Задача A. В бухгалтерии опять всё перепутали

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

Лупа и Пупа пошли получать зарплату. Но в бухгалтерии опять всё перепутали. Лупа получил зарплату за Пупу, а Пупа ...

Пупа не хочет получать за Лупу и хочет доказать бухгалтерии, что она не права.

Пупа работает в крупной компании «MST Inc.», занимающейся информационным сопровождением «Всеберляндской олимпиады школьников по информатике». В компании «MST Inc.» работает n сотрудников, причём у каждого из них, кроме самой «MST», есть ровно один непосредственный начальник и несколько (возможно ноль) непосредственных подчинённых.

Всеми начальниками сотрудника компании «MST Inc.» называется множество, состоящее из его непосредственного начальника и множества начальников его непосредственного начальника. Известно, что у каждого сотрудника кроме самой «MST», «MST» входит в множество начальников этого сотрудника.

Множеством подчинённых у сотрудника называется множество, состоящее из него самого и множеств подчинённых у всех непосредственных подчинённых данного сотрудника. В частности, все сотрудники входят в множество подчинённых у «MST».

Каждый месяц каждому сотруднику начисляется зарплата, причём немаленькая, ведь иначе ни один сотрудник не согласился бы работать с «MST». Известно, что в нулевой месяц работы организации, каждому сотруднику заплатили по c_i бурлей. В качестве поощрения сотрудников «MST» придумала следующее правило: В каждый из следующих m месяцев берётся сотрудник с номером a_i и берётся число s_i — сумма зарплат всех сотрудников во множестве его начальников и подчинённых (включая его самого). Если это число оказывалось слишком большим, s_i берётся по модулю $10^9 + 7$. После этого берётся сотрудник с номером b_i, и к зарплате всех сотрудников, входящих во множество его начальников и подчинённых (включая его самого) прибавляется число s_i. С учётом этого изменения платится зарплата в i-й месяц и пересчитывается зарплата в следующие месяцы.

Вернёмся к Пупе. Пупа хочет показать бухгалтерии компании «MST Inc.» что она всё перепутала, а для этого ему надо узнать, сколько же ему должны были заплатить в каждый из месяцев с нулевого по m-й. К сожалению, в гениальной системе поощрения, разработанной «MST», не может разобраться никто. Поэтому эту задачу поручили вам.

Формат входных данных

В первой строке входных данных даны 2 числа n и m ($1 \leqslant n, m \leqslant 10^5$) — число сотрудников компании «MST Inc.» и последний день, когда выплачивалась зарплата Пупе.

Во второй строке записано $n - 1$ число. i-е из них — номер непосредственного начальника сотрудника номер i (i принимает значения от 1 до $n - 1$). При этом «MST» имеет номер 0 и не имеет непосредственного начальника. Пупа имеет номер $n - 1$.

В третьей строке записано n чисел c_i ($1 \leqslant c_i \leqslant 10^9$) — зарплата i-го сотрудника в нулевой день.

В каждой из следующих m строк записано по 2 числа a_i и b_i ($0 \leqslant a_i, b_i \leqslant n-1$) — номер человека, на основе которого происходит поощрение и номер человека, к подчинённым и начальникам которого поощрение применяется (более подробно описано в условии).

Формат выходных данных

В единственной строке выведите $m + 1$ число — зарплату Пупы в каждый из дней с 0-го по m-й. Напоминаем, что Пупа имеет номер $n - 1$. Обратите внимание, что зарплата **не считается** по модулю $10^9 + 7$.

Примеры

стандартный ввод	стандартный вывод
3 3 0 0 1 1 1 0 0 2 1 1 2	1 4 4 28
4 3 0 1 1 0 1 0 0 0 1 1 3 2 3	0 1 6 20

Замечание

Пояснение к первому примеру:

В первый день к зарплате каждого сотрудника прибавилось 3 бурля и зарплаты стали соответственно $4, 4, 4$.

Во второй день к зарплате сотрудников с номерами $0, 1$ прибавилось по 8 бурлей и зарплаты стали соответственно $12, 12, 4$.

Во третий день к зарплате сотрудников с номерами $0, 2$ прибавилось по 24 бурля и зарплаты стали соответственно $36, 12, 28$.

Задача B. Найти количество

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	0.6 секунд
Ограничение по памяти:	256 мегабайт

Дано подвешенное дерево из n вершин с корнем в 1. Необходимо обработать q запросов (v_i, h_i): найти количество вершин, лежащих в поддереве v_i, расстояние до которых от v_i равно h_i. Расстояние между вершинами — минимальное количество рёбер в пути между ними.

Формат входных данных

Первая строка содержит n ($1 \leqslant n \leqslant 3 \cdot 10^5$).
Следующая строка содержит $n - 1$ число $p_2, \ldots p_n$ ($1 \leqslant p_i \leqslant i$). p_i — отец вершины i.
Следующая строка содержит число q ($1 \leqslant q \leqslant 3 \cdot 10^5$).
Следующие q строк содержат числа v_i, h_i ($1 \leqslant v_i \leqslant n$, $1 \leqslant h_i \leqslant n$).

Формат выходных данных

Для каждого запроса выведите ответ.

Пример

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

Задача C. Ближайший лист

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	4 секунды
Ограничение по памяти:	512 мегабайт

Определим эйлеров обход дерева (связного неориентированного графа без циклов) следующим образом: рассмотрим рекурсивный алгоритм поиска в глубину, который обходит вершины дерева и нумерует вершины в том порядке, в котором их посещает, при этом учитывается только первое посещение каждой вершины. Данная функция стартует из вершины с номером 1, а затем рекурсивно вызывается от всех вершин, которые соединены ребром с текущей и ещё не посещены, в порядке возрастания номеров вершин. Формально данную функцию можно описать так:

```
next_id = 1
id = массив длины n, заполненный -1
visited = массив длины n, заполненный false

function dfs(v):
    visited[v] = true
    id[v] = next_id
    next_id += 1
    for to по соседям v в порядке возрастания:
        if not visited[to]:
            dfs(to)
```

Дано взвешенное дерево, вершины которого пронумеровали в порядке эйлерова обхода целыми числами от 1 до n при помощи алгоритма, описанного выше.

Назовём листом вершину дерева, соединённую ребром ровно с одной другой вершиной. В данном вам дереве вершина 1 не является листом. Расстоянием между двумя вершинами дерева назовём сумму весов рёбер на единственном простом пути между ними.

Требуется ответить на q запросов следующего вида: по заданным числам v, l и r сообщить кратчайшее расстояние от вершины v до одного из листов дерева, имеющего номер от l до r (включительно).

Формат входных данных

В первой строке даны два целых числа n и q ($3 \leqslant n \leqslant 500\,000, 1 \leqslant q \leqslant 500\,000$) — количество вершин в дереве и количество запросов соответственно.

Следующие $n-1$ строк задают рёбра дерева: $(i-1)$-я строка содержит два целых числа p_i и w_i ($1 \leqslant p_i < i, 1 \leqslant w_i \leqslant 10^9$), обозначающие ребро между вершинами p_i и i с весом w_i.

Гарантируется, что заданные рёбра задают дерево, вершины которого пронумерованы в порядке эйлерова обхода, и что вершина с номером 1 не является листом.

Следующие q строк содержат описания запросов. Каждая из них содержит три целых числа v_i, l_i, r_i ($1 \leqslant v_i \leqslant n, 1 \leqslant l_i \leqslant r_i \leqslant n$), обозначающие параметры запроса, описанные в условии. Гарантируется, что существует хотя бы один лист с номером x такой, что $l_i \leqslant x \leqslant r_i$.

Формат выходных данных

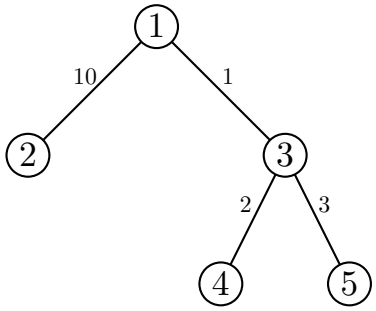
Выведите q чисел — ответы на запросы в порядке, в котором они заданы во входных данных.

Примеры

стандартный ввод	стандартный вывод
5 3 1 10 1 1 3 2 3 3 1 1 5 5 4 5 4 1 2	3 0 13
5 3 1 1000000000 2 1000000000 1 1000000000 1 1000000000 3 4 5 2 1 5 2 4 5	3000000000 1000000000 2000000000
11 8 1 7 2 1 1 20 1 2 5 6 6 2 6 3 5 1 9 10 9 11 5 1 11 1 1 4 9 4 8 6 1 4 9 7 11 9 10 11 8 1 11 11 4 5	8 8 9 16 9 10 0 34

Замечание

В первом примере дерево выглядит так:



В первом запросе ближайший к вершине 1 лист имеет номер 4. Расстояние до него равно 3. Во втором запросе ближайшим к вершине 5 листом является вершина с номером 5, расстояние до которой равно 0. В третьем примере ближайшим к вершине 4 листом является вершина с номером 4, однако она не попадает в отрезок вершин $[1, 2]$ запроса. Единственным листом с номером, попадающим в отрезок $[1, 2]$ является вершина с номером 2, расстояние до которой от вершины 4 равно 13.

Задача D. Дерево

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1.5 секунд
Ограничение по памяти:	256 мегабайт

Дано дерево из n вершин и q запросов.

Каждый запрос начинается с трех целых чисел k, m и r, и продолжается k вершинами дерева $a_1, a_2, \ldots, a_k$. Чтобы ответить на запрос, предположите, что дерево подвешено за вершину r. Рассмотрим разбиения данных k вершин на **не более чем** m групп так, что выполняются следующие условия:

- Каждая вершина принадлежит ровно одной группе, каждая группа содержит хотя бы одну вершину.
- Ни в одной группе нет двух вершин таких, что одна является предком (не обязательно непосредственным) другой.

Выведите количество различных таких разбиений по модулю $10^9 + 7$ для каждого запроса.

Формат входных данных

Первая строка содержит два целых числа n и q ($1 \leqslant n, q \leqslant 10^5$) — количество вершин в дереве и количество запросов, соответственно.

Каждая из следующих $n - 1$ вершин содержит два целых числа u и v ($1 \leqslant u, v \leqslant n, u \neq v$), обозначающие ребро между вершинами u и v. Гарантируется, что данный граф является деревом.

Каждая из следующих q строк начинается с трех целых чисел k, m и r ($1 \leqslant k, r \leqslant n$, $1 \leqslant m \leqslant min(300, k)$) — количество вершин, максимальный размер группы и корень дерева для данного запроса, соответственно. После этого следуют k различных целых чисел $a_1, a_2, \ldots, a_k$ ($1 \leqslant a_i \leqslant n$) — вершины текущего запроса.

Гарантируется, что сумма значений k по всем запросам не превосходит 10^5.

Формат выходных данных

Выведите q строк, где i-я строка содержит ответ на i-й запрос.

Примеры

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

Замечание

Рассмотрим первый пример.

В первом запросе нужно разделить три данные вершины (7, 4 и 3) на не более чем три группы, считая, что корнем дерева является вершина 2. Когда дерево подвешено за вершину 2, вершина 4 является предком вершин 3 и 7. Поэтому нельзя все вершины отнести к одной группе. Есть только 1 способ разделить эти вершины на две группы: [4] и [3, 7]. Кроме того, есть один способ разделить данные вершины на три группы: [7], [4] и [3]. Таким образом, есть всего 2 способа разбить данные вершины на не более чем три группы.

Во втором запросе дерево подвешено за вершину 4, при этом 6 является предком 2, а 2 является предком 1. Поэтому нельзя все вершины отнести к одной группе.

Задача E. Найти ближайшую

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

Дано дерево из n вершин, цвет i-й вершины равен a_i. Необходимо обработать q запросов (v_i, c_i): найти расстояние от v_i до ближайшей вершины цвета c_i. Расстояние между вершинами — минимальное количество рёбер в пути между ними.

Формат входных данных

Первая строка содержит n ($1 \leqslant n \leqslant 10^5$).
Следующая строка содержит $n-1$ число $p_1, \ldots p_{n-1}$ ($0 \leqslant p_i < i$). p_i — отец вершины i.
Следующая строка содержит числа $a_1, \ldots, a_n$ ($0 \leqslant a_i < n$).
Следующая строка содержит число q ($1 \leqslant q \leqslant 10^5$).
Следующие q строк содержат числа v_i, c_i ($0 \leqslant v_i < n$, $0 \leqslant c_i < n$).

Формат выходных данных

Для каждого запроса выведите расстояние до ближайшей вершины требуемого цвета, или -1, если такой нет.

Пример

стандартный ввод	стандартный вывод
5 0 1 1 3 1 2 3 2 1 9 0 1 0 2 0 3 1 0 2 1 2 2 3 3 3 1 4 2	0 1 2 -1 2 1 2 1 1

Задача F. Древландия

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1.5 секунд
Ограничение по памяти:	512 мегабайт

В древландии есть города, и первый город — *столица*. Города соединены автобусными маршрутами так, что для города $i \neq 1$ есть автобус, который идёт оттуда в город p_i ($p_i < i$), *именно в таком направлении*.

В стране есть национальные блюда. Каждый город имеет своё специальное блюдо, оно и только оно может быть куплено там. Тип специального блюда для каждого города — один из типов национальных блюд.

Несколько друзей из нескольких городов хотят встретиться в одном городе для вечеринки. Они выбирают город такой, что, если они одновременно начнут туда идти, они встретятся там быстро, как только возможно. Путешествие на автобусе требует 1 единицу времени.

Они хотят купить некоторые блюда для вечеринки, соблюдая следующие требования:

1. Каждый друг должен купить одно и то же количество блюд.
2. Не должно быть двух блюд одного типа на вечеринке.
3. Каждый друг может купить только блюда, соответствующие городам, которые он посетил.

Для заданных запросов, найдите максимальное количество блюд, которое может быть на вечеринке.

Формат входных данных

Первая строка содержит три числа n, m, q.

- n — количество городов
- m — количество типов национальных групп
- q — количество запросов

Вторая строка содержит $n - 1$ чисел $p_2, \ldots, p_n$, описывающие автобусные маршруты.

Третья строка содержит n чисел $a_1, \ldots, a_n$, описывающие типы блюд, продающиеся в соответсвующих городах.

Следующие строки содержат описания запросов. Каждый запрос описывается в следующем формате: число c, обозначающее количество друзей, а затем c чисел $v_1, \ldots v_c$. Пусть ответ на предыдущий ответ равен X (для первого запроса $X = 0$). Тогда друзья находятся в вершинах $(v_1 - 1 + X) \bmod n + 1, \ldots, (v_c - 1 + X) \bmod n + 1$.

Не гарантируется, что для конкретного запроса все v_i различны.

- $2 \leqslant n \leqslant 3 \cdot 10^5$
- $1 \leqslant m \leqslant 1000$
- $1 \leqslant q \leqslant 5 \cdot 10^4$
- $1 \leqslant p_i < i$
- $1 \leqslant a_i \leqslant m$
- $2 \leqslant c \leqslant 5$
- $1 \leqslant v_i \leqslant n$

Формат выходных данных

Выведите q строк, i-я из которых — ответ на i-й запрос.

Примеры

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

Задача G. Гоша и праздники

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	4 секунды
Ограничение по памяти:	512 мегабайт

Как известно, жители планеты Иннополис — очень педантичные люди. И даже когда дело касается праздников, они всегда хотят быть уверенными в том, что все пройдёт как по маслу. Так, расписание празднований всех событий на этой планете составлено почти на три миллиона лет вперёд! Гоша — большой любитель праздников. Он решил прилететь в какой-то из городов планеты Иннополис и посетить как можно больше праздников.

На планете Иннополис n городов, соединённых $n - 1$ двунаправленными дорогами так, что из любого города планеты можно добраться до любого другого, возможно, посещая другие города. Каждое событие на Иннополисе характеризуется номером города c_i, в котором оно будет отпраздновано, и номером дня d_i, в который его будут праздновать.

Гоша настолько везучий человек, что день его прибытия на планету имеет номер 0 в календаре планеты Иннополис, причём исходно он может прилететь в любой город планеты. Гоша решил узнать, какое максимальное количество праздников он может посетить на этой планете. Для этого он обратился за помощью к вам.

Формат входных данных

В первой строке входного файла задано одно число n ($n \geqslant 1$) — количество городов Иннополиса.

В следующих $n - 1$ строках заданы описания дорог, каждая дорога задается числами a_i, b_i и l_i ($1 \leqslant a_i, b_i, \leqslant n$; $l_i \geqslant 1$) — номера городов, которые соединяет дорога и число дней, необходимых на ее преодоление.

В следующей строке задано число m ($m \geqslant 1$) — число праздников на планете.

В следующих m строках заданы пары чисел c_i и d_i ($1 \leqslant c_i \leqslant n$; $d_i \geqslant 1$) — номер города и номер дня, в который пройдёт i-й праздник.

Ограничения: $n \leqslant 2 \cdot 10^5$, $m \leqslant 2 \cdot 10^5$, $l_i \leqslant 10^9$, $d_i \leqslant 10^9$.

Формат выходных данных

В единственной строке выходного файла выведите одно число — максимальное количество праздников, которое может посетить Гоша.

Примеры

стандартный ввод	стандартный вывод
4 1 2 1 2 3 1 2 4 3 4 1 3 2 4 3 1 4 5	3
11 2 1 2 3 2 5 4 1 5 5 2 4 6 5 1 7 1 2 8 3 4 9 6 2 10 7 2 11 2 2 9 1 67 1 34 11 16 5 97 4 70 2 20 2 61 2 26 2 70	8
10 2 1 1 3 2 4 4 2 4 5 3 2 6 4 5 7 5 4 8 3 1 9 6 2 10 7 5 9 7 34 10 82 2 48 3 66 8 98 2 66 3 3 8 59 5 22	8

Задача H. Количество путей

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	5 секунд
Ограничение по памяти:	256 мегабайт

Дано дерево из n вершин. Для каждого $d = 1 \dots n-1$ найдите количество путей длины d.

Формат входных данных

Первая строка содержит n ($1 \leqslant n \leqslant 50000$) — количество вершин.
Следующие $n-1$ строк содержат пары чисел u_i, v_i ($1 \leqslant u_i, v_i \leqslant n$), описывающие рёбра дерева.

Формат выходных данных

Выведите $n-1$ число, где i-е — количество путей длины i.

Пример

стандартный ввод	стандартный вывод
3 1 2 2 3	2 1

Задача I. Yet Another Tree Problem

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	5 секунд
Ограничение по памяти:	512 мегабайт

Дано дерево на n вершинах (дерево — это неориентированный связный граф без циклов). Каждое ребро имеет вес — целое число. Некоторые вершины дерева помечены. Вам нужно реализовать программу, выполняющую следующие операции с этим деревом:

1. Пометить вершину. Гарантируется, что перед этой операцией она была не помечена.
2. Сделать вершину не помеченной. Гарантируется, что до этого она была помечена.
3. Изменить вес ребра.

До выполнения всех операций и после каждой операции ваша программа должна вывести максимальное число x такое, что существует простой путь с **хотя бы двумя** помеченными вершинами на нем и суммарным весом x. Если в дереве нет двух помеченных вершин, выведите строку «`BAD`».

Формат входных данных

В первой строке заданы два целых числа n и q ($1 \leqslant n, q \leqslant 150\,000$), число вершин и число операций соответственно.

В следующей строке находятся n чисел, означающих, помечена ли вершина: 1 означает, что вершина помечена, 0 — не помечена.

В следующих $n-1$ строках задаются ребра дерева. Для всех i от 2 до n записаны два числа p_i ($1 \leqslant p_i < i$) и w_i ($-10^9 \leqslant w_i \leqslant 10^9$), которые означают, что в дереве есть ребро между вершинами i и p_i веса w_i.

В следующих q строках задаются операции. Каждая операция имеет одну из следующих форм:

- 1 x ($1 \leqslant x \leqslant n$): сделать вершину x помеченной,
- 2 x ($1 \leqslant x \leqslant n$): сделать вершину x не помеченной,
- 3 x w ($2 \leqslant x \leqslant n$, $-10^9 \leqslant w \leqslant 10^9$): сделать вес ребра между вершинами p_x и x равным w.

Формат выходных данных

Выведите $q+1$ строку. В первой строке выведите ответ до выполнения операций. В каждой следующей строке выведите ответ после выполнения очередной операции.

Система оценки

Обозначим за D максимальное число ребер, выходящих из одной вершины, а за H — максимальное число вершин на простом пути от вершины 1 до какой-то другой вершины.

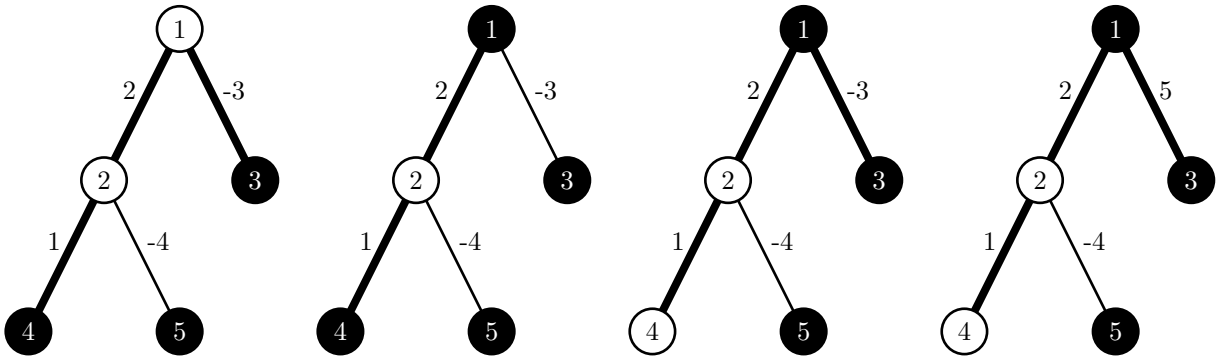
Подзадача	Баллы	Ограничения		
		n	q	Additional
1	3	$n \leqslant 20$	$q \leqslant 20$	—
2	3	$n \leqslant 500$	$q \leqslant 500$	—
3	3	$n \leqslant 100$	$q \leqslant 5000$	—
4	5	$n \leqslant 5000$	$q \leqslant 5000$	—
5	7	$n \leqslant 10^5$	$q \leqslant 10^5$	$D \leqslant 30$, $H \leqslant 20$
6	20	$n \leqslant 10^5$	$q \leqslant 10^5$	$H \leqslant 20$
7	10	$n \leqslant 10^5$	$q \leqslant 10^5$	$p_v = v - 1$ для всех v
8	17	$n \leqslant 10^5$	$q \leqslant 10^5$	$w_i \leqslant 0$, $w \leqslant 0$ для всех операций
9	22	$n \leqslant 50\,000$	$q \leqslant 50\,000$	—
10	5	$n \leqslant 10^5$	$q \leqslant 10^5$	—
11	5	$n \leqslant 150\,000$	$q \leqslant 150\,000$	—

Пример

стандартный ввод	стандартный вывод
5 7 0 0 1 1 1 1 2 1 -3 2 1 2 -4 1 1 2 4 3 3 5 2 3 2 1 3 5 -1 1 1	0 3 0 8 3 BAD BAD 6

Замечание

Иллюстрации к примеру. Жирным выделен путь веса x с хотя бы двумя помеченными вершинами.

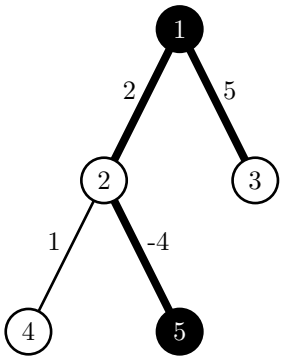


до операций

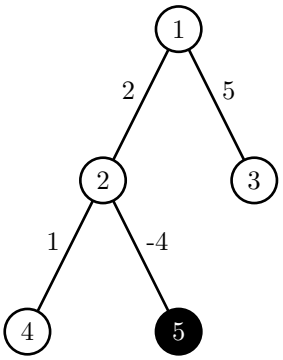
после операции 1

после операции 2

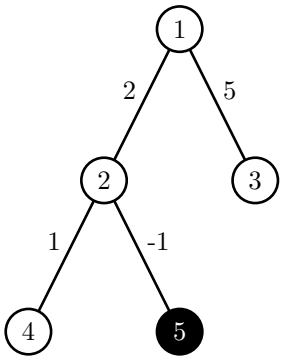
после операции 3



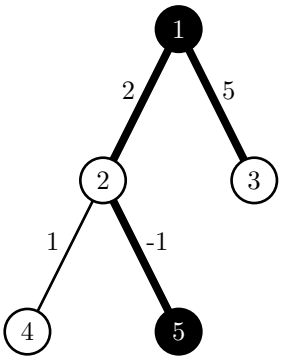
после операции 4



после операции 5



после операции 6



после операции 7

Задача J. Интерактивная вершина

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Это интерактивная задача

У Ильдара есть дерево из n вершин и он показал его вам. Он выбирает одну вершину u как специальную вершину, но он не сообщает вам ничего о ней!

Вместо этого, вы можете задавать ему вопросы. Для каждого вопроса вы должны выбрать вершину x, натуральное число k и k вершин $v_1, v_2, \ldots, v_k$ и он вам скажет, правда ли, что $\min(\text{dist}(u, v_i)) \geqslant \text{dist}(u, x)$. Здесь, $\text{dist}(p, q)$ это количество ребер на простом пути между вершинами p и q в дереве.

Вы должны угадать специальную вершину за не больше, чем $4\lceil \log_2 n \rceil$ вопросов.

Ильдар очень добрый, поэтому он не будет менять специальную вершину между вашими вопросами (другими словами, интерактор неадаптивный).

Поскольку ограничения большие и flush это тяжелая операция, убедитесь, что вы не делаете операцию flush слишком часто. Рекомендуется делать flush только после вывода каждого вопроса.

Протокол взаимодействия

Процесс взаимодействия начинается с того, что на первой строке вводится целое число n: количество вершин в дереве Ильдара ($2 \leqslant n \leqslant 200\,000$).

Каждая из следующих $n - 1$ строк содержит два целых числа u и v ($1 \leqslant u, v \leqslant n$), означающих ребро между u и v. Гарантируется, что данные ребра образуют дерево.

После этого, вы можете задавать вопросы.

Чтобы задать вопрос, выведите одну строку, содержащую "`?` k" ($1 \leqslant k \leqslant n$), целое число x ($1 \leqslant x \leqslant n$) и затем k различных целых чисел $v_1, v_2 \ldots, v_k$ ($1 \leqslant v_i \leqslant n$). Разделяйте соседние числа в строке ровно одним пробелом. Затем сделайте `flush` выходного потока.

После каждого вопроса, считайте одно целое число $ans \in \{0, 1\}$. Если $\min(\text{dist}(u, v_i)) \geqslant \text{dist}(u, x)$, тогда *ans* будет равно 1. Иначе, *ans* будет равно 0.

Когда вы нашли специальную вершину u ($1 \leqslant u \leqslant n$), выведите одно целое число "`!` u", сделайте flush выходного потока и завершите работу программы.

Ваше решение получит `Wrong Answer` или `Time Limit Exceeded` если вы сделаете больше чем $4\lceil \log_2 n \rceil$ вопросов.

Ваше решение получит `Idleness Limit Exceeded` если оно не совершает никаких действий или не делает flush выходного потока.

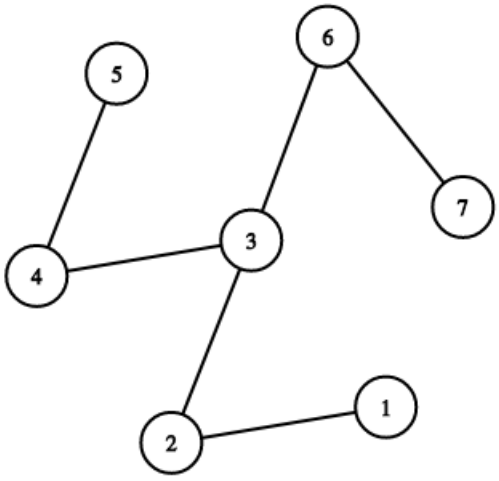
Чтобы сделать `flush` выходного потока, вы можете использовать:

- `fflush(stdout)` или `cout.flush()` в C++;
- `System.out.flush()` в Java;
- `flush(output)` в Pascal;
- `stdout.flush()` в Python;
- используйте документацию других языков.

Примеры

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

Замечание



Задача K. Iqea

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

Страна Гридландия располагается на бесконечном клетчатом поле и имеет форму клетчатой фигуры. Каждая клетка Гридландии является городом. Города, находящиеся в соседних по сторонам клетках, соединены дорогой длины 1. Из любого города Гридландии можно добраться до любого другого города Гридландии, перемещаясь по дорогам. Расстоянием между двумя городами является минимальная суммарная длина дорог, переместившись по которым можно добраться от одного города до другого. А также из любой клетки, не принадлежащей Гридландии, можно добраться до любой другой клетки, не принадлежащей Гридландии, перемещаясь только по клеткам, не принадлежащим Гридландии, в соседние по стороне клетки. Иными словами, Гридландия связна и дополнение Гридландии связно.

На данный момент ни в одном городе Гридландии нет магазинов популярной сети Iqea. Но у Iqea имеются большие планы по строительству сети магазинов в Гридландии. Для удобства покупателей Iqea решила разработать приложение, в котором житель любого города сможет узнать расстояние до ближайшего магазина Iqea. Вам поручили написать это приложение.

Вам будут поступать запросы двух типов:

- в городе, находящемся в клетке с координатами (x, y), открылся магазин Iqea;
- покупатель хочет узнать, каково минимальное расстояние до одного из уже открытых магазинов Iqea от его родного города, находящегося в клетке (x, y).

Обратите внимание, что покупатель может перемещаться только по дорогам и никогда не может покидать территорию Гридландии по пути до магазина.

Формат входных данных

В первой строке дано одно целое число n — количество городов в Гридландии ($1 \leqslant n \leqslant 100\,000$). В следующих n строках дано по два целых числа x_i и y_i — координаты клетки, в которой находится i-й город ($1 \leqslant x_i, y_i \leqslant 100\,000$). Все клетки различны и образуют связную область, дополнение которой тоже связно.

В следующей строке записано одно целое число q — количество запросов ($0 \leqslant q \leqslant 100\,000$). В следующих q строках описываются запросы. В i-й строке содержится три целых числа t_i, x_i и y_i ($t_i \in \{1, 2\}$, $1 \leqslant x_i, y_i \leqslant 100\,000$). Если $t_i = 1$, в городе, находящемся в клетке с координатами (x_i, y_i), открылся магазин Iqea. Гарантируется, что ранее в этом городе магазина Iqea не было. Если $t_i = 2$, требуется найти минимальное расстояние от города (x_i, y_i) до одного из уже открытых магазинов Iqea. Гарантируется, что в запросах обоих типов клетка (x_i, y_i) принадлежит Гридландии.

Формат выходных данных

На каждый запрос второго типа выведите в новой строке одно число — искомое минимальное расстояние. Если ни один магазин Iqea еще не открыт, выведите -1.

Система оценки

Подзадача	Баллы	Дополнительные ограничения
1	17	$n, q \leqslant 5\,000$
2	19	$n \leqslant 5\,000$
3	15	любой столбец и любая строка пересекают Гридландию по отрезку подряд идущих клеток
4	20	не существует таких x и y, что все 4 клетки (x, y), $(x, y+1)$, $(x+1, y)$ и $(x+1, y+1)$ принадлежат Гридландии
5	29	нет дополнительных ограничений

Примеры

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

Задача L. Peterson Polyglot

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	0.25 секунд
Ограничение по памяти:	256 мегабайт

Петрович обожает изучать новые языки, но самое любимое его увлечение — составление своих собственных. Языком Петрович называет множество слов, а словом — последовательность маленьких букв латинского алфавита.

Каждое утро Петрович составляет свой новый язык. Хранить все языки в явном виде очень сложно, поэтому Петрович придумал веник — специальную структуру данных для хранения языка, представляющую собой подвешенное дерево, на ребрах которого написаны буквы. Перед придумыванием языка веник представляет одну вершину — корень. При добавлении нового слова в язык Петрович встает в корень веника и обрабатывает буквы слова по одной. Пусть Петрович стоит в вершине u. Если из u есть ребро, на котором написана текущая буква, он переходит по нему. Иначе же, Петрович добавляет ребро из u в новую вершину v, пишет на нем текущую букву и переходит по этому ребру. Размером веника Петрович называет количество вершин в нем.

Вечером, приходя со смены, Петрович не может понять язык, придуманный утром: он ему кажется слишком сложным. Тогда Петрович старается упростить свой язык. Упрощением языка Петрович называет удаление букв из некоторых слов языка. Формально, Петрович фиксирует некоторое целое положительное число p, берет все слова, содержащие хотя бы p букв, и выкидывает из каждого из них букву с номером p. Буквы в слове Петрович предпочитает нумеровать, начиная с 1. Петрович считает, что при упрощении языка хотя бы одно слово должно измениться, то есть в языке должно быть хотя бы слово с длиной хотя бы p. Так как Петрович стремится сделать язык, придуманный утром, как можно проще, он старается подобрать число p таким образом, чтобы минимизировать размер веника, в котором он будет хранить язык.

Петровичу надоело заниматься одним и тем же каждый вечер, поэтому он обратился за помощью к вам. Напишите программу, которая будет находить минимальный размер веника, который может получиться в результате упрощения языка, придуманного Петровичем, и число p, которое нужно выбрать, чтобы получить такой размер.

Формат входных данных

В первой строке входного файла задано число n ($2 \leqslant n \leqslant 3 \cdot 10^5$) — размер веника языка Петровича.

В следующих $n - 1$ строках задано описание веника. В i-й из них записаны числа u_i, v_i и буква x_i, что соответствует ребру из u_i в v_i, на котором написана буква x_i.

Вершины пронумерованы числами от 1 до n. Все x_i являются маленькими буквами латинского алфавита. Вершина с номером 1 является корнем веника.

Гарантируется, что ребра описывают корректный веник, построенный по языку Петровича.

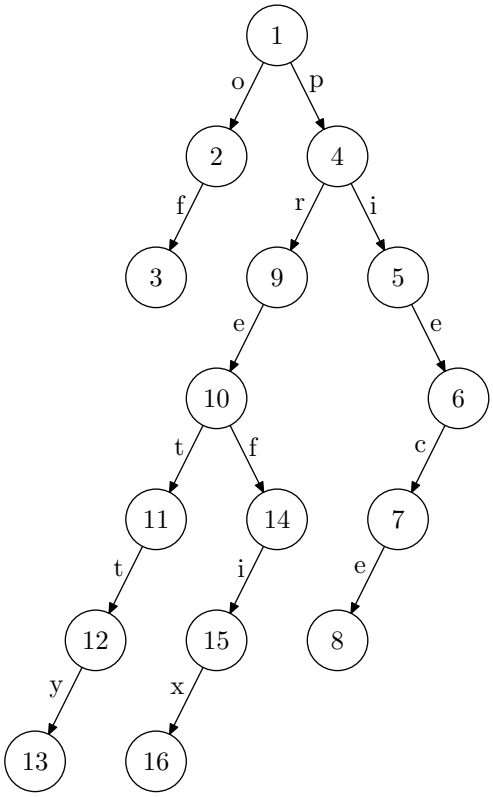
Формат выходных данных

В первой строке выведите минимальный возможный размер веника, который может получиться в результате упрощения языка.

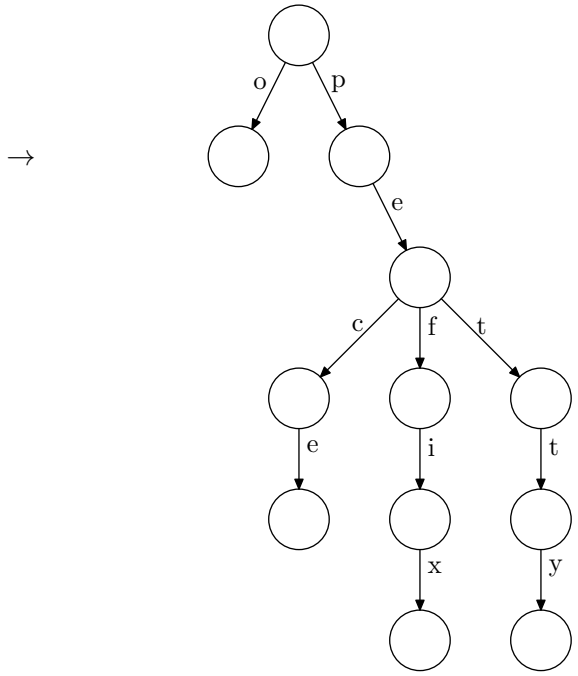
Во второй строке выведите число p, которое следует выбрать Петровичу для получения минимального размера. Если таких чисел p несколько, выведите минимальное из них.

Примеры

стандартный ввод	стандартный вывод
5 1 2 c 2 3 a 3 4 t 2 5 t	3 2
16 1 2 o 2 3 f 1 4 p 4 5 i 5 6 e 6 7 c 7 8 e 4 9 r 9 10 e 10 11 t 11 12 t 12 13 y 10 14 f 14 15 i 15 16 x	12 2



Веник для исходного языка из примера 2.



Веник после упрощения при $p = 2$.

Веник из второго примера может быть составлен из множества слов «`piece`», «`of`», «`pie`», «`pretty`», «`prefix`». После упрощения языка с $p = 2$ получается язык из слов «`pece`», «`o`», «`pe`», «`petty`», «`pefix`». Этот язык и задаёт веник минимального возможного размера.

Задача M. Учиться! - HARD

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	0.8 секунд
Ограничение по памяти:	256 мегабайт

Каждый год огномное количество выпускников, сдавшие ЕГЭ, выбирают, куда же они пойдут учиться. Не удивительно, что многие из них предпочитают перебраться поближе к столице. Транспортная инфраструктура страны переживает не лучшие времена, и в приемлемом качестве поддреживается минимально возможное число городов, необходимое для того, чтобы от любого города можно было добраться до любого другого.

Каждый выпускник оценивает свои результаты сдачи экзаменов, и решает, насколько далеко от свого родного города в сторону столицы он сможет уехать.

Выпускников настолько много, что вам не требуется выводить для каждого из них, до какого города он сможет доехать. Достаточно вывести сумму ответов для каждого выпускника.

Запросы генерируются следующим образом. Заданы числа a_1, a_2 и числа x, y и z. Числа $a_3, \ldots, a_{2m}$ генерируются следующим образом: $a_i = (x \cdot a_{i-2} + y \cdot a_{i-1} + z) \bmod n$. Первый запрос имеет вид $\langle a_1, a_2 \rangle$. Если ответ на $i-1$-й запрос равен v, то i-й запрос имеет вид $\langle (a_{2i-1} + v) \bmod n, a_{2i} \rangle$. В i-м запросе первое число соответствует городу, в котором окончил школу i-й выпускник, а второе — насколько далеко от родного города он может уехать. Все выпускники стараются перебраться как можно ближе к столице.

Формат входных данных

Первая строка содержит два числа: n ($1 \leqslant n \leqslant 100\,000$) и m ($1 \leqslant m \leqslant 10\,000\,000$). Столица имеет номер 0. Вторая строка содержит $n-1$ целых чисел, i-е из этих чисел равно номеру следующего за городом i на пути к столице. Третья строка содержит два целых числа в диапазоне от 0 до $n-1$: a_1 и a_2. Четвертая строка содержит три целых числа: x, y и z, эти числа неотрицательны и не превосходят 10^9.

Формат выходных данных

Выведите в выходной файл сумму номеров городов — ответов на все запросы.

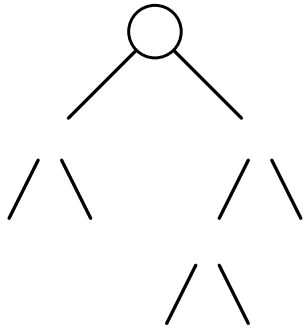
Примеры

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

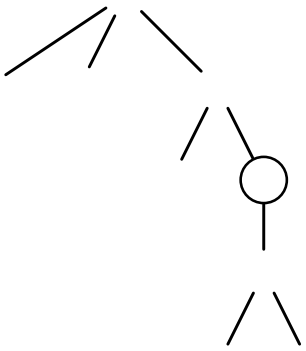
Задача N. Dynamic LCA

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

Постановка задачи о *наименьшем общем предке* такова: дано дерево T с выделенным корнем и две вершины u и v, $\mathrm{lca}(u, v)$ — вершина с максимальной глубиной, которая является предком и u, и v. Например, на картинке внизу $\mathrm{lca}(8, 7)$ — вершина 3.



С помощью операции $\mathrm{chroot}(u)$ мы можем менять корень дерева, достаточно отметить u, как новый корень, и направить ребра вдоль пути от корня. Наименьшие общие предки вершин поменяются соответствующе. Например, если мы сделаем $\mathrm{chroot}(6)$ на картинке сверху, $\mathrm{lca}(8, 7)$ станет вершина 6. Получившееся дерево изображено внизу.



Вам дано дерево T. Изначально корень этого дерева — вершина 1. Напишите программу, которая поддерживает эти две операции: $\mathrm{lca}(u, v)$ и $\mathrm{chroot}(u)$.

Формат входных данных

Входной файл состоит из нескольких тестов.

Первая строка каждого теста содержит натуральное число n — количество вершин в дереве ($1 \leqslant n \leqslant 100\,000$). Следующие $n - 1$ строк содержат по 2 натуральных числа и описывают ребра дерева. Далее идет строка с единственным натуральным числом m — число операций. Следующие m строк содержат операции. Строка `? u v` означает операцию $\mathrm{lca}(u, v)$, а строка `! u` — $\mathrm{chroot}(u)$. Последняя строка содержит число 0.

Сумма n для всех тестов не превосходит 100 000. Сумма m для всех тестов не превосходит 200 000.

Формат выходных данных

Для каждой операции `? u v` выведите значение $\mathrm{lca}(u, v)$. Числа разделяйте переводами строк.

Пример

стандартный ввод	стандартный вывод
9 1 2 1 3 2 4 2 5 3 6 3 7 6 8 6 9 10 ? 4 5 ? 5 6 ? 8 7 ! 6 ? 8 7 ? 4 5 ? 4 7 ? 5 9 ! 2 ? 4 3 0	2 1 3 6 2 3 6 2