Задача A. 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 4 3 2 1

Задача B. Точки сочленения

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

256 мегабайт

Дан неориентированный граф. Требуется найти все точки сочленения в нём.

Формат входных данных

Первая строка входного файла содержит два натуральных числа n и m — количества вершин и рёбер графа соответственно ($1 \leqslant n \leqslant 20\,000$, $1 \leqslant m \leqslant 200\,000$).

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

Формат выходных данных

Первая строка выходного файла должна содержать одно натуральное число b — количество точек сочленения в заданном графе. На следующей строке выведите b целых чисел — номера вершин, которые являются точками сочленения, в возрастающем порядке.

Пример

стандартный ввод	стандартный вывод
6 7 1 2 2 3 2 4 2 5 4 5 1 3 3 6	2 2 3

Задача C. Магнитные подушки

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

Город будущего застроен небоскребами, для передвижения между которыми и парковки транспорта многие тройки небоскребов соединены треугольной подушкой из однополярных магнитов. Каждая подушка соединяет ровно 3 небоскреба и вид сверху на нее представляет собой треугольник, с вершинами в небоскребах. Это позволяет беспрепятственно передвигаться между соответствующими небоскребами. Подушки можно делать на разных уровнях, поэтому один небоскреб может быть соединен различными подушками с парами других, причем два небоскреба могут соединять несколько подушек (как с разными третьими небоскребами, так и с одинаковым). Например, возможны две подушки на разных уровнях между небоскребами 1, 2 и 3, и, кроме того, магнитная подушка между 1, 2, 5.

Система магнитных подушек организована так, что с их помощью можно добираться от одного небоскреба, до любого другого в этом городе (с одной подушки на другую можно перемещаться внутри небоскреба), но поддержание каждой из них требует больших затрат энергии.

Требуется написать программу, которая определит, какие из магнитных подушек нельзя удалять из подушечной системы города, так как удаление даже только этой подушки может привести к тому, что найдутся небоскребы из которых теперь нельзя добраться до некоторых других небоскребов, и жителям станет очень грустно.

Формат входных данных

В первой строке входного файла находятся числа N и M — количество небоскребов в городе и количество работающих магнитных подушек соответственно ($3 \leqslant N \leqslant 100000$, $1 \leqslant M \leqslant 100000$). В каждой из следующих M строк через пробел записаны три числа — номера небоскребов, соединенных подушкой. Небоскребы пронумерованы от 1 до N. Гарантируется, что имеющиеся магнитные подушки позволяют перемещаться от одного небоскреба до любого другого.

Формат выходных данных

Выведите в выходной файл сначала количество тех магнитных подушек, отключение которых невозможно без нарушения сообщения в городе, а потом их номера. Нумерация должна соответствовать тому порядку, в котором подушки перечислены во входном файле. Нумерация начинается с единицы.

Примеры

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

Задача D. Минимизация мостов

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

Добавить в граф $G = \langle V, E \rangle$ (возможно несвязный, с петлями и кратными рёбрами) ровно одно ребро, так чтобы количество мостов в данном графе стало минимально возможным.

Напомним, что мостом в графе называется такое ребро, удаление которого увеличивает число компонент связности графа.

Формат входных данных

Первая строка входного файла содержит два натуральных числа n и m – количества вершин и рёбер графа соответственно ($1 \leqslant n \leqslant 200\,000$, $1 \leqslant m \leqslant 200\,000$).

Следующие m строк содержат описание рёбер по одному на строке. Ребро номер i описывается двумя натуральными числами v_i , u_i – номерами концов ребра ($1 \leqslant v_i, u_i \leqslant n$).

Формат выходных данных

Выведите наименьшее число мостов, которое можно получить добавлением ровно одного ребра.

Пример

стандартный ввод	стандартный вывод
6 7 1 2 2 3 3 4 1 3 4 5 4 6 5 6	0

Задача E. 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 F. 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 G. Размещение данных

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

Телекоммуникационная сеть крупной IT-компании содержит n серверов, пронумерованных от 1 до n. Некоторые пары серверов соединены двусторонними каналами связи, всего в сети m каналов. Гарантируется, что сеть серверов устроена таким образом, что по каналам связи можно передавать данные с любого сервера на любой другой сервер, возможно с использованием одного или нескольких промежуточных серверов.

Множество серверов A называется отказоустойчивым, если при недоступности любого канала связи выполнено следующее условие. Для любого не входящего в это множество сервера X существует способ передать данные по остальным каналам на сервер X хотя бы от одного сервера из множества A.

На рис. 1 показан пример сети и отказоустойчивого множества из серверов с номерами 1 и 4. Данные на сервер 2 можно передать следующим образом. При недоступности канала между серверами 1 и 2 — с сервера 4, при недоступности канала между серверами 2 и 3 — с сервера 1. На серверы 3 и 5 при недоступности любого канала связи можно по другим каналам передать данные с сервера 4.

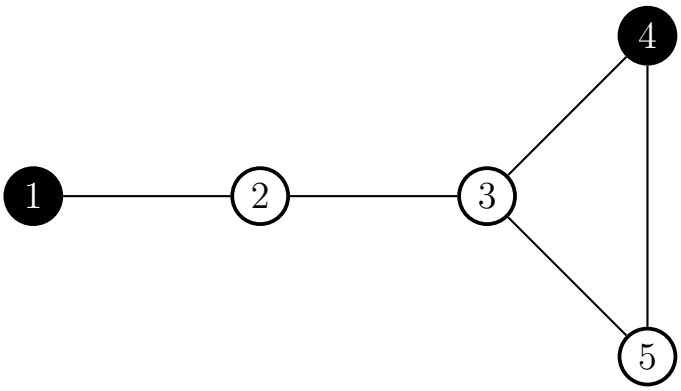

Рис. 1: Пример сети и отказоустойчивого множества серверов.

В рамках проекта группе разработчиков компании необходимо разместить свои данные в сети. Для повышения доступности данных и устойчивости к авариям разработчики хотят продублировать свои данные, разместив их одновременно на нескольких серверах, образующих отказоустойчивое множество. Чтобы минимизировать издержки, необходимо выбрать минимальное по количеству серверов отказоустойчивое множество. Кроме того, чтобы узнать, насколько гибко устроена сеть, необходимо подсчитать количество способов выбора такого множества, и поскольку это количество способов может быть большим, необходимо найти остаток от деления этого количества способов на число $10^9 + 7$.

Требуется написать программу, которая по заданному описанию сети определяет следующие числа: k — минимальное количество серверов в отказоустойчивом множестве серверов, c — остаток от деления количества способов выбора отказоустойчивого множества из k серверов на число $10^9 + 7$

Формат входных данных

Первая строка входного файла содержит целые числа n и m — количество серверов и количество каналов связи соответственно ($2 \leqslant n \leqslant 200\,000$, $1 \leqslant m \leqslant 200\,000$). Следующие m строк содержат по два целых числа и описывают каналы связи между серверами. Каждый канал связи задается двумя целыми числами: номерами серверов, которые он соединяет.

Гарантируется, что любые два сервера соединены напрямую не более чем одним каналом связи, никакой канал не соединяет сервер сам с собой, и для любой пары серверов существует способ

передачи данных с одного из них на другой, возможно с использованием одного или нескольких промежуточных серверов.

Формат выходных данных

Выведите два целых числа, разделенных пробелом: k — минимальное число серверов в отказоустойчивом множестве серверов, c — количество способов выбора отказоустойчивого множества из k серверов, взятое по модулю $10^9 + 7$.

Пример

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

Задача H. 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.

Задача I. Возвращение домой

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

С наступлением новых ограничений из-за пандемии в Лёшином вузе снова ввели дистанционное обучение, поэтому он решил наконец-то съездить домой к родителям. К сожалению, эта поездка всегда занимает у него много времени, потому что ему сначала приходится лететь на самолетё в Москву, а потом лететь ещё столько же до своего родного города.

Всего в стране n городов и m прямых двусторонних рейсов. Между парой городов может быть несколько двусторонних рейсов, может существовать двусторонний рейс между одним и тем же городом (некоторым людям просто нравится летать и читать книжку).

Сидя в самолёте Лёша подумал, как было бы хорошо если бы рейс из города A в город B, а после из города B в город C превратился бы в один прямой рейс между городами A и C, а рейсы AB и BC просто бы исчезли (сокращение AB означает рейс между городами A и B).

Так как Лёша любит решать различные задачи, а в самолете было скучно, он решил узнать можно ли, зная все рейсы в стране и последовательно используя операцию превращения существующих рейсов AB и BC в рейс AC с удалением рейсов AB и BC, оставить всего один рейс в стране. При мысленном выполнении такой операции Лёша может выбирать любые два различных рейса, в том числе два двусторонних рейса между одной и той же парой городов или рейс между городом и им самим. Если при выполнении операции должен исчезнуть рейс AB, а таких рейсов несколько, то рейс AB удаляется столько раз, сколько он используется в этой операции (то есть один или два).

Помогите Леше по заданным исходным рейсам между городами узнать можно ли при помощи применения произвольное число раз операции, описанной выше, оставить всего один рейс.

Формат входных данных

В первой строке через пробел записаны три числа n, m и p ($1 \leqslant n, m \leqslant 10^5$, $p \in \{0, 1\}$) — количество городов в стране, количество двусторонних рейсов между городами и нужно ли выводить последовательность операций, если можно оставить ровно один рейс.

В следующих m строках находятся пары вершин a_i и b_i ($1 \leqslant a_i, b_i \leqslant n$) — номера городов, между которыми проходит i-й рейс.

Формат выходных данных

В первой строке выведите «NO», если нельзя оставить ровно один рейс. В противном случае выведите «YES».

Если $p = 0$, то не нужно выводить последовательность операций при существовании ответа. Если же $p = 1$, то далее выведите $m - 1$ строк, в каждой строке через пробел должны быть записаны числа a_i, b_i и c_i ($1 \leqslant a_i, b_i, c_i \leqslant n$), означающие номера городов, для которых рейсы $a_i b_i$ и $b_i c_i$ превращаются в $a_i c_i$, а сами рейсы удаляются.

Примеры

стандартный ввод	стандартный вывод
3 2 1 1 2 2 3	YES 3 2 1
3 3 1 1 2 2 3 1 3	YES 1 2 3 1 3 1
3 3 0 1 2 2 3 1 3	YES
4 6 1 1 2 2 3 3 4 4 1 1 3 2 4	NO

Система оценки

Тесты к этой задаче состоят из пяти групп. Баллы за каждую группу ставятся только при прохождении всех тестов группы и всех тестов **необходимых** групп.

Группа	Баллы	Дополнительные ограничения		Треб. группы	Комментарий
		n, m	p		
0	0	–	–	–	Sample tests.
1	10	$n, m \leqslant 5$	–	0	
2	20	$n, m \leqslant 1000$	–	0, 1	
3	15	$n, m \leqslant 100\,000$	–	–	Каждый город имеет не более 2 рейсов (если рейс идет из города в него самого, то он сам по себе рассматривается как 2 рейса для этого города)
4	20	$n, m \leqslant 100\,000$	$p = 0$	–	Вам нужно только вывести «NO»/«YES», последовательность операций выводить не нужно даже в случае положительного ответа
5	35	–	–	0 – 4	

Задача J. Диаграммы Юнга выходят в интернет

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мегабайт

Рома и Сеня дали контест из 10 задач про диаграмму Юнга и теперь вынуждены скрываться от ДемидКомНадзора. Но они слишком любят диаграммы Юнга и хотят делиться новыми открытиями. Однако держаться вместе очень рискованно, так как в случае чего повяжут их обоих, поэтому они вынуждены общаться через Интернет. Но "обычный Интернет" полностью контролируется ДемидКомНадзором, поэтому они пользуются даркнетом.

В даркнете любое сообщение может проделать длинный запутанный путь до получателя через множество серверов. Более того, оно даже может проходить через один и тот же сервер несколько раз. За счет этого сообщение сложнее отследить.

Компьютер Ромы связан с сервером 1, а компьютер Сени — с сервером n.

ДемидКомНадзор хочет перехватить Ромино сообщение с диаграммой Юнга. Для этого ему необходимо взломать такой сервер, что сообщение, посланное Ромой, по любому пути к Сене пройдет через этот сервер **ровно один** раз.

Найдите все подходящие сервера.

Формат входных данных

В первой строке файла дано количество тестовых примеров t ($1 \leqslant t \leqslant 500$).

Каждый тестовый пример выглядит так: в первой строке даны два числа: n и m ($2 \leqslant n \leqslant 5 \cdot 10^5$, $0 \leqslant m \leqslant 10^6$), число серверов и число прямых соединений между серверами.

В каждой из последующих m строк содержится упорядоченная пара чисел a и b ($1 \leqslant a, b \leqslant n$), это означает, что с сервера a можно переслать сообщение напрямую на сервер b.

Гарантируется, что эти упорядоченные пары не повторяются внутри одного тестового примера.

Так же гарантируется, что и сумма по n, и сумма по m по всем тестовым примерам не превосходит 10^6.

Формат выходных данных

Для каждого тестового примера требуется вывести две строки: в первой число подходящих серверов, а во второй — номера этих серверов в порядке их следования на пути от a до b через пробел.

Пример

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

Задача K. Петербург?

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

– Это что за остановка –
Бологое иль Поповка? –
А с платформы говорят:
– Это город Ленинград.

«Вот какой рассеянный», Самуил Маршак

Пытаясь спастись от мира спортивного программирования, Алина сбежала на вокзал и уехала прочь на ночной электричке. Минуты медленно уплывали в даль, и уставшую девочку клонило в сон. Ей снился город-сказка, где не надо программировать, а можно гулять, мечтать и наслаждаться жизнью. Внезапно дождь из **интерактивных** задач разрушил эту идиллию.

Проснувшись и открыв окно, Алина задалась вопросом весьма философского свойства: «Где я?». С перрона потерявшейся девочке сообщили, что этот город, не похожий ни на что вокруг, представляет собой неориентированный граф на n вершинах и m ребрах. Сей невероятный факт, однако, нисколько не удивил Алину. Она давно мечтала побывать в одном таком городе — Петербурге. Его уникальной отличительной особенностью является то, что хотя бы **половина** его ребер — мосты (определение дано в конце условия). Так как никакие другие города Алине не интересны, она решила ограничиться расспросом находящихся на платформе эрудированных путешественников. Любой из их них может по данной вершине v сообщить любое ещё не названное ребро, исходящее из нее, или же заявить об отсутствии таковых.

Алина неуверена в своих силах, поэтому попросила вас помочь ей определить, попала ли она в Петербург. Так как её поезд скоро продолжит свой путь, задать больше $3n$ вопросов не получится.

Обратите внимание, что в графе **могут** присутствовать петли и кратные ребра.

Протокол взаимодействия

В первой строке стандартного потока ввода даны два целых числа n и m ($1 \leqslant n, m \leqslant 100\,000$) — число вершин и ребер в графе соответственно.

Для того, чтобы узнать очередное ребро, исходящее из u-й вершины ($1 \leqslant u \leqslant n$), нужно вывести «`?` u». После этого ваша программа на вход получит целое число v ($-2 \leqslant v \leqslant -1$ или $1 \leqslant v \leqslant n$) — $v = a + b - u$, если существует ребро ab, которое инцидентно вершине u и **ещё не было названо**, -1, если такого ребра не существует и -2, если вы превысили допустимое число запросов. В последнем случае ваша программа должна немедленно завершиться, в ином случае жюри не гарантирует корректность полученного вами вердикта.

Вам разрешается задать не более $3n$ вопросов.

Чтобы сообщить, что ответ найден, требуется вывести «`! Yes`» или «`! No`», в зависимости от того, является ли загаданный граф Петербургом. В случае положительного ответа выведите $\lceil \frac{m}{2} \rceil$ строк, по два целых числа u_i и v_i в каждой ($1 \leqslant u_i, v_i \leqslant n$), обозначающих, что ребро (u_i, v_i) является мостом. Любое ребро в приведенном списке должно встречаться не более одного раза (кратные ребра считаются различными).

Запрос на вывод ответа не входит в ограничение на $3n$ запросов.

Не забывайте сбрасывать буфер после каждого запроса. Например, на языке C++ надо использовать функцию `fflush(stdout)` или вызов `cout.flush()`, на Java вызов `System.out.flush()`, на Pascal `flush(output)` и `stdout.flush()` для языка Python.

Примеры

стандартный ввод	стандартный вывод
3 3 2 2 -1 3 -1 -1	 ? 3 ? 1 ? 2 ? 1 ? 1 ? 3 ! No
4 4 2 3 2 -1 4 -1 -1 -1	 ? 1 ? 2 ? 3 ? 1 ? 3 ? 3 ? 2 ? 4 ! Yes 1 2 3 4

Замечание

В условии в примере взаимодействия вводимые и выводимые данные расположены для удобства восприятия в хронологическом порядке, при реальном взаимодействии никакие «лишние» переводы строк возникать не должны.

Ввод-вывод в примерах демонстрирует пример взаимодействия вашей программы с проверяющей системой.

В первом примере был загадан граф на трех вершинах с ребрами $(1, 2)$, $(2, 3)$ и $(3, 1)$.

Во втором примере была загадан граф на четырех вершинах с ребрами $(1, 2)$, $(2, 3)$, $(3, 4)$ и $(2, 3)$.

Ребро, соединяющее вершины u и v, называется мостом, если после его удаления между вершинами u и v не существует пути.

Задача L. 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