Сложная

**Мосты и компоненты**

Дан неориентированный граф (не обязательно связный).  
Граф может содержать петли и кратные ребра. Выведите все компоненты реберной двусвязности графа (максимальные подмножества вершин, такие что подграф на них не теряет связность при удалении любого ребра).

ВХОДНЫЕ ДАННЫЕ

Первая строка содержит числа n и m (1⩽n⩽100000, 0⩽m⩽100000) -— количество вершин и ребер в графе. Следующие m строк задают ребра графа.

ВЫХОДНЫЕ ДАННЫЕ

В первой строке выведите количество компонент, в следующих за ней строках выведите сами компоненты, по одной на строку. Вершины в каждой компоненте должны идти в возрастающем порядке, компоненты нужно вывести в лексикографическом порядке. Компонента — вектор номеров своих вершин. Лексикографически сравниваются вектора.

|  |  |
| --- | --- |
| **ввод** | **вывод** |
| 3 2 1 2 2 3 | 3 1  2 3 |
| 7 8 1 5 5 6 1 6 5 4 4 3 4 2 3 2 7 2 | 3 1 5 6 2 3 4 7 |

Сложная

**Точки сочленения**

Дан неориентированный граф без петель а кратных рёбер. Требуется найти все точки сочленения в нем.

ВХОДНЫЕ ДАННЫЕ

Первая строка входного файла содержит два натуральных числа n и m -—  
количество вершин и ребер графа соответственно (n⩽20000, m⩽200000). Следующие m строк содержат описание ребер по одному на строке. Ребро номер  
i описывается двумя натуральными числами bi, ei -— номерами  
концов ребра (1⩽bi,ei⩽n).

ВЫХОДНЫЕ ДАННЫЕ

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.

|  |  |
| --- | --- |
| **ввод** | **вывод** |
| 9 12 1 2 2 3 4 5 2 6 2 7 8 9 1 3 1 4 1 5 6 7  3 8 3 9 | 3 1 2 3 |

Средняя

**Магнитные подушки**

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.

## **Входные данные**

В первой строке входного файла находятся числа N и M — количество небоскребов в городе и количество работающих магнитных подушек соответственно (3 ≤ N ≤ 100000, 1 ≤ M ≤ 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 воздушные подушки позволяют перемещаться от одного небоскреба до любого другого.

## **Выходные данные**

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.

|  |  |
| --- | --- |
| **ввод** | **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 |

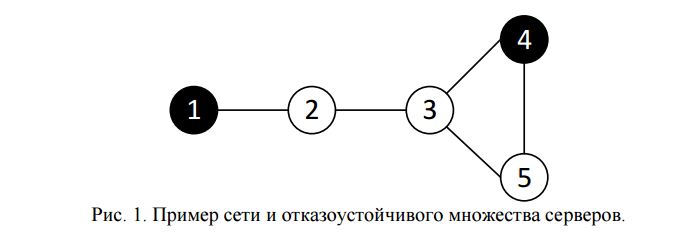
Олимпиадная

**Размещение данных**

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.



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

Входные данные

Первая строка входного файла содержит целые числа n и m — количество серверов и количество каналов связи соответственно (2 <= n <= 200000, 1 <= m <= 200000). 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.

Выходные данные

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

|  |  |
| --- | --- |
| **ввод** | **вывод** |
| 5 5  1 2  2 3  3 4  3 5  4 5 | 2 3 |