

Задача А. 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 встрече 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 ($1 \leq n \leq 3000$) — количество домов. Во второй строке записано количество дорог m ($1 \leq m \leq 10^5$). 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 ($1 \leq a_i, b_i \leq n$).

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	1
3	
1 2	
2 3	
1 3	

Задача В. 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 ($1 \leq n \leq 20\,000$, $1 \leq m \leq 200\,000$).

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

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	2
1 2	2 3
2 3	
2 4	
2 5	
4 5	
1 3	
3 6	

Задача С. 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 добраться 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 ($3 \leq N \leq 100000$, $1 \leq M \leq 100000$). 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 ($1 \leq n \leq 200\,000$, $1 \leq m \leq 200\,000$).

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

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 Е. 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 ($0 \leq u_j \leq h-1$), 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 ($2 \leq n \leq 100\,000$, $1 \leq m \leq 100\,000$, $2 \leq h \leq 100\,000$) — число дата-центров компании, число клиентов компании и количество часов в сутках.

Во второй строке вам даны n чисел u_1, u_2, \dots, u_n ($0 \leq u_j < h$), 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 $c_{i,2}$ ($1 \leq c_{i,1}, c_{i,2} \leq n$, $c_{i,1} \neq c_{i,2}$), 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 ($1 \leq k \leq n$), которые должны затронуть учения, чтобы не потерять гарантию доступности. Во второй строке выведите k различных целых чисел — номера кластеров x_1, x_2, \dots, x_k ($1 \leq x_i \leq n$), 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.