Задача A. 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 B. Компоненты реберной двусвязности

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Компонентой реберной двусвязности графа $\langle V, E\rangle$ называется подмножество вершин $S \subset V$, такое что для любых различных u и v из этого множества существует не менее двух реберно не пересекающихся путей из u в v.

Дан неориентированный граф. Требуется выделить компоненты реберной двусвязности в нем.

Формат входных данных

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

Следующие m строк содержат описание ребер 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

В первой строке выходного файла выведите целое число k — количество компонент реберной двусвязности графа.

Во второй строке выведите n натуральных чисел $a_1, a_2, \ldots, a_n$, не превосходящих k, где a_i — номер компоненты реберной двусвязности, которой принадлежит i-я вершина.

Компоненты требуется нумеровать в порядке возрастания минимального номера вершины, входящей в компоненту.

Пример

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

Задача C. 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 D. 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 E. Компоненты вершинной двусвязности

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Компонентой вершинной двусвязности графа $\langle V, E\rangle$ называется максимальный по включению подграф (состоящий из вершин и ребер), такой что любые два ребра из него лежат на вершинно простом цикле.

Дан неориентированный граф без петель. Требуется выделить компоненты вершинной двусвязности в нем.

Формат входных данных

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

Следующие m строк содержат описание ребер 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

В первой строке выходного файла выведите целое число k — количество компонент вершинной двусвязности графа.

Во второй строке выведите m натуральных чисел $a_1, a_2, \ldots, a_m$, не превосходящих k, где a_i — номер компоненты вершинной двусвязности, которой принадлежит i-е ребро. Ребра нумеруются с единицы в том порядке, в котором они заданы во входном файле.

Компоненты требуется нумеровать в порядке возрастания минимального номера ребра, входящего в компоненту.

Пример

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

Задача F. Мосты

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

Владения короля Джулиана расположены на n островах, пронумерованных от 1 до n. Некоторые пары островов соединены друг с другом мостами, по которым можно перемещаться в две стороны. Всего между островами есть m мостов. От любого острова можно добраться до любого другого, перемещаясь по мостам.

Будем называть мост мост *критическим*, если в случае обрушения этого моста будут существовать такие две острова, что от одного из них нельзя добраться до другого, перемещаясь по оставшимся мостам.

Король Джулиан очень беспокоится о безопасности и доступности сообщения в своих владениях. Он хочет построить дополнительные мосты между некоторыми парами островов так, чтобы между островами не осталось критических мостов. Так как король в то же время еще и экономный, он хочет выяснить, какое минимальное количество дополнительных мостов можно построить, чтобы выполнить данное требование.

Формат входных данных

В первой строке даны два целых числа n и m — количество островов и количество мостов между ними ($2 \leqslant n \leqslant 100\,000$, $1 \leqslant m \leqslant 200\,000$).

В следующих m строках дано по два целых числа a_i и b_i — номера островов, соединенных i-м мостом ($1 \leqslant a_i, b_i \leqslant n$, $a_i \neq b_i$).

Гарантируется, что от любого острова можно добраться до любого другого, перемещаясь по мостам.

Формат выходных данных

Выведите одно целое число — минимальное количество дополнительных мостов, которое нужно построить, чтобы между островами не было критических мостов.

Примеры

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

Задача G. Путешествие к примитиву

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

Так как у Морти осталось ещё много купонов на приключения, то он решил потратить один из них на путешествие в измерение примитивной речи. Но когда Рик и Морти прибыли в него, то поняли, что совсем не понимают местный язык. Но для Рика это была не проблема, и вот через несколько минут у героев есть устройство, помогающее говорить на примитивном языке.

Через некоторое время Морти обнаружил странную особенность примитивного языка — некоторые слова можно заменять другими. Тогда Морти решил помочь жителям этого измерения и создать такой язык, который может быть получен из исходного путем замены некоторых слов, при этом в нем должно содержаться наименьшее количество слов (ведь он считает, что чем меньше слов, тем проще будет язык учить).

Всего в текущем языке n слов, и известно, какие слова на какие можно заменять. Произвольным количеством замен приведите данный язык к языку, содержащему минимальное количество различных слов.

Формат входных данных

В первой строке даны два целых числа n и m — количество слов и возможных замен, соответственно ($1 \leqslant n, m \leqslant 2 \cdot 10^5$).

В следующих n строках даны слова в примитивном языке, длина одного слова не более 10, и гарантируется, что все слова различны.

В следующих m строках даны пары слов a_i и b_i, разделенные пробелом, означающие, что возможна замена слова a_i на слово b_i.

Формат выходных данных

В единственной строке выведите одно число — минимальное возможное количество слов в языке, который может получить Морти.

Пример

стандартный ввод	стандартный вывод
5 5 hello world first word second hello world world first world second second first word world	1

Замечание

В первом примере Морти может применить следующие преобразования, чтобы получить одно слово: `hello` → `world`, `word` → `world`, `world` → `second`, `second` → `first`. После таких преобразований в языке останется только одно слово: `first`.

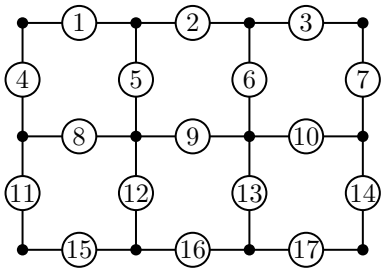
Задача H. Золотые монеты

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

Саша очень любит играть в игры на своем планшете. Недавно он скачал новую игру, которая называется «Золотые монеты».

Игра проходит на поле, имеющем вид сетки с тремя горизонтальными и четырьмя вертикальными рядами. Это поле представляет собой город: линии сетки являются дорогами, а узлы сетки — перекрестками. По дорогам можно перемещаться в обоих направлениях. В начале игры на каждой дороге расположено несколько золотых монет.

Пример игрового поля приведен на рисунке, перекрестки обозначены черными точками, дороги — отрезками, а число на дороге задает начальное количество золотых монет на ней.



Персонажем игры является бородатый электромонтер Томас. Сначала игрок может поместить Томаса на любой перекресток. Затем каждый ход игрок перемещает Томаса с перекрестка, на котором он находится, на соседний перекресток по одной из дорог, на которой еще лежит хотя бы одна монета.

Проходя по дороге, Томас забирает округленную вверх половину лежащих на ней монет. Таким образом, если на дороге лежит x монет, то, пройдя по этой дороге, Томас заберет $\lceil x/2 \rceil$ монет, а на дороге останется $\lfloor x/2 \rfloor$ монет. По дорогам, на которых монет уже нет, перемещать Томаса не разрешается.

Когда Томас оказывается в ситуации, что на всех соседних дорогах не осталось ни одной монеты, игра заканчивается. Очки игрока равны числу монет, которые собрал Томас.

Помогите Саше понять, какое максимальное количество монет он сможет собрать.

Формат входных данных

Входной файл состоит из пяти строк. Первая, третья и пятая строки содержат по три целых числа и описывают соответствующие горизонтальные улицы. Вторая и четвертая строки содержат по четыре целых числа и описывают соответствующие вертикальные улицы. Все числа неотрицательные и не превосходят 10^9.

Формат выходных данных

В единственной строке выходного файла выведите число s — максимальное количество монет, которые удастся собрать.

Примеры

стандартный ввод	стандартный вывод
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17	150
1 1 1 0 0 0 0 1 1 1 0 0 0 1 1 1 1	7

Задача I. Кодовый замок

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

На очередном задании Лэнс Стерлинг в очередной раз столкнулся с запертым сейфом. «Да сколько можно, снова взламывать», — подумал про себя супершпион.

Данный сейф закрывается на кодовый замок, который может быть представлен в виде таблицы $n \times n$. В этой таблице есть клетки трех типов:

- Пустые
- Клетки, содержащие *центральные элементы*
- Клетки, содержащие *поворотные ручки*

Причем, из-за конструкционных особеностей сейфа, в каждом столбце и каждой строке таблицы содержится не более одного центрального элемента.

Каждую из ручек можно ориентировать одним из двух способов: вертикально или горизонтально. Для того, чтобы замок открылся, ручки нужно ориентировать таким образом, чтобы от каждой ручки можно было добраться до центрального элемента, переходя только по ручкам ориентированным таким же образом, в направлении ориентации. Иными словами, если ручка ориентированна вертикально, должен существовать центральный элемент, находящийся на одной вертикали с этой ручкой, и все клетки между этим центральным элементом и этой ручкой должны быть заняты ручками, ориентированными вертикально. Аналогично для всех горизонтальных ручек.

Лэнсу нельзя терять ни минуты, поэтому он попросил помощи с открытием сейфа у вас. Помогите ему определить ориентации ручек, при которых сейф откроется, либо сообщите, что это невозможно.

Формат входных данных

В первой строке дано одно целое число n — размер таблицы ($1 \leqslant n \leqslant 500$).

Далее в n строках дано по n символов — описание таблицы. Символ «`.`» соответствует пустой клетке, символ «`O`» (заглавная английская буква O, ASCII код 79) — центральному элементу, а «`+`» — поворотной ручке.

Гарантируется, что в каждой строке и каждом столбце таблицы содержится не более одного центрального элемента.

Формат выходных данных

Если не существует способа ориентировать ручки, чтобы сейф открылся, выведите «`No`». Иначе, выведите «`Yes`», а в следующих n строках по n символов — описание таблицы, аналогичное описанию во входных данных. Ручки, которые нужно ориентировать вертикально, должны быть обозначены символами «`|`», а ручки, которые нужно ориентировать горизонтально — символами «`-`».

Если существует несколько различных ответов, выведите любой из них.

Примеры

стандартный ввод	стандартный вывод
3 O++ +.+ ++O	Yes O-| |.| --O
4 ..+. .... ..O. ..+.	No
3 .+. O++ .O.	No

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. Алексей и Иван пишут код

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

Алексей и Иван — программисты из компании Yango. Алексей специализируется на Golang и работает с очередью задач `YGO` , а Иван — эксперт по C++ и работает с очередью `YCPP`. В компании есть 26 проектов, обозначенных буквами от «`a`» до «`z`».

Каждый день в очереди YGO появляется одна приоритетная задача из какого-то проекта. В очереди YCPP тоже появляется одна приоритетная задача (возможно, из другого проекта).

Алексей планирует весь ближайший спринт, состоящий из N дней, брать и выполнять приоритетные задачи из своей очереди `YGO`. Иван планирует делать то же самое со своей очередью `YCPP`.

Каждый мог бы работать только со своей привычной очередью, но у Алексея и Ивана есть общий KPI — они хотят, чтобы после N дней работы у них была абсолютно одинаковая статистика по проектам (то есть по каждому проекту от «`a`» до «`z`» Иван должен выполнить столько же задач, сколько и Алексей). Хоть программисты предпочитают работать на своем языке, ради достижения общего KPI в некоторые дни они готовы поменяться очередями задач.

Помогите программистам определить, можно ли им в какие-то дни поменяться очередями (то есть Алексей возьмет задачу из `YCPP`, а Иван из `YGO`), чтобы после N дней их статистика по всем проектам была одинаковой. И если можно, то помогите найти хотя бы один подходящий набор дней для обмена очередями.

Формат входных данных

Первая строка входного файла содержит число $1 \leqslant N \leqslant 2 \cdot 10^5$ — длина спринта, в рамках которого берут задачи Иван и Алексей.

Следующая строка содержит строку длины N, состоящую из маленьких латинских букв. 26 проектов закодированы маленькими латинскими буквами. i-й символ строки содержит код проекта, задача из которого в i-й день будет приоритетной в очереди `YGO`.

Следующая строка содержит строку длины N. Эта строка содержит аналогичную информацию для очереди `YCPP`.

Формат выходных данных

Выведите -1, если не существует набора дней, в который программисты могут поменяться своими очередями и при этом получить полностью одинаковую статистику по проектам.

Если такой набор существует, в первой строке выведите единственное число K — количество дней, в которые программистам следует поменяться очередями ($0 \leqslant K \leqslant N$). В следующей строке через пробел выведите K различных чисел от 1 до N — номера этих дней (дни нумеруются с единицы).

Примеры

стандартный ввод	стандартный вывод
4 aabc bcdd	2 2 4
3 abc abd	-1
7 players parsley	0

Задача L. Петя и монеты

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	10 секунд
Ограничение по памяти:	512 мегабайт

Филателист Петя отправился в магазин, чтобы приобрести его заветную мечту — настоящую медную монету XVII века. В магазине продавец показал Пете n монет XVII века, имеющихся в наличии, каждая из которых сделана из одного из трех материалов: железа, меди или бронзы. К сожалению, время не пощадило монеты, поэтому они покрылись ржавчиной, и невозможно определить, какая монета из какого материала сделана. Чтобы Пете было проще найти нужную монету, продавец показал ему m пар монет таких, что монеты из пары сделаны из различных материалов. Кроме того, продавец сказал Пете, что ровно одна из n монет сделана из меди. Чтобы заполучить заветную монету, Петя решил приобрести все монеты, которые могут быть медными, исходя из информации, полученной им от продавца.

Более формально, Петя купит монету с номером p если и только если могло оказаться так, что все монеты с номерами, отличиными от p сделаны из железа и бронзы, а монета с номером p сделана из меди и при этом в каждой из m пар номеров монет, озвученных продавцом, монеты с соответствующими номерами сделаны из разных материалов.

Определите, какие монеты купит Петя.

Формат входных данных

В первой строке заданы два целых числа n и m ($1 \leqslant n \leqslant 3 \cdot 10^6, 0 \leqslant m \leqslant 3 \cdot 10^6$) — количество монет XVII века в магазине и количество пар монет, про которые продавец сказал Пете, что они сделаны из разных материалов.

В следующих m строках заданы пары целых чисел a_i и b_i ($1 \leqslant a_i, b_i \leqslant n, a_i \neq b_i$) — номера монет, которые сделаны из различных материалов. Все пары монет во вводе различны.

Формат выходных данных

Если продавец ошибся и ни одна монета не может быть медной, выведите «`0`» (без кавычек). Иначе в первой строке выведите одно целое число x — количество монет, которые купит Петя.

Во второй строке выведите x целых чисел c_i ($1 \leqslant c_i \leqslant n$) в порядке возрастания — номера монет, которые купит Петя.

Примеры

стандартный ввод	стандартный вывод
4 3 1 2 1 4 4 2	3 1 2 4
2 1 1 2	2 1 2
6 6 1 2 4 5 2 3 5 6 1 3 4 6	0
12 14 1 2 3 2 3 4 5 4 5 6 6 7 3 8 8 9 9 6 2 10 10 12 12 11 11 7 1 7	4 2 3 6 7

Замечание

В первом примере:

- Монета с номером 1 может быть медной, если, например, монеты с номерами 2 и 3 железные, а монета с номером 4 бронзовая.
- Монета с номером 2 может быть медной, если, например, монеты с номерами 1 и 3 железные, а монета с номером 4 бронзовая.
- Монета с номером 3 не может быть медной, так как среди монет с номерами 1, 2 и 4 не могут быть только железные и бронзовые.
- Монета с номером 4 может быть медной, если, например, монеты с номерами 2 и 3 железные, а монета с номером 1 бронзовая.

Во втором примере любая монета может быть медной, в случае, если, например, другая монета сделана из железа.

В третьем примере продавец ошибся и ни одна монета не может быть медной.

Задача M. X Aura

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	1024 мегабайта

Гора ICPC может быть представлена в виде сетки из R строк (нумеруемых от 1 до R) и C столбцов (нумеруемых от 1 до C). Ячейка, расположенная в строке r и столбце c, обозначается как (r, c) и имеет высоту $H_{r,c}$. Две ячейки являются соседними, если они имеют общую сторону. Формально, (r, c) соседствует с $(r-1, c)$, $(r+1, c)$, $(r, c-1)$ и $(r, c+1)$, если такие существуют.

Вы можете перемещаться только между соседними ячейками, и каждое движение связано с штрафом. При ауре нечетного положительного целого числа X, перемещение из ячейки с высотой h_1 в ячейку с высотой h_2 дает вам штраф $(h_1 - h_2)^X$. Обратите внимание, что штраф может быть отрицательным.

Вы хотите ответить на Q независимых сценариев. В каждом сценарии вы начинаете с начальной ячейки (R_s, C_s) и хотите добраться до целевой ячейки (R_f, C_f) с минимальным общим штрафом. В некоторых сценариях общий штраф может стать произвольно малым; такой сценарий называется *недопустимым*. Найдите минимальный общий штраф для перемещения от начальной ячейки до целевой ячейки или определите, является ли сценарий недопустимым.

Формат входных данных

Первая строка состоит из трех целых чисел $R\ C\ X$ ($1 \leqslant R, C \leqslant 1000$; $1 \leqslant X \leqslant 9$; X — нечетное целое число).

Каждая из следующих R строк состоит из строки H_r длиной C. Каждый символ в H_r — это число от 0 до 9. c-й символ H_r представляет высоту ячейки (r, c), или $H_{r,c}$.

Следующая строка состоит из целого числа Q ($1 \leqslant Q \leqslant 100000$).

Каждая из следующих Q строк состоит из четырех целых чисел $R_s\ C_s\ R_f\ C_f$ ($1 \leqslant R_s, R_f \leqslant R$; $1 \leqslant C_s, C_f \leqslant C$).

Формат выходных данных

Для каждого сценария выведите следующее в одной строке. Если сценарий недопустим, выведите `INVALID`. В противном случае выведите одно целое число, представляющее минимальный общий штраф для перемещения от начальной ячейки до целевой ячейки.

Примеры

стандартный ввод	стандартный вывод
3 4 1 3359 4294 3681 5 1 1 3 4 3 3 2 1 2 2 1 4 1 3 3 2 1 1 1 1	2 4 -7 -1 0
2 4 5 1908 2023 2 1 1 2 4 1 1 1 1	INVALID INVALID
3 3 9 135 357 579 2 3 3 1 1 2 2 2 2	2048 0

Замечание

Объяснение для примера входных/выходных данных #1

Для первого сценария одно из решений состоит в том, чтобы перемещаться следующим образом: $(1,1) \to (2,1) \to (3,1) \to (3,2) \to (3,3) \to (3,4)$. Общий штраф этого решения составляет $(3-4)^1 + (4-3)^1 + (3-6)^1 + (6-8)^1 + (8-1)^1 = 2$.

Объяснение для примера входных/выходных данных #2

Для первого сценария цикл $(1,1) \to (2,1) \to (2,2) \to (1,2) \to (1,1)$ имеет штраф $(1-2)^5 + (2-0)^5 + (0-9)^5 + (9-1)^5 = -26250$. Вы можете продолжать повторять этот цикл, чтобы сделать ваш общий штраф произвольно малым. Аналогично, для второго сценария вы можете сначала переместиться в $(1,1)$, а затем повторить тот же цикл.