

Задача А. Друзья и последовательности

Имя входного файла: `friends.in`
Имя выходного файла: `friends.out`
Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт

Майк и !Майк соперничают еще со школьных лет, они противоположны во всем что делают, кроме программирования. Сегодня у них возникла проблема, которую сами друзья сами решить не могут, но вместе с вами — кто знает?

Каждый из них знает две последовательности n чисел a и b . По запросу в виде пары целых чисел (l, r) Майк может сразу сообщить значение $\max_{i=l}^r a_i$, а !Майк — значение $\min_{i=l}^r b_i$.

Предположим, что робот задает им каждый из возможных различных запросов в виде пары целых чисел (l, r) ($1 \leq l \leq r \leq n$) (то есть он сделает ровно $n(n+1)/2$ запросов) и считает, сколько раз их ответы на один и тот же запрос совпадают, то есть для скольких пар выполняется $\max_{i=l}^r a_i = \min_{i=l}^r b_i$.

Сколько случаев совпадения посчитает робот?

Формат входных данных

В первой строке содержится единственное целое число n ($1 \leq n \leq 200\,000$).

Во второй строке содержатся n целых чисел $a_1, a_2, \dots, a_n$ ($-10^9 \leq a_i \leq 10^9$) — элементы последовательности a .

В третьей строке содержатся n целых чисел $b_1, b_2, \dots, b_n$ ($-10^9 \leq b_i \leq 10^9$) — элементы последовательности b .

Формат выходных данных

Выведите одно целое число — количество совпадений ответов, которые посчитает робот, то есть для скольких пар выполняется $\max_{i=l}^r a_i = \min_{i=l}^r b_i$.

Примеры

friends.in	friends.out
6 1 2 3 2 1 4 6 7 1 2 3 2	2
3 3 3 3 1 1 1	0

Задача В. Разрезанные таблицы

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

Дан массив из n чисел. Требуется написать программу, которая будет отвечать на запросы следующего вида: найти минимум на отрезке между u и v включительно.

Формат входных данных

В первой строке входного файла даны три натуральных числа n, m ($1 \leq n \leq 10^5, 1 \leq m \leq 10^7$) и a_1 ($0 \leq a_1 < 16\,714\,589$) — количество элементов в массиве, количество запросов и первый элемент массива соответственно. Вторая строка содержит два натуральных числа u_1 и v_1 ($1 \leq u_1, v_1 \leq n$) — первый запрос.

Элементы $a_2, a_3, \dots, a_n$ задаются следующей формулой:

$$a_{i+1} = (23 \cdot a_i + 21563) \bmod 16714589.$$

Например, при $n = 10, a_1 = 12345$ получается следующий массив: $a = (12345, 305498, 7048017, 11694653, 1565158, 2591019, 9471233, 570265, 13137658, 1325095)$.

Запросы генерируются следующим образом:

$$\begin{aligned} u_{i+1} &= ((17 \cdot u_i + 751 + ans_i + 2i) \bmod n) + 1, \\ v_{i+1} &= ((13 \cdot v_i + 593 + ans_i + 5i) \bmod n) + 1, \end{aligned}$$

где ans_i — ответ на запрос номер i .

Обратите внимание, что u_i может быть больше, чем v_i .

Формат выходных данных

В выходной файл выведите u_m, v_m и ans_m (последний запрос и ответ на него).

Примеры

<code>sparse.in</code>	<code>sparse.out</code>
10 8 12345 3 9	5 3 1565158

Замечание

Пояснение к тесту из примера: запросы и результаты.

a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
12345	305498	7048017	11694653	1565158	2591019	9471233	570265	13137658	1325095

#	u	v	ans
1	3	9	570265
2	10	1	12345
3	1	2	12345
4	10	10	1325095
5	5	9	570265
6	2	1	12345
7	3	2	305498
8	5	3	1565158

Задача С. Предок

Имя входного файла: `ancestor.in`
Имя выходного файла: `ancestor.out`
Ограничение по времени: 1 секунда
Ограничение по памяти: 256 мегабайт

Напишите программу, которая для двух вершин дерева определяет, является ли одна из них предком другой.

Формат входных данных

Первая строка входного файла содержит целое число n ($1 \leq n \leq 100\,000$) — количество вершин в дереве. Во второй строке находятся n чисел, i -е из которых определяет номер непосредственного родителя вершины с номером i . Если это число равно нулю, то вершина является корнем дерева.

В третьей строке находится число m ($1 \leq m \leq 100\,000$) — количество запросов.

Каждая из следующих m строк содержит два различных числа a и b ($1 \leq a, b \leq n$).

Формат выходных данных

Для каждого из m запросов выведите на отдельной строке число 1, если вершина a является одним из предков вершины b , и 0 в противном случае.

Примеры

ancestor.in	ancestor.out
6	0
0 1 1 2 3 3	1
5	1
4 1	0
1 4	0
3 6	
2 6	
6 5	

Задача D. LCA

Имя входного файла: lca2.in
Имя выходного файла: lca2.out
Ограничение по времени: 6 секунд
Ограничение по памяти: 256 мегабайт

Дано подвешенное дерево с корнем в первой вершине. Вам нужно ответить на m запросов вида “найти LCA двух вершин”.

LCA вершин u и v в подвешенном дереве — это наиболее удалённая от корня дерева вершина, лежащая на обоих путях от u и v до корня.

Формат входных данных

В первой строке задано целое число n — число вершин в дереве ($1 \leq n \leq 2 \cdot 10^5$).

В следующих $n - 1$ строках записано одно целое число x . Число x на строке i означает, что x — предок вершины i ($x < i$).

Затем дано число m .

Далее заданы m ($0 \leq m \leq 5 \cdot 10^5$) запросов вида (u, v) — найти LCA двух вершин u и v ($1 \leq u, v \leq n$; $u \neq v$).

Формат выходных данных

Для каждого запроса выведите LCA двух вершин на отдельной строке.

Примеры

lca2.in	lca2.out
5	1
1	1
1	
2	
3	
2	
2 3	
4 5	
5	2
1	2
1	1
2	
2	
3	
4 5	
4 2	
3 5	

Задача Е. Самое дешёвое ребро

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

Дано подвешенное дерево с корнем в первой вершине. Все ребра имеют веса (стоимости). Вам нужно ответить на m запросов вида «найти ребро минимального веса на пути между двумя заданными вершинами».

Формат входных данных

В первой строке файла записано одно число n — количество вершин в дереве ($1 \leq n \leq 50000$).

В i -й из следующих $n-1$ строк записано два целых числа p_{i+1} и w_{i+1} ($p_{i+1} < i+1$, $|w_{i+1}| \leq 10^6$) — предок вершины $i+1$ и вес ребра из вершины $i+1$ в её предка.

Следующая строка содержит число m ($0 \leq m \leq 5 \cdot 10^4$) — количество запросов. Следующие m строк содержат по одному запросу вида (x_i, y_i) — найти минимальный вес ребра на пути из вершины x_i в вершину y_i ($1 \leq x_i \leq n$, $1 \leq y_i \leq n$, $x_i \neq y_i$).

Формат выходных данных

Программа должна вывести m чисел — ответы на запросы.

Примеры

minonpath.in	minonpath.out
5	2
1 2	2
1 3	
2 5	
3 2	
2	
2 3	
4 5	

Задача F. LCA - 2

Имя входного файла: lca2.in
Имя выходного файла: lca2.out
Ограничение по времени: 3 секунды
Ограничение по памяти: 256 мегабайт

Задано подвешенное дерево, содержащее n вершин, пронумерованных от 0 до $n - 1$. Требуется ответить на m запросов о наименьшем общем предке для пары вершин.

Запросы генерируются следующим образом. Заданы числа a_1, a_2 и числа x, y и z .

Числа $a_3, \dots, a_{2m}$ 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 $\langle (a_{2i-1} + v) \bmod n, a_{2i} \rangle$.

Формат входных данных

Первая строка содержит два числа: n ($1 \leq n \leq 100\,000$) и m ($1 \leq m \leq 10\,000\,000$). Корень дерева имеет номер 0. Вторая строка содержит $n - 1$ целых чисел, i -е из этих чисел это предок вершины i . Третья строка содержит целые числа a_1 и a_2 ($0 \leq a_i \leq n - 1$).

Четвёртая строка содержит три целых числа: x, y и z ($0 \leq x, y, z \leq 10^9$)

Формат выходных данных

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

Примеры

lca2.in	lca2.out
3 2 0 1 2 1 1 1 0	2
1 2 0 0 1 1 1	0

Задача G. LCA

Имя входного файла: `lca.in`
Имя выходного файла: `lca.out`
Ограничение по времени: 5 секунды
Ограничение по памяти: 256 мегабайт

Дан ориентированный граф. Подсчитайте, сколько пар вершин (i, j) имеют общего предка. Общим предком вершин i и j называется такая вершину k , что и i , и j достижимы из k .

Формат входных данных

В первой строке входного файла содержатся целые числа $1 \leq N \leq 10^4$, $0 \leq M \leq 10^4$ — количество вершин и рёбер в графе. Далее следуют M строк по два числа от 1 до N . Пара чисел (a, b) означает, что из вершины a есть ребро в вершину b .

Формат выходных данных

Выведите одно целое число — количество пар.

Примеры

<code>lca.in</code>	<code>lca.out</code>
2 1 1 2	4
3 2 2 1 3 1	7

Задача Н. Классическая задача про гориллу

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

Однажды огромный самец гориллы нашёл дерево из n вершин с корнем в вершине 1. Детальное исследование этого дерева показало, что в каждой вершине лежит какое-то количество бананов, а именно, в вершине с номером v находятся a_v бананов.

Совершенно внезапно, вам предстоит ответить на q запросов. Запросы бывают двух видов:

1. заменить количество бананов в вершине v на x ;
2. вывести количество бананов в поддереве v .

Помогите горилле ответить на все запросы и спокойно насытиться бананами.

Формат входных данных

В первой строке даны целые числа n и q ($1 \leq n, q \leq 2 \cdot 10^5$) — количество вершин и запросов.

Во второй строке даны n целых чисел $a_1, a_2, \dots, a_n$ ($1 \leq a_i \leq 10^9$) — количества бананов в вершинах.

Следующие $n - 1$ строк содержат описания рёбер. Каждое ребро задаётся парой вершин v и u ($1 \leq v, u \leq n$).

Следующие q строк содержат описания запросов. Запросы описываются в следующем формате:

- $1 \ v \ x$ — запрос первого типа;
- $2 \ v$ — запрос второго типа.

Формат выходных данных

Выведите ответ на каждый запрос второго типа в отдельной строке.

Пример

стандартный ввод	стандартный вывод
5 3	8
4 2 5 2 1	10
1 2	
1 3	
3 4	
3 5	
2 3	
1 5 3	
2 3	

Замечание



Задача I. Дерево

Имя входного файла: `tree.in`
Имя выходного файла: `tree.out`
Ограничение по времени: 4 секунды
Ограничение по памяти: 256 мегабайт

Задано подвешенное дерево, содержащее n ($1 \leq n \leq 1\,000\,000$) вершин. Каждая вершина покрашена в один из n цветов. Требуется для каждой вершины v вычислить количество различных цветов, встречающихся в поддереве с корнем v .

Формат входных данных

В первой строке входного файла задано число n .

Следующие n строк описывают вершины, по одной в строке. Описание очередной вершины i имеет вид $p_i c_i$, где p_i — номер родителя вершины i , а c_i — цвет вершины i ($1 \leq c_i \leq n$). Для корня дерева $p_i = 0$.

Формат выходных данных

Выведите n чисел, обозначающих количества различных цветов в поддеревьях с корнями в вершинах $1, \dots, n$.

Примеры

tree.in	tree.out
5 2 1 3 2 0 3 3 3 2 1	1 2 3 1 1

Задача J. Опекуны карнотавров

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

Карнотавры очень внимательно относятся к заботе о своем потомстве. У каждого динозавра обязательно есть старший динозавр, который его опекает. В случае, если опекуна съедают (к сожалению, в юрский период такое не было редкостью), забота о его подопечных ложится на плечи того, кто опекал съеденного динозавра. Карнотавры — смертоносные хищники, поэтому их обычаи строго запрещают им драться между собой. Если у них возникает какой-то конфликт, то, чтобы решить его, они обращаются к кому-то из старших, которому доверяют, а доверяют они только тем, кто является их опекуном или опекуном их опекуна и так далее (назовем таких динозавров суперопекунами). Поэтому для того, чтобы решить спор двух карнотавров, нужно найти такого динозавра, который является суперопекуном для них обоих. Разумеется, беспокоить старших по пустякам не стоит, поэтому спорщики стараются найти самого младшего из динозавров, который удовлетворяет этому условию. Если у динозавра возник конфликт с его суперопекуном, то этот суперопекун сам решит проблему. Если у динозавра нелады с самим собой, он должен разобраться с этим самостоятельно, не беспокоя старших. Помогите динозаврам разрешить их споры.

Формат входных данных

В первой строке содержит целое число M ($1 \leq M \leq 200\,000$) — количество запросов.

Далее следуют M запросов, описывающие события:

- $+ \ v$ — родился новый динозавр и опекуном над ним взял динозавр с номером v . Родившемуся динозавру нужно присвоить наименьший натуральный номер, который до этого еще никогда не встречался.
- $- \ v$ — динозавра номер v съели.
- $? \ u \ v$ — у динозавров с номерами u и v возник конфликт и вам надо найти им третьего судью.

Изначально есть один прадинозавр номер 1. Гарантируется, что он никогда не будет съеден.

Формат выходных данных

Для каждого запроса типа «?» в выходной файл нужно вывести на отдельной строке одно число — номер самого молодого динозавра, который может выступить в роли третьего судьи.

Примеры

<code>carno.in</code>	<code>carno.out</code>
11	1
+ 1	1
+ 1	2
+ 2	2
? 2 3	5
? 1 3	
? 2 4	
+ 4	
+ 4	
- 4	
? 5 6	
? 5 5	