Задача A. Лабиринт

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

В одном из уровней компьютерной игры вы попали в лабиринт, состоящий из n строк, каждая из которых содержит m клеток. Каждая клетка либо свободна, либо занята препятствием. Стартовая клетка находится в строке r и столбце c. За один шаг вы можете переместиться на одну клетку вверх, влево, вниз или вправо, если она не занята препятствием. Вы не можете перемещаться за границы лабиринта.

К сожалению, ваша клавиатура крайне близка к поломке, поэтому вы можете переместиться влево не более x раз и вправо не более y раз. При этом ограничений на перемещения вверх и вниз нет, поскольку клавиши, используемые для движения вверх и вниз, всё ещё в идеальном состоянии.

Теперь вы для каждой клетки поля решили установить, можно ли выбрать такую последовательность нажатий, которая приведёт вас из стартовой в эту клетку. Посчитайте, сколько клеток поля обладают таким свойством.

Формат входных данных

Первая строка содержит два целых числа n, m ($1 \leqslant n, m \leqslant 2000$) — количество строк и столбцов в лабиринте, соответственно.

Вторая строка содержит два целых числа r, c ($1 \leqslant r \leqslant n, 1 \leqslant c \leqslant m$) — номер строки и столбца, на пересечении которых расположена стартовая клетка.

Третья строка содержит два целых числа x, y ($0 \leqslant x, y \leqslant 10^9$) — максимальное количество перемещений влево и вправо, соответственно.

Следующие n строк содержат описание лабиринта. Каждая из этих строк имеет длину m и состоит только из символов '.' и '*'. В i-й строке j-й символ соответствует клетке лабиринта с номерами строки и столбца i и j, соответственно. Символ '.' соответствует свободной клетке лабиринта, а символ '*' — клетке с препятствием.

Гарантируется, что стартовая клетка не занята препятствием.

Формат выходных данных

Выведите одно число — количество клеток лабиринта, достижимых из стартовой, включая её саму.

Примеры

стандартный ввод	стандартный вывод
4 5 3 2 1 2 ..... .***. ...** *....	10
5 5 5 4 3 1 **... **.*. ...*. .***.	16

Задача B. Испорченный паркет

Имя входного файла:	`floor.in`
Имя выходного файла:	`floor.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Пол в некоторой комнате размером $M \times N$ замощен паркетом. При этом некоторые плитки паркета оказались испорчены. Петя решил сделать ремонт в этой комнате, заменив только испорченные клетки. Придя в магазин, он обнаружил, что паркетные плитки бывают двух типов — размера 1×2, которые стоят A рублей (немного подумав, Петя понял, что плитки 1×2 можно поворачивать на 90 градусов, получая тем самым плитки 2×1) и размера 1×1, которые стоят B рублей. Разрезать плитку размера 1×2 на две размера 1×1 Петя не может.

Определите, какая минимальная сумма денег нужна Пете, чтобы сделать ремонт.

Формат входных данных

Первая строка входного файла содержит 4 числа N, M, A, B ($1 \leqslant N, M \leqslant 300$, A, B — целые числа, по модулю не превосходящие 1000). Каждая из последующих N строк содержит по M символов: символ «`.`» (точка) обозначает неиспорченную плитку паркета, а символ «`*`» (звездочка) — испорченную. В конце строк могут идти незначащие пробелы. В конце файла могут быть пустые строки.

Формат выходных данных

В выходной файл выведите одно число — минимальную сумму денег, имея которую можно заменить испорченные паркетины (и только их).

Примеры

floor.in	floor.out
2 3 3 2 .** .*.	5

Задача C. Такси

Имя входного файла:	`taxi.in`
Имя выходного файла:	`taxi.out`
Ограничение по времени:	0.5 секунда
Ограничение по памяти:	256 мегабайт

Управлять службой такси — сосвем не простое дело. Помимо естественной необходимости централизованного управления машинами для того, чтобы обслуживать заказы по мере их поступления и как можно быстрее, нужно также планировать поездки для обслуживания тех клиентов, которые сделали заказы заранее.

В вашем распоряжении находится список заказов такси на следующий день. Вам необходимо минимимизировать число машин такси, необходимых чтобы выполнить все заказы.

Для простоты будем считать, что план города представляет собой квадратную решетку. Адрес в городе будем обозначать парой целых чисел: x-координатой и y-координатой. Время, необходимое для того, чтобы добраться из точки с адресом (a, b) в точку (c, d), равно $|a - c| + |b - d|$ минут. Машина такси может выполнить очередной заказ, либо если это первый ее заказ за день, либо она успевает приехать в начальную точку из предыдущей конечной хотя бы за минуту до указанного срока. Обратите внимание, что выполнение некоторых заказов может окончиться после полуночи.

Формат входных данных

В первой строке входного файла записано число заказов M ($0 < M < 500$). Последующие M строк описывают сами заказы, по одному в строке. Про каждый заказ указано время отправления в формате `hh:mm` (в интервале с `00:00` по `23:59`), координаты (a, b) точки отправления и координаты (c, d) точки назначения. Все координаты во входном файле неотрицательные и не превосходят 200. Заказы записаны упорядоченными по времени отправления.

Формат выходных данных

В выходной файл выведите единственное целое число — минимальное количество машин такси, необходимых для обслуживания всех заказов.

Примеры

taxi.in	taxi.out
2 08:00 10 11 9 16 08:07 9 16 10 11	1
2 08:00 10 11 9 16 08:06 9 16 10 11	2

Задача D. Минимальное контролирующее множество

Имя входного файла:	`min-dominating-set.in`
Имя выходного файла:	`min-dominating-set.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Множество вершин V' графа $G = (V, E)$, содержащееся в V, называется контролирующим, если каждое ребро графа инцидентно хотя бы одной вершине из V. Тривиальным примером контролирующего множества является множество всех вершин V. Ваша задача состоит в том, чтобы в данном двудольном графе найти какое-либо контролирующее множество вершин минимальной мощности.

Формат входных данных

В первой строке входного файла содержатся два числа n и m — количесто вершин в первой и второй долях ($1 \leqslant n, m \leqslant 200$). Далее до конца файла в каждой строке содержатся номера смежных вершин из первой и второй доли. В графе не бывает кратных рёбер.

Формат выходных данных

В выходной файл в первой строке выведите мощность минимального контролирующего множества. Во второй строке выведите через пробел количества вершин из первой и второй долей, входящих в контролирующее множество. В третьей строке через пробел выведите номера вершин первой доли, входящих в контролирующее множество, а в четвертой — номера вершин второй доли, входящих в это множество

Примеры

min-dominating-set.in	min-dominating-set.out
6 6 1 1 2 5 3 4 4 4 5 2 5 3 6 1 6 6	5 4 1 1 2 5 6 4
6 6 1 1 1 2 2 2 3 2 3 3 4 3 4 4 4 5 4 6 5 2 6 3	4 2 2 1 4 2 3

Задача E. RMQ

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

Есть массив из N целых чисел и M запросов вида: найдите минимум на отрезке с концами l_i, r_i.

Формат входных данных

Входной файл содержит T наборов тестовых данных. Каждый набор тестовых данных задаётся числами N, M, A, B ($1 \leqslant N \leqslant 25\,000, 1 \leqslant A, B \leqslant 10^9$), где N — размер массива, M — число запросов.

Массив и запросы нужно получить следующим образом: выпишем последовательность чисел $C_i = (A \cdot i + B) \bmod 2^{32}$.

Элементы последовательности с номерами от 1 до N — элементы массива. Элементы последовательности с номерами от $N + 1$ до $N + 2 \cdot M$ взятые по модулю N образуют M пар чисел, которые являются границами отрезков запросов. Ввод заканчивается числами 0 0 0 0. Массив индексируется с нуля.

Сумма N по всем наборам тестовых данных не превосходит 10^8. Сумма M по всем наборам тестовых данных не превосходит $2 \cdot 10^7$.

Формат выходных данных

Для каждого набора тестовых данных выведите сумму по всем запросам.

Примеры

стандартный ввод	стандартный вывод
10 10 955379886 619166003 0 0 0 0	7671393960

Замечание

Массив:

```
1574545889 2529925775 3485305661 145718251 1101098137 2056478023 3011857909
3967237795 627650385 1583030271
```

Запросы:

```
7 3
3 9
5 1
7 7
3 9
5 5
1 7
3 9
9 5
1 7
```

Задача F. Паросочетание максимального веса

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

Дан двудольный граф. Количество вершин в левой и правой доле совпадает и равно n. У каждой вершины левой доли есть вес, i-й вершине соответствует вес w_i. Вес просочетания, ребрам которого инцидентны вершины левой доли $a_1, a_2, \ldots, a_k$ есть $\sqrt{\sum_{i=1}^{k} w_{a_i}^2}$. Требуется найти паросочетание максимального веса.

Формат входных данных

Первая строка входного файла содержит натуральное число n — количество вершин в обеих долях ($1 \leqslant n \leqslant 1000$). Вторая строка входного файла содержит n целых чисел $w_1, w_2, \ldots, w_n$ ($1 \leqslant w_i \leqslant 1000$). Следующие n строк содержат описания ребер, инцидентных соответствующей вершине левой доли. Формат описания: количество ребер, затем номера вершин правой доли, разделенные пробелом. Суммарное количество ребер не превосходит 200000.

Формат выходных данных

Выведите n чисел — для каждой вершины левой доли выведите номер вершины правой доли, с которой ее надо взять в паросочетание. Если вершина не входит в паросочетание, выведите 0.

Примеры

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

Задача G. Граф

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	0.75 секунд
Ограничение по памяти:	256 мегабайт

Дан неориентированный граф, в котором каждое ребро покрашено либо в чёрный цвет, либо в красный цвет.

Ваша задача присвоить каждой вершине по вещественному числу таким образом, что:

- сумма чисел на обеих концах каждого чёрного ребра равна 1;
- сумма чисел на обеих концах каждого красного ребра равна 2;
- сумма модулей всех присвоенных чисел наименьшая возможная.

Если это не возможно, выведите, что не существует такого комплекта чисел

Формат входных данных

Первая строка содержит два целых числа N ($1 \leqslant N \leqslant 100\,000$) и M ($0 \leqslant M \leqslant 200\,000$) — количество вершин и рёбер, соответственно. Вершины пронумерованы последовательными числами от 1 до N.

Следующие M строк содержат описание рёбер. Каждая строка содержит три целых числа a, b и c, которые обозначают, что между вершинами a и b ($1 \leqslant a, b \leqslant N$) есть ребро цвета c (1 обозначает чёрное, 2 обозначает красное).

Формат выходных данных

Если решение существует, выведите в первой строке слово «`YES`» и во второй строке выведите N чисел. Для каждого i ($1 \leqslant i \leqslant N$), i-тое из этих чисел должно равняться числу, присвоенному вершине с номером i.

Вывод должен соответствовать следующим ограничениям точности:

- сумма чисел на концах каждого ребра должна отличаться от нужной суммы для этого ребра меньше, чем на 10^{-6};
- сумма модулей всех присвоенных чисел отличается от наименьшего возможного меньше, чем на 10^{-6}.

Если существует несколько решений, выведите любое из них. Если решения не существует, выведите одно слово «`NO`».

Примеры

стандартный ввод	стандартный вывод
4 4 1 2 1 2 3 2 1 3 2 3 4 1	YES 0.5 0.5 1.5 -0.5
2 1 1 2 1	YES 1.0 0.0
3 2 1 2 2 2 3 2	YES 0.0 2.0 0.0
3 4 1 2 2 2 2 1 2 1 1 1 2 2	NO

Задача H. День рождения

Имя входного файла:	`birthday.in`
Имя выходного файла:	`birthday.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Митя знаком с m юношами и n девушками и хочет пригласить часть из них на свой день рождения. Ему известно, с какими девушками знаком каждый юноша, и с какими юношами знакома каждая девушка. Он хочет добиться того, чтобы каждый приглашённый был знаком со всеми приглашёнными противоположного пола, пригласив при этом максимально возможное число своих знакомых. Помогите ему это сделать!

Формат входных данных

Входной файл состоит из одного или нескольких наборов входных данных. В первой строке входного файла записано число наборов k ($1 \leqslant k \leqslant 20$). В последующих строках записаны сами наборы входных данных.

В первой строке каждого набора задаются числа $0 \leqslant m \leqslant 150$ и $0 \leqslant n \leqslant 150$. Далее следуют m строк, в каждой из которых записано одно или несколько чисел — номера девушек, с которыми знаком i-й юноша (каждый номер встречается не более одного раза). Строка завершается числом 0.

Формат выходных данных

Для каждого набора выведите четыре строки. В первой из них выведите максимальное число знакомых, которых сможет пригласить Митя. В следующей строке выведите количество юношей и количество девушек в максимальном наборе знакомых. Следующие две строки должны содержать номера приглашённых юношей и приглашённых девушек соответственно. Если максимальных наборов несколько, то выведите любой из них.

Примеры

birthday.in	birthday.out
2 2 2 1 2 0 1 2 0 3 2 1 2 0 2 0 1 2 0	4 2 2 1 2 1 2 4 2 2 1 3 1 2

Задача I. Шарады

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1.5 секунд
Ограничение по памяти:	64 мегабайта

В ещё не изведанной части вселенной есть планета, на которой живут одни математики. На этой планете живут N математиков, каждый — в своём городе. Никакие два города не соединены дорогами, потому что математики могут общаться онлайн, оставляя комментарии о научных трудах друг друга.

Всё шло тихо и спокойно, пока один математик не решил написать научную работу со своего мобильного телефона. Автоисправление в телефоне заменило «очевидно» на «шарада». Не перечитав свою работу, математик так и опубликовал её. Совсем скоро об игре в шарады узнали все математики планеты, и им захотелось собраться и поиграть всем вместе. Поэтому в скором времени началась постройка дорог между городами. Строительство дорог будет идти M дней в соответствии со следующим расписанием: в первый день строятся дороги между всеми парами городов, у номеров которых наибольший общий делитель равен M. Во второй день строятся дороги между всеми парами городов, наибольший делитель номеров которых равен $M - 1$. И так далее до M-го дня, в который дороги строятся между всеми парами городов с взаимно простыми номерами. Говоря более формально, в i-й день (нумеруя дни с единицы) дороги строятся между всеми такими парами городов A и B, что НОД$(A, B) = M + 1 - i$.

Математики очень заняты постройкой дорог, поэтому они просят вас помочь определить минимальное число дней с начала строительства, через которое данная пара математиков сможет встретиться, чтобы поиграть в шарады.

Формат входных данных

В первой строке даны три целых целых положительных числа N, M и Q ($1 \leqslant N, Q \leqslant 100\,000$, $1 \leqslant M \leqslant N$) - количество городов, длительность строительства дорог и количество запросов соответственно.

В следующих Q строках вводятся по два целых числа A и B ($1 \leqslant A, B \leqslant N$) — номера городов двух математиков, которым интересно, через сколько дней они смогут встретиться (добраться из одного город в другой, проехав по уже построенным дорогам).

Формат выходных данных

На каждый из Q запросов выведите ответы — Q чисел, каждое в отдельной строке.

Система оценки

Программы, правильно работающие при $N \leqslant 1000$, будут оцениваться в 40 баллов.

Примеры

стандартный ввод	стандартный вывод
8 3 3 2 5 3 6 4 8	3 1 2
25 6 1 20 9	4
9999 2222 2 1025 2405 3154 8949	1980 2160

Замечание

Пояснение к первому тесту:

В первый день строится дорога $(3, 6)$. Поэтому ответ на второй запрос 1. На второй день строятся дороги $(2, 4)$, $(2, 6)$, $(2, 8)$, $(4, 6)$ и $(6, 8)$. Города 4 и 8 теперь связаны (можно добраться из первого во второй используя город 6). На третий день строятся дороги между взаимно простыми городами, поэтому города 2 и 5 оказываются соединены.

Пояснение ко второму тесту:

На второй день строится дорога $(20, 15)$, на четвертый день — дорога $(15, 9)$. Таким образом, начиная с четвёртого дня, города, 20 и 9 связаны (через город 15).

Задача J. Декартово

Имя входного файла:	`cartesius.in`
Имя выходного файла:	`cartesius.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Государство Иксово состоит из N_x городов, некоторые пары которых связаны дорогами с двусторонним движением. Каждая дорога имеет свою длину. Всего межгородских дорог в стране M_x, при чем известно, что из каждого города Иксевщины можно доехать по дорогам до каждого другого города этой страны. Города Иксово пронуменрованы натуральными числами от 1 до N_x .

Государство Игреково состоит из N_y городов, некоторые пары которых связаны дорогами с двусторонним движением. Каждая дорога имеет свою длину. Всего межгородских дорог в стране M_y, при чем известно, что из каждого города Игреково можно доехать по дорогам до каждого другого города этой страны. Города Игреково пронуменрованы натуральными числами от 1 до N_y .

Страна Декартово состоит из $N = N_x \cdot N_y$ городов: каждому городу Декартово во взаимно однозначное соответствие можно поставить пару городов-побратимов (x, y), где x — город Иксово, а y — город Игреково. Некоторые пары городов Декартово также соединены дорогами с двусторонним движением. Дорог в стране ровно $M = N_x \cdot M_y + N_y \cdot M_x$. При этом дорога между городами (x_1, y_1) и (x_2, y_2) существует только в одном из таких двух случаев:

1. Если $x_1 = x_2 = x$, а между городами y_1 и y_2 Игреково проложена дорога. При этом длина дороги между городами (x, y_1) и (x, y_2) Декартово равно длине дороги между городами y_1 и y_2 Игреково.

2. Если $y_1 = y_2 = y$, а между городами x_1 и x_2 Иксевщены проложена дорога. При этом длина дороги между городами (x_1, y) и (x_2, y) Декартово равно длине дороги между городами x_1 и x_2 Иксево. Города разных государств между собой дорогами не соединены.

Некоторые дороги Декартовщины требуется закрыть. Ваша задача — определить, дороги какой наименьшей суммарной длины можно оставить в Декартовщине, чтобы из любого ее города все еще можно было попасть в любой другой.

Формат входных данных

Первая строка содержит натуральные числа N_x и M_x ($1 \leqslant N_x, M_x \leqslant 5 \cdot 10^4$) — количество городов и дорог в Иксово. В последующих M_x строках описаны дороги Иксово: в каждой строке по три числа, где первые два задают номера разных городов, соеденненных дорогой, а третье есть длина соответствующей дороги (натуральное число, которое не превышает 10^7).

В следующей строке входного файла указаны натуральные числа N_y и M_y ($1 \leqslant N_y, M_y \leqslant 5 \cdot 10^4$) — количество городов и дорог в Игреково. Последующие M_y строк содержат описание дорог Игреково; формат данных и ограничения соответствуют описанным выше.

Формат выходных данных

Выходной файл должен содержать единственно целое число — ответ на вопрос подзадачи.

Примеры

cartesius.in	cartesius.out
3 2 2 1 15 3 1 14 3 2 2 1 15 3 2 15	117

Задача K. Северус Снейп

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

Армия Пожирателей Смерти наступает на Хогвартс. Тёмным лордом были назначены два её руководителя — Люциус Малфой и Антонин Долохов.

- Люциус руководит армией днём и водит армию по дорогам.
- Антонин руководит армией ночью и совершает манёвры по секретным тропам.

Каждый лидер перед маршем спрашивает дорогу у Северуса Снейпа. Северус, будучи ОЙ СПОЙЛЕР тайным агентом Дамблдора, хочет задержать наступление тёмных сил.

Карта дорог известна Люциусу. Аналогично, карта секретных троп известна Антонину. Поэтому Северусу не удастся их так просто обмануть — он должен каждый раз выбрать переход так, что минимальное расстояние между Хогвартсом и войском по соответствующей карте строго уменьшилось.

Вы знаете карту дорог и троп вместе с их длинами. Помогите Северусу как можно дольше (желательно, бесконечно) вести армию на Хогвартс.

Формат входных данных

В первой строке вводятся три целых числа n ($2 \leqslant n \leqslant 1000$), s и t ($s \neq t; 1 \leqslant s, t \leqslant n$) — количество вершин, номер вершины текущего расположения армии и номер вершины с Хогвартсом.

Далее идут описания карты Люциуса и карты Антонина.

Первая строка описания карты содержит число m — количество дорог или троп соответственно ($1 \leqslant M \leqslant 10^5$). Каждая из следующих m строк содержит 3 целых числа a, b и w — описывающих дорогу/тропу между вершинами a и b с указанием длины w ($1 \leqslant w \leqslant 10^6$).

Формат выходных данных

Выведите общую длину пути (вдоль дорог и троп), который Северус заставит пройти армию. Если он может заставить армию ходить вечно, то выведите -1.

Гарантируется, что из любой вершины армия Пожирателей Смерти может дойти до Хогвартса.

Примеры

стандартный ввод	стандартный вывод
5 1 5 5 1 2 2 1 4 2 2 3 1 3 4 1 5 3 1 4 1 2 2 2 4 2 2 3 1 2 5 2	-1
3 1 3 4 1 2 10 2 3 10 1 3 20 2 3 30 4 2 1 10 1 3 10 1 1 10 2 3 10	20

Задача L. Кукушки

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

Британские учёные решили заняться орнитологией и понаблюдать за жизнью необычных кукушек. Для этого они вырастили дерево и построили на нём n гнёзд, в каждом из которых живёт кукушка. Наблюдение за деревом состоит в том, что в некоторые моменты времени учёные оценивают, можно ли подложить определённое яйцо в гнездо к некоторой кукушке или нет.

Каждое яйцо может вынашиваться только в двух определённых гнёздах. Каждое яйцо задаётся неупорядоченной парой различных чисел (x, y). Яйцо (x, y) может вынашиваться в любом из гнёзд x и y и не может вынашиваться в других гнёздах. Обратите внимание, яйцо (x, y) не отличается от яйца (y, x).

Теперь опишем процесс подкладывания яйца в имеющиеся гнезда: пусть учёные хотят подложить яйцо (x, y) в гнездо x. Если в гнезде x нет яйца, то яйцо (x, y) просто остаётся в этом гнезде, и процесс на данном шаге завершается. Если же в гнезде x лежит какое-то яйцо (x, p), то кукушка кладёт яйцо (x, y) в данное гнездо, а яйцо (x, p) пытается подложить в гнездо p аналогичным образом, и процесс продолжается.

Вам предлагается отвечать на вопросы учёных. Всего есть три типа вопросов:

1. (Теоретический) Закончится ли процесс, если подложить яйцо (x, y) в гнездо x? Так как вопрос чисто теоретический, оно **не добавляется** на самом деле, и состояние гнёзд не меняется.

2. (Практический) Закончится ли процесс, если подложить яйцо (x, y) в гнездо x? Если процесс закончится, то яйцо **добавляется** в реальности согласно описанному процессу.

3. (Теоретический) Сколько существует **упорядоченных** пар различных чисел (x, y), таких что яйцо (x, y) можно подложить в гнездо x с учётом имеющихся в гнёздах яиц? При этом для каждого яйца ответ определяется независимо от других добавляемых яиц.

Формат входных данных

В первой строке вводятся три целых числа n, m, q, ($2 \leqslant n \leqslant 200\,000$, $0 \leqslant m \leqslant n$, $1 \leqslant q \leqslant 600\,000$), где n — количество гнёзд на дереве, m —количество яиц, которые учёные уже положили, q — количество вопросов, которые задают учёные.

В каждой из m последующих строк следуют по два числа x_i, y_i, означающих, что в гнезде x_i лежит яйцо (x_i, y_i). Гарантируется, что все x_i различны и что $x_i \neq y_i$ для всех i.

В следующих q строках описаны вопросы учёных. Вопросы даны в том порядке, в котором на них требуется отвечать. Первое число t_j в строке описывает тип вопроса.

Если $t_j = 1$ или $t_j = 2$, то далее идут два различных числа x_j и y_j, описывающих яйцо, которое фигурирует в соответствующем вопросе.

Если $t_j = 1$, то яйцо не требуется добавлять в текущую расстановку.

Если $t_j = 2$, то яйцо требуется добавить, если процесс добавления потребует конечного числа перекладываний.

Если $t_j = 3$, то требуется определить количество упорядоченных пар (x, y), таких что яйцо (x, y) можно добавить в гнездо x с тем, чтобы процесс когда-нибудь завершился. В реальности никакие яйца в расстановку не добавляются.

Формат выходных данных

Для каждого вопроса первого и второго типа выведите единственное слово «`Yes`» или «`No`» в зависимости от того, закончится ли процесс перекладывания.

Для каждого запроса третьего типа выведите количество искомых упорядоченных пар.

Пример

стандартный ввод	стандартный вывод
5 3 8 1 2 5 1 2 4 1 1 2 3 2 1 2 3 2 4 2 2 5 3 3 1 4 5	Yes 20 Yes 8 No Yes 0 No

Замечание

Изначальное расположение яиц в тесте из условия такое: в первом гнезде лежит яйцо $(1, 2)$, во втором — $(2, 4)$, в пятом — $(5, 1)$, а в третьем и четвёртом яиц нет.

Яйцо $(1, 2)$ добавить можно, несмотря на то что подобное яйцо на дереве уже есть, это приведёт к перекладыванию имеющегося яйца $(1, 2)$ в другое гнездо.

Также в начальную конфигурацию можно добавить любое из 10 яиц, существующих для дерева с пятью гнёздами, и каждое яйцо можно положить в любое из двух гнёзд, ему отвечающих, и для любого из добавляемых яиц и гнёзд это потребует конечное количество шагов. Таким образом, ответ на второй запрос — 20.

В результате следующего запроса яйцо $(1, 2)$ будет добавлено реально, и распределение яиц будет таким: в первом гнезде лежит яйцо $(1, 2)$, во втором — также $(1, 2)$, в четвёртом — $(2, 4)$, в пятом $(5, 1)$.

Теперь уже можно добавить только яйца $(1, 3)$, $(2, 3)$, $(4, 3)$ и $(5, 3)$, причём по-прежнему любое яйцо можно положить в каждое из двух упомянутых на нём гнёзд, поэтому ответ на запрос — 8.

Яйцо $(4, 2)$ добавить на дерево нельзя, поэтому состояние гнёзд не изменится.

Для добавления яйца $(5, 3)$ понадобится 5 перекладываний яиц, а после этого никакое новое яйцо за конечное количество шагов добавить уже нельзя.

Задача M. Рекурсивная схема

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

В схеме рекурсивной микросхемы имеется N точек контактов, причем некоторые пары контактов соединены напрямую проводами. Кроме того, имеется всего S подсистем внутри схемы, каждая из которых является точной копией рассматриваемой схемы.

В схеме цепи есть три типа контактов:

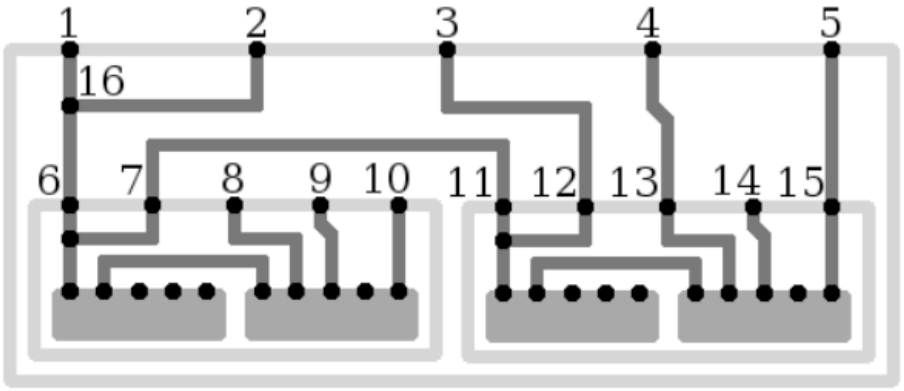
1. Входные контакты цепи (K контактов). Это единственные контакты, соединяющие цепь со внешними проводящими путями.
2. Входные контакты вложенных подсхем ($S \cdot K$ контактов).
3. Вспомогательные контакты.

Все эти контакты могут быть связаны друг с другом проводами без каких-либо ограничений.

Сигналы распространяются по проводам. Когда сигнал достигает контакта, он может следовать по любому проводу, связанному с этим контактом. Если внешний сигнал достигает входной контакт подсхемы, он может войти в подсхему и двигаться дальше по ее проводам. Если внутренний сигнал достигает входной контакт подсхемы, он может выйти из подсхемы (если есть провод снаружи, и если внешняя цепь сама является подсхемой другой цепи).

Рассмотрим самую внешнюю схему. Определите, связаны ли два контакта путями. Контакты связаны путём, если сигнал может пройти по проводам от одного контакта от другого, возможно входя в ряд различных подсхем конечное количество раз.

Помимо факта подключения, в некоторых группах тестов от вас будет требоваться выяснить, насколько глубоко сигнал должен попасть в подсхемы, чтобы достичь одного контакта от другого. Внешняя цепь имеет глубину вложения 0; для её подсхем глубина вложения равна 1, а их подсхемы, в свою очередь, имеют глубину вложения 2 и т. д. Для произвольного пути сигнала, критической глубиной называется самая глубокая подсхема, через которую проходит путь. Определить минимальное значение критической глубины для пути между двумя заданными входными контактами внешней цепи.



Формат входных данных

Первая строка содержит пять целых чисел: N — количество контактов в схеме, K — количество входных контактов цепи, S — количество подсхем в цепи, M — число проводов в схеме цепи, T — номер группы тестов ($1 \leqslant K \leqslant 100\,000$, $0 \leqslant S \leqslant 1\,000$, $K \cdot (S+1) \leqslant N \leqslant 100\,000$, $0 \leqslant M \leqslant 100\,000$).

Следующие M строк определяют провода в схеме цепи. Каждый провод определяется двумя целыми числами a и b — номера контактов, напрямую связанных этим проводом ($1 \leqslant a \neq b \leqslant N$).

Контакты в схеме пронумерованы в порядке от 1 до N. Входные контакты пронумерованы от 1 до K. Входные контакты подсистемы t пронумерованы от $t \cdot K + 1$ до $t \cdot K + K$ (для $1 \leqslant t \leqslant S$).

j-й входной контакт на схеме t-й цепи является $(t \cdot K + j)$-ым контактом на схеме внешней схемы. Остальные контакты, если таковые существуют, являются вспомогательными.

Следующая строка содержит целое число Q — количество запросов ($1 \leqslant Q \leqslant 100\,000$). Каждый из остальные Q строк содержат один запрос, который нуждается в ответе. Каждый запрос определяется двумя целыми числами u и v — номера входных контактов внешней цепи ($1 \leqslant u \neq v \leqslant K$).

Формат выходных данных

В выходном файле выведите Q целых чисел, по одному числу в строке. i-е число должно быть ответом к i-му запросу: глубина вложения, необходимая для перехода от одного из входных контактов к другому. Если нет пути между двумя входными контактами, выведите число -1 вместо значения глубины.

В некоторых группах вам не надо выяснять, глубину сигнала. В этом случае для i-го запропса выведите -1 если от одного контакта нельзя добраться до другого и любое неотрицательное число, если путь между этими двумя контактами существует.

В тестах из условия требуется узнать глубину.

Пример

стандартный ввод	стандартный вывод
16 5 2 7 0 1 16 2 16 6 16 7 11 3 12 4 13 5 15 4 1 2 2 3 3 4 4 5	0 1 2 -1