

Задача А. Декомпозиция

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

Рассмотрим дерево T . Назовем деревом декомпозиции корневое дерево $D(T)$. Выберем любую из вершин дерева T , назовем ее r . Рассмотрим все компоненты связности дерева T , после удаления вершины r : $S_1, S_2, \dots, S_k$. Тогда корнем $D(T)$ будет вершина r , а детьми r в $D(T)$ будут $D(S_1), D(S_2), \dots, D(S_k)$.

Вам дано дерево T . Найдите дерево декомпозиции высоты не более 20. Высота дерева — максимальное число вершин в пути от корня до какой-то вершины.

Формат входных данных

Первая строка содержит n ($1 \leq n \leq 2 \cdot 10^5$) — количество вершин дерева.

Следующие $n - 1$ строк содержат пары чисел u_i, v_i ($1 \leq u_i, v_i \leq n$), описывающие рёбра дерева.

Формат выходных данных

Выведите n чисел, где i -е — родитель вершины i в дереве декомпозиции. Если вершина — корень, выведите 0.

Примеры

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

Задача В. 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 ($1 \leq n \leq 10^5$).

Следующая строка содержит $n - 1$ число $p_1, \dots, p_{n-1}$ ($0 \leq p_i < i$). p_i — отец вершины i .

Следующая строка содержит числа $a_1, \dots, a_n$ ($0 \leq a_i < n$).

Следующая строка содержит число q ($1 \leq q \leq 10^5$).

Следующие q строк содержат числа v_i, c_i ($0 \leq v_i < n, 0 \leq c_i < n$).

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 2 -1 2 1 2 1 1
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	

Задача С. Красим дерево

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

Дано взвешенное дерево. Вам необходимо выполнять 2 типа запросов:

- «1 v d c » — покрасить все вершины на расстоянии не более d от v в цвет c . Изначально все вершины имеют цвет 0.
- «2 v » — вывести цвет вершины v .

Формат входных данных

Первая строка содержит целое число n ($1 \leq n \leq 10^5$) — количество вершин в дереве.

Следующие $n - 1$ содержат тройки чисел u_i, v_i, w_i ($1 \leq u_i, v_i \leq n, 1 \leq w_i \leq 10^4$). i -е ребро соединяет вершины u_i, v_i и имеет вес w_i .

В следующей строке содержится количество запросов q ($1 \leq q \leq 10^5$).

Каждая из следующих q строк содержит запрос какого-то типа:

- 1 v d c ($1 \leq v \leq n, 0 \leq d \leq 10^9, 0 \leq c \leq 10^9$).
- 2 v ($1 \leq v \leq n$).

Формат выходных данных

Для каждого запроса второго типа выведите ответ на него.

Примеры

стандартный ввод	стандартный вывод
2 1 2 1 4 1 1 1 10 1 1 0 20 2 1 2 2	20 10
5 1 2 30 1 3 50 3 4 70 3 5 60 8 1 3 72 6 2 5 1 4 60 5 2 3 2 2 1 2 144 7 2 4 2 5	6 6 0 5 7

Задача D. 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 ($1 \leq n \leq 3 \cdot 10^5$).
Следующая строка содержит $n - 1$ число $p_2, \dots, p_n$ ($1 \leq p_i \leq i$). p_i — отец вершины i .
Следующая строка содержит число q ($1 \leq q \leq 3 \cdot 10^5$).
Следующие q строк содержат числа v_i, h_i ($1 \leq v_i \leq n, 0 \leq h_i \leq n$).

Формат выходных данных

Для каждого запроса выведите ответ.

Пример

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

Задача E. 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 ($2 \leq n \leq 3 \cdot 10^5$) — 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.

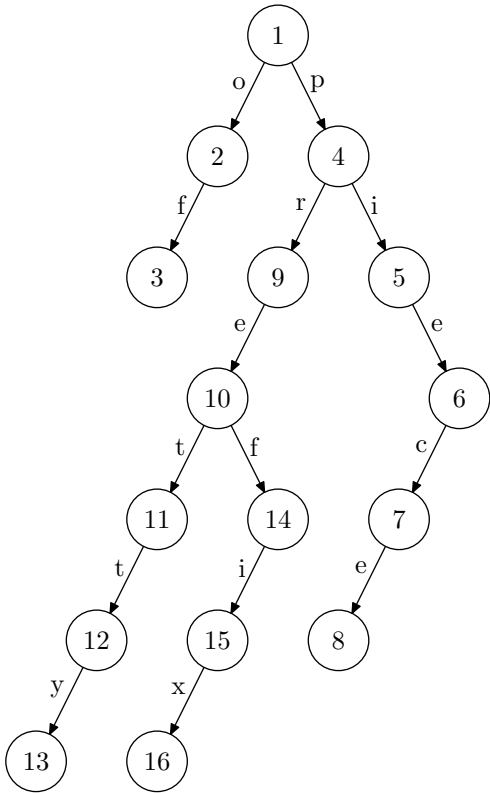
Формат выходных данных

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.

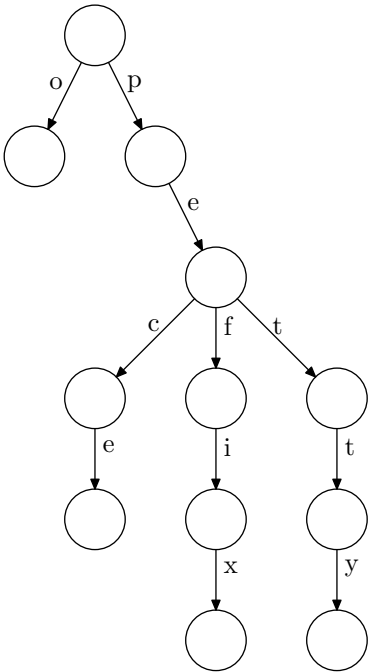
Примеры

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 «**pie**ce», «**of**», «**pie**», «**pre**tt**y**», «**pre**fix». После упрощения языка с $p = 2$ получается язык из слов «**pe**ce», «**o**», «**pe**», «**pet**t**y**», «**pe**fix». Этот язык и задаёт веник минимального возможного размера.

Задача F. Дерево и запросы

Имя входного файла: стандартный ввод
Имя выходного файла: стандартный вывод
Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Задано корневое дерево, состоящее из n вершин. Каждая вершина дерева имеет определенный цвет. Будем считать, что вершины дерева пронумерованы целыми числами от 1 до n . Тогда цвет вершины v будем обозначать c_v . Корнем дерева является вершина с номером 1.

В задаче вам требуется ответить на m запросов. Каждый запрос характеризуется двумя целыми числами v_j, k_j . Ответ на запрос v_j, k_j — это количество таких цветов вершин x , что в поддереве вершины v_j содержится как минимум k_j вершин цвета x .

Формат входных данных

В первой строке записаны два целых числа n и m ($2 \leq n \leq 10^5$; $1 \leq m \leq 10^5$). В следующей строке записана последовательность целых чисел $c_1, c_2, \dots, c_n$ ($1 \leq c_i \leq 10^5$). В следующих $n - 1$ строках записаны ребра дерева. В i -ой строке записаны числа a_i, b_i ($1 \leq a_i, b_i \leq n$; $a_i \neq b_i$) — вершины, соединенные ребром дерева.

Далее в m строках записаны запросы. В j -той строке записаны два целых числа v_j, k_j ($1 \leq v_j \leq n$; $1 \leq k_j \leq 10^5$).

Формат выходных данных

Выведите m целых чисел — ответы на запросы в порядке появления запросов во входных данных.

Пример

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

Задача G. Найти количество

Имя входного файла: стандартный ввод
Имя выходного файла: стандартный вывод
Ограничение по времени: 2 секунды
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 ($1 \leq n \leq 3 \cdot 10^5$).
Следующая строка содержит $n - 1$ число $p_2, \dots, p_n$ ($1 \leq p_i \leq i$). p_i — отец вершины i .
Следующая строка содержит число q ($1 \leq q \leq 3 \cdot 10^5$).
Следующие q строк содержат числа v_i, h_i ($1 \leq v_i \leq n, 1 \leq h_i \leq n$).

Формат выходных данных

Для каждого запроса выведите ответ.

Пример

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

Задача Н. Столицы

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

В стране Триландии близятся выборы новых столиц. Столицы в Триландии необычные, поскольку ими являются одновременно сразу три различных города. Такая идея размещения столиц основана на исследованиях эффективности управления страной, выполненных ведущими экономистами Триландии.

Всего в Триландии n городов, из которых некоторые пары городов соединены дорогами, и по каждой из них можно проехать в обе стороны. Время проезда по каждой дороге в одну сторону равно одному часу. При этом все города соединены дорогами таким образом, что из каждого города можно добраться в любой другой, причем это можно сделать единственным способом, если по каждой дороге проезжать не более одного раза и только в одну сторону.

Как показали результаты проведенных триландскими экономистами исследований, управление страной будет наиболее эффективным, если три столицы будут выбраны так, что время кратчайшего пути между каждой парой столиц составит ровно d часов. Перед проведением выборов необходимо знать, сколько существует различных троек городов, удовлетворяющих описанным выше свойствам. Две тройки городов считаются различными, если в первой тройке есть хотя бы один город, которого нет во второй тройке, и наоборот.

Требуется написать программу, которая по количеству городов в Триландии и описанию дорог находит количество троек городов, которые могут быть столицами.

Формат входных данных

Первая строка входного файла содержит два разделенных пробелом целых числа: количество городов в Триландии n и требуемое время в пути между столицами d ($3 \leq n \leq 10^5$, $1 \leq d < n$). Каждая из последующих $(n - 1)$ строк содержит описание одной дороги: пару разделенных пробелом различных целых чисел a_i и b_i — номера городов, которые соединены двусторонней дорогой ($1 \leq a_i \leq n$, $1 \leq b_i \leq n$, $a_i \neq b_i$). Каждая пара городов соединена не более чем одной дорогой.

Формат выходных данных

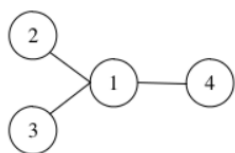
Выходной файл должен содержать одно целое число — количество подходящих троек городов, которые могут быть выбраны столицами. В случае, если нужных троек городов не окажется, выходной файл должен содержать ноль.

Примеры

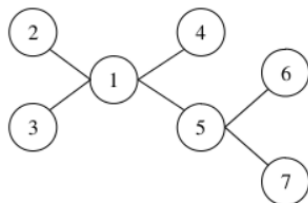
стандартный ввод	стандартный вывод
4 2 1 2 1 3 1 4	1
7 2 1 2 1 3 1 4 5 1 5 6 5 7	5

Замечание

В первом примере существует единственный способ выбрать три столицы: города под номерами 2, 3 и 4.



Во втором примере существует четыре варианта выбора трёх столиц из четверки городов: 2, 3, 4 и 5. Можно также выбрать столицами города с номерами 1, 6 и 7.



Задача I. Сигнализация

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

Подземный бункер состоит из n комнат, соединённых $n - 1$ коридорами. Каждый коридор соединяет две различные комнаты и имеет определённую длину. Бункер устроен таким образом, что из любой комнаты i можно дойти в любую другую комнату j . Заметим, что существует единственный такой путь, не проходящий по одному и тому же коридору дважды. Сумма длин коридоров, составляющих этот путь, называется расстоянием между комнатами i и j и обозначается $\rho(i, j)$.

Каждая комната бункера оборудована звуковой сигнализацией, состоящей из сирены и датчика звука, который её включает. Сирена, включённая в комнате i , активирует датчик звука в каждой комнате, расстояние до которой не превосходит расстояние d_i , определяемое мощностью этой сирены. Другими словами, включение сирены в комнате i автоматически включает сирену во всех комнатах j , таких что $\rho(i, j) \leq d_i$. Эта сирена, в свою очередь, может вызвать автоматическое включение других сирен и так далее.

В случае возникновения чрезвычайной ситуации некоторые сирены необходимо включить вручную, после чего звук от них автоматически включит сирены в других комнатах. Правила безопасности предписывают выбор такого набора сирен для ручного включения, который в конце концов приведёт к автоматическому включению сирен во всех комнатах.

Требуется написать программу, которая определяет минимальное количество сирен в наборе, удовлетворяющем правилам безопасности.

Формат входных данных

Первая строка входных данных содержит единственное число n — количество комнат.

Вторая строка содержит последовательность из n целых чисел d_i , i -е из них равно максимальному расстоянию, на котором расположенная в комнате i сирена активирует датчики ($0 \leq d_i \leq 10^9$).

Последующие $n - 1$ строк описывают коридоры бункера. В i -й из них находятся три целых числа: u_i, v_i, l_i , где u_i, v_i — номера различных комнат, соединённых коридором i , а l_i — длина этого коридора ($1 \leq u_i, v_i \leq n; 1 \leq l_i \leq 10^9$).

Формат выходных данных

Выходные данные должны состоять из единственного числа — минимального количества сирен, которые необходимо включить вручную.

Система оценки

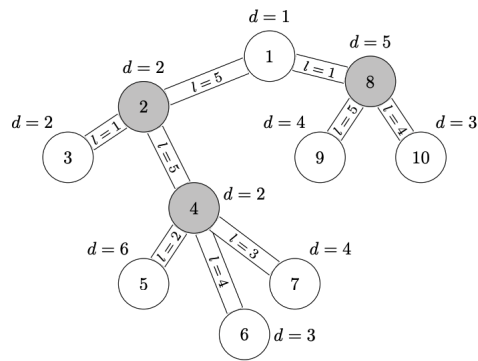
Подзадача	Баллы	Ограничения	Необх. подзадачи	Результаты во время тура
1	16	$1 \leq n \leq 15$	—	Потестовые
2	23	$1 \leq n \leq 100$	1	Потестовые
3	17	$1 \leq n \leq 3\,000$	1, 2	Потестовые
4	24	$1 \leq n \leq 100\,000$	1 - 3	Потестовые
5	20	$1 \leq n \leq 300\,000$	1 - 4	Потестовые

Пример

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

Замечание

В тесте из примера сирена в комнате 4 включает сирену в комнате 5, которая, в свою очередь, включает сирены в комнатах 6 и 7. Сирена в комнате 2 включает сирену в комнате 3. Сирена в комнате 8 включает сирены в комнатах 1, 9 и 10.



Задача J. Гигантский Пингвин

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	3 секунды
Ограничение по памяти:	512 мегабайт



Pengsoo — крайне популярный гигантский Корейский пингвин. Он очень невоспитанный и любит пить.

Сейчас Pengsoo очень занят — он отвечает на запросы в очередной задаче на графы.

У него есть связный неориентированный граф, в котором каждая **вершина** лежит не более, чем на k вершинно-простых циклах.

Он хочет отвечать на запросы двух типов.

- Пометить вершину v .
- Найти ближайшую помеченную вершину для вершины v (гарантируется, что на момент данного запроса в графе имеется хотя бы одна помеченная вершина).

Pengsoo очень ленивый, поэтому он решил вздремнуть и попросил вас ответить на данные запросы. Если вы не справитесь до того, как он проснется, он будет над вами издеваться, так что поторопитесь!

Формат входных данных

В первой строке записаны три числа n, m, k ($1 \leq n \leq 100\,000, n-1 \leq m \leq 200\,000, 0 \leq k \leq 10$) — количество вершин, ребер и максимальное количество вершинно-простых циклов, проходящих через одну вершину.

В следующих m строках содержится описание ребер. В i -й строке записаны два числа u_i, v_i ($1 \leq u_i, v_i \leq n, u_i \neq v_i$), означающие, что в графе есть ребро между вершинами u_i и v_i .

Гарантируется, что в графе нет красных ребер, граф является связным и каждая вершина лежит не более, чем на k вершинно-простых циклах.

В следующей строке записано число q ($1 \leq q \leq 200\,000$) — количество запросов.

В каждой из следующих q строк содержится описание запроса. В i -й строке записаны два числа t_i, v_i ($1 \leq t_i \leq 2, 1 \leq v_i \leq n$).

Если $t_i = 1$, пометьте вершину v_i . Гарантируется, что данная вершина не была помечена ранее.

Если $t_i = 2$, найдите расстояние до ближайшей помеченной вершины от вершины v_i . Гарантируется, что в графе уже есть хотя бы одна помеченная вершина.

Формат выходных данных

Для каждого запроса с $t_i = 2$ выведите расстояние до ближайшей помеченной вершины.

Примеры

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