем на 1.

Таким образом, после продажи билетов ценителям прекрасного билетёры должны распределить оставшиеся N кресел на женские и мужские с соблюдением этих правил.

Каждое место в зале определяется двумя числами от 1 до S — номером ряда и номером самого места в этом ряду. Студенческое кресло номер i расположено в a_i-м ряду и имеет в нём номер b_i. Поскольку ценители прекрасного могли занять совершенно любые места, числа a_i и b_i могут принимать любые значения от 1 до S. В частности, может оказаться так, что в каком-нибудь ряду не будет ни одного студенческого места.

Ради упрощения работы билетёров администрация обращается к вам с заданием написать программу, которая автоматизирует процесс распределения студенческих мест на мужские и женские.

Формат входных данных

Сначала вводятся два целых числа S и N ($1 \leqslant S \leqslant 100\,000$, $1 \leqslant N \leqslant \min\{100\,000, S^2\}$). Далее расположены N пар натуральных чисел (a_i, b_i), не превосходящих S. Гарантируется, что все места различные.

Формат выходных данных

Если искомого способа не существует, выведите `Impossible`. Иначе выведите единственную строку из N символов '`M`' (мужское) и '`W`' (женское). Символ на i-й позиции соответствует статусу i-го места в той же нумерации, в которой они были перечислены во входных данных.

Примеры

стандартный ввод	стандартный вывод
2 2 2 1 1 2	MW
3 5 1 2 2 3 1 3 2 1 1 1	WMWWM

Задача G. Экскурсия

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Группа из n человек решила поехать на экскурсию. В процессе экскурсии можно заехать в некоторые из m городов.

Экскурсовод попросила каждого человека высказать свои пожелания по поводу посещения городов. Каждый человек может про какие-то города заявить, что он хочет их непременно посетить, а про какие-то — что точно не хочет.

Группа всегда путешествует вместе. Если группа заезжает в город, то все люди, заявившие, что точно не хотят его посетить, расстраиваются. Если группа не заезжает в город, то расстраиваются все люди, которые заявили, что хотят его непременно посетить.

Экскурсовод понимает, что удовлетворить все пожелания не всегда можно. Она хочет составить план, чтобы каждый человек расстроился не более одного раза.

Помогите экскурсоводу справиться с этой нелегкой задачей и составьте план посещения городов или выясните, что это невозможно.

Формат входных данных

В первой строке входных данных содержатся три целых числа n, m, k — количество человек, количество городов и количество пожеланий ($1 \leqslant n, m, k \leqslant 100\,000$).

В каждой из последующих k строк содержатся по два целых числа a и b, обозначающих пожелания ($1 \leqslant a \leqslant n, 1 \leqslant |b| \leqslant m$). Если $b > 0$, то человек под номером a хочет посетить город под номером b. Если же $b < 0$, то человек под номером a не хочет посетить город под номером $-b$. Каждое пожелание указано во вводе не более одного раза, ни для кого из участников нет одновременно пожелания посетить и не посещать один и тот же город.

Формат выходных данных

Если решения не существует, то выведите -1.

Иначе, в первой строке выходных данных выведите единственное целое число k — количество городов, которые войдут в план.

Во второй строке выведите k целых чисел — номера городов, которые следует посетить. Номера городов можно выводить в любом порядке.

Если существует несколько возможных ответов, можно вывести любой из них. Обратите внимание, что не требуется искать максимальное или минимальное k, можно вывести любой корректный ответ.

Примеры

стандартный ввод	стандартный вывод
3 5 6 1 2 1 3 1 -4 2 3 2 4 2 5	3 2 3 5
3 3 6 1 -1 1 2 2 -2 2 3 3 -3 3 1	0

Задача H. Таня и пароль

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

Пока папа был на работе, маленькая девочка Таня решила поиграть с папиным паролем к секретной базе данных. Папин пароль представляет собой строку, состоящую из $n+2$ символов. Она выписала все возможные n трёхбуквенных подстрок пароля на бумажки, по одной на каждую бумажку, а сам пароль выкинула. Каждая трёхбуквенная подстрока была выписана на бумажки столько раз, сколько она встречалась в пароле. Таким образом, в итоге у Тани оказалось n бумажек.

Потом Таня поняла, что папа расстроится, если узнает о ее игре, и решила восстановить пароль или, по крайней мере, хотя бы какую-то строку, соответствующую получившемуся набору трёхбуквенных строк. Вам предстоит помочь ей в этой непростой задаче. Известно, что папин пароль состоял из строчных и заглавных букв латинского алфавита, а также из цифр. Строчные и заглавные буквы латинского алфавита считаются различными.

Формат входных данных

В первой строке следует целое число n ($1 \leqslant n \leqslant 2 \cdot 10^5$), количество трёхбуквенных подстрок, которые получились у Тани.

Следующие n строк каждая содержат по три буквы, образующие подстроку пароля папы. Каждый символ во вводе — строчная или заглавная буква латинского алфавита или цифра.

Формат выходных данных

Если во время игры Таня что-то напутала, и строк, соответствующих данному набору подстрок, не существует, то выведите «`NO`».

Если же возможно восстановить строку, соответствующую данному набору подстрок, то выведите «`YES`», а затем любой подходящий вариант пароля.

Примеры

стандартный ввод	стандартный вывод
5 aca aba aba cab bac	YES abacaba
4 abc bCb cb1 b13	NO
7 aaa aaa aaa aaa aaa aaa aaa	YES aaaaaaaaa

Задача I. Обновление дата-центров

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мегабайт

У компании BigData Inc. есть n дата-центров, пронумерованных от 1 до n, расположенных по всему миру. В этих дата-центрах хранятся данные клиентов компании (как можно догадаться из названия — большие данные!)

Основой предлагаемых компанией BigData Inc. услуг является гарантия возможности работы с пользовательскими данными даже при условии выхода какого-либо из дата-центров компании из доступности. Подобная гарантия достигается путём использования *двойной репликации* данных. Двойная репликация — это подход, при котором любые данные хранятся в двух идентичных копиях в двух различных дата-центрах.

Про каждого из m клиентов компании известны номера двух различных дата-центров $c_{i,1}$ и $c_{i,2}$, в которых хранятся его данные.

Для поддержания работоспособности дата-центра и безопасности данных программное обеспечение каждого дата-центра требует регулярного обновления. Релизный цикл в компании BigData Inc. составляет один день, то есть новая версия программного обеспечения выкладывается на каждый компьютер дата-центра каждый день.

Обновление дата-центра, состоящего из множества компьютеров, является сложной и длительной задачей, поэтому для каждого дата-центра выделен временной интервал длиной в час, в течение которого компьютеры дата-центра обновляются и, как следствие, могут быть недоступны. Будем считать, что в сутках h часов. Таким образом, для каждого дата-центра зафиксировано целое число u_j ($0 \leqslant u_j \leqslant h-1$), обозначающее номер часа в сутках, в течение которого j-й дата-центр недоступен в связи с плановым обновлением.

Из всего вышесказанного следует, что для любого клиента должны выполняться условия $u_{c_{i,1}} \neq u_{c_{i,2}}$, так как иначе во время одновременного обновления обоих дата-центров, компания будет не в состоянии обеспечить клиенту доступ к его данным.

В связи с переводом часов в разных странах и городах мира, время обновления в некоторых дата-центрах может сдвинуться на один час вперёд. Для подготовки к непредвиденным ситуациям руководство компании хочет провести учения, в ходе которых будет выбрано некоторое непустое подмножество дата-центров, и время обновления каждого из них будет сдвинуто на один час позже внутри суток (то есть, если $u_j = h - 1$, то новым часом обновления будет 0, иначе новым часом обновления станет $u_j + 1$). При этом учения не должны нарушать гарантии доступности, то есть, после смены графика обновления должно по-прежнему выполняться условие, что данные любого клиента доступны хотя бы в одном экземпляре в любой час.

Учения — полезное мероприятие, но трудоёмкое и затратное, поэтому руководство компании обратилось к вам за помощью в определении минимального по размеру непустого подходящего подмножества дата-центров, чтобы провести учения только на этом подмножестве.

Формат входных данных

В первой строке находятся три целых числа n, m и h ($2 \leqslant n \leqslant 100\,000$, $1 \leqslant m \leqslant 100\,000$, $2 \leqslant h \leqslant 100\,000$) — число дата-центров компании, число клиентов компании и количество часов в сутках.

Во второй строке вам даны n чисел $u_1, u_2, \ldots, u_n$ ($0 \leqslant u_j < h$), j-е из которых задаёт номер часа, в который происходит плановое обновление программного обеспечения на компьютерах дата-центра j.

Далее в m строках находятся пары чисел $c_{i,1}$ и $c_{i,2}$ ($1 \leqslant c_{i,1}, c_{i,2} \leqslant n$, $c_{i,1} \neq c_{i,2}$), задающие номера дата-центров, на которых находятся данные клиента i.

Гарантируется, что при заданном расписании обновлений в дата-центрах любому клиенту в любой момент доступна хотя бы одна копия его данных.

Формат выходных данных

В первой строке выведите минимальное количество дата-центров k ($1 \leqslant k \leqslant n$), которые должны затронуть учения, чтобы не потерять гарантию доступности. Во второй строке выведите k различных целых чисел — номера кластеров $x_1, x_2, \ldots, x_k$ ($1 \leqslant x_i \leqslant n$), на которых в рамках учений обновления станут проводиться на час позже. Номера кластеров можно выводить в любом порядке.

Если возможных ответов несколько, разрешается вывести любой из них. Гарантируется, что хотя бы один ответ, удовлетворяющий условиям задачи, существует.

Примеры

стандартный ввод	стандартный вывод
3 3 5 4 4 0 1 3 3 2 3 1	1 3
4 5 4 2 1 0 3 4 3 3 2 1 2 1 4 1 3	4 1 2 3 4

Замечание

Рассмотрим первый тест из условия. Приведённый ответ является единственным способом провести учения, затронув только один дата-центр. В таком сценарии третий сервер начинает обновляться в первый час дня, и никакие два сервера, хранящие данные одного и того же пользователя, не обновляются в один и тот же час.

С другой стороны, например, сдвинуть только время обновления первого сервера на один час вперёд нельзя — в таком случае данные пользователей 1 и 3 будут недоступны в течение нулевого часа.