A. Разреженные таблицы

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: стандартный ввод

output: стандартный вывод

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

Входные данные:
В первой строке зданы три натуральных числа $m, n (1 \le n \le 10^5, 1 \le m \le 10^7)$ и $a_1 (0 \le a_1 \le 16714589)$ — количество элементов в массиве, количество запросов и первый элемент массива соответственно. Вторая строка содержит два натуральных числа u_1 и v_1 $(1 \le u_1, v_1 \le n)$ — первый запрос.

Для того, чтобы размер ввода был небольшой, массив и запросы генерируются.

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

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

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:

$$u_{i+1} = ((17 \cdot u_i + 751 + r_i + 2i) mod\ n) + 1,$$
$$v_i + 1((13 \cdot v_i + 593 + r_i + 5i) mod\ n) + 1,$$

где r_i — ответ на запрос номер i.
Обратите внимание, что u_i может быть больше, чем v_i.

Выходные данные:
В выходной файл выведите u_m, v_m и r_m (последний запрос и ответ на него).

Примеры:

входные данные:

```
10 8 12345
3 9
```

выходные данные:

```
5 3 1565158
```

B. LCA

time limit per test: 5 seconds

memory limit per test: 256 megabytes

input: стандартный ввод

output: стандартный вывод

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.

Входные данные:
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 ($1 \leq u, v \leq n; u \neq v$).

Выходные данные:
Для каждого запроса выведите LCA двух вершин на отдельной строке.

Примеры:

входные данные:

```
5
1
1
2
3
2
2 3
4 5
```

выходные данные:

```
1
1
```

входные данные:

```
5
1
1
2
2
3
4 5
4 2
3 5
```

выходные данные:

```
2
2
1
```

C. Самое дешевое ребро

time limit per test: 1 seconds

memory limit per test: 256 megabytes

input: стандартный ввод

output: стандартный вывод

Дано подвешенное дерево с корнем в первой вершине. Все ребра имеют веса (стоимости). Вам нужно ответить на m запросов вида «найти для двух вершин минимум среди стоимостей ребер на пути между ними»

Входные данные:
В первой строке файла записано одно число — n. (количество вершин).

В следующих $n-1$ строках записаны два числа — x и y. Число x на строке i означает, что x — предок вершины $i+1$, y означает стоимость ребра.

$x \leq i, |y| \leq 10^6$.

В следующей строке файла записано число m — количество запросов.

Далее m запросов вида (x, y) — найти минимум на пути из x в $y (x \neq y)$.

Ограничения: $2 \leq n \leq 5 \cdot 10^4, 0 \leq m \leq 5 \cdot 10^4$.

Выходные данные:
m строк — ответы на запросы.

Пример:
входные данные:

```
5
1 2
1 3
2 5
3 2
2
2 3
4 5
```

выходные данные:

```
2
2
```

D. Генеалогия

time limit per test: 4 seconds

memory limit per test: 256 megabytes

input: стандартный ввод

output: стандартный вывод

Во время обсуждений в Парламенте лорды, с похожими взглядами на решение проблемы, обычно объединяются в группы. Как правило, результат обсуждения зависит от решения наиболее влиятельной группы лордов. Именно поэтому подсчёт влиятельности группы является наиболее важной задачей.

Естественно, каждый лорд дорожит древностью своего рода, поэтому влиятельность лорда равна древности его рода. Древность рода лорда — количество предков лорда: его отец, его дед, его прадед, и т.д. Чтобы посчитать влиятельность группы лордов, требуется посчитать количество лордов в группе вместе с их предками. Отметим, что если лорд является предком двух или более лордов в группе, то этот лорд должен быть посчитан только один раз.

Вам дано фамильное дерево лордов (удивительно, но все лорды произошли от одного пра-лорда) и список групп. Для каждой группы найдите её влиятельность.

Входные данные:
Первая строка входного файла содержит число n — количество лордов ($1 \le n \le 100000$). Лорды нумеруются целыми числами от 1 до n. Следующая строка содержит n целых чисел $p_1, p_2, , p_n$, где p_i — отец лорда с номером i. Если лорд является основателем рода, то p_i равно -1. Гарантируется, что исходные данные формируют дерево. Третья строка входного файла содержит одно число g — количество групп ($1 \le g \le 3000000$). Следующие g строк содержат описания групп. j-ая строка содержит число k_j — размер j-ой группы, после которого следуют k_j различных чисел — номера лордов, состоящих в j-ой группе. Гарантируется, что сумма всех k_j во входном файле не превосходит 3000000.

Выходные данные:
В выходной файл выведите g строк. В j-ой строке выведите единственное число: влиятельность j-ой группы. Гарантируется, что размер выходного файла не превосходит шести мегабайт.

Примеры:

входные данные:

```
4
-1 1 2 3
4
1 4
2 3 4
3 2 3 4
4 1 2 3 4
```

выходные данные:

```
4
4
4
4
```

входные данные:

```
5
2 -1 1 2 3
10
3 3 4 1
3 2 4 3
4 1 3 5 4
1 4
2 2 3
3 1 4 3
1 2
3 3 4 5
1 1
3 1 2 4
```

выходные данные:

```
4
4
5
2
3
4
1
5
2
3
```

E. Опекуны карнотавров

time limit per test: 2 seconds

memory limit per test: 512 megabytes

input: стандартный ввод

output: стандартный вывод

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 динамозавра. 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 супероопекунами). Поэтому для того, чтобы решить спор двух карнотавров, нужно найти такого динозавра, который является супероопекуном 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 супероопекуном, то этот супероопекун сам решит проблему. Если у динозавра нелады с самим собой, он должен разобраться с этим самостоятельно, не беспокоя старших. Помогите динозаврам разрешить их споры.

Входные данные:
Во входном файле записано число M, обозначающее количество запросов ($1 \le M \le 200000$). Далее на отдельных строках следуют M запросов, обозначающих следующие события:

- $+v$ — родился новый динозавр и опекунство 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 третейского судью.

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

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

Пример:

входные данные:

```
11
+ 1
+ 1
+ 2
? 2 3
? 1 3
? 2 4
+ 4
+ 4
- 4
? 5 6
? 5 5
```

выходные данные:

```
1
1
2
2
5
```

F. Игры со стеками

time limit per test: 2 seconds

memory limit per test: 64 megabytes

input: stdin

output: stdout

Миша играет со стеками. В начале игры у него есть один пустой стек с номером 0. На i-ом шаге игры он выбирает существующий стек с номером v, копирует его и может сделать одну из следующих операций:

a) положить число i на вершину нового стека

b) извлечь число с вершины нового стека

c) выбрать другой стек с номером w и посчитать сколько различных чисел присутствует в обоих стеках — в новом стеке и в стеке с номером w

Новый стек имеет номер i.

Мише не нравится самому работать со стеками, поэтому он хочет, чтобы вы написали для него программу. Для каждой операции типа b выведите число, которое извлекли из стека, и для каждой операции типа c посчитайте количество различных чисел и выведите его.

Входные данные:
В первой строке задано число n ($1 \leq n \leq 300000$), количество шагов в игре Миши. Все шаги игры пронумерована числами от 1 до n. Каждая из следующих n строк содержит описание i-го шага игры в одном из трех вариантов:

- a v для операции типа a
- b v для операции типа b
- c v w для операции типа c

Первый символ в строке обозначает тип операции, а следующие одно или два числа обозначают номера стеков из интервала $[0, i-1]$
Для каждой операции типа b гарантируется, что стек не пуст.

Выходные данные:
Для каждого запроса типа b или c выведите соответствующее число, в том порядке, в котором следуют запросы.

Примеры:

входные данные:

```
5
a 0
a 1
b 2
c 2 3
b 4
```

выходные данные:

```
2
1
2
```

входные данные:

```
11
a 0
a 1
a 2
a 3
a 2
c 4 5
a 5
a 6
c 8 7
b 8
b 8
```

выходные данные:

```
2
2
8
8
```

Замечание:

Пояснение первого примера: В начале у нас есть стек $s_0 = \{\}$. На первом шаге, мы копируем стек s_0 и кладем 1 на вершину, теперь $s_1 = \{1\}$. На втором шаге мы копируем стек s_1 и кладем 2 на его вершину, $s_2 = \{1, 2\}$. На третьем шаге мы копируем s_2 и извлекаем из него 2, $s_3 = \{1\}$. На четвертом шаге мы копируем s2, нумеруем его как s4 и считаем количество совпадающих чисел в новом стеке s_4 и стеке s_3, единственное такое число это 1, поэтому ответ 1. На пятом шаге мы копируем s_4 и извлекаем из него число 2, $s_5 = \{1\}$.

G. Дерево

time limit per test: 3.5 seconds

memory limit per test: 384 megabytes

input: стандартный ввод

output: стандартный вывод

Задано подвешенное дерево, содержащее n ($1 \leq n \leq 1000000$) вершин. Каждая вершина покрашена в один из n цветов. Требуется для каждой вершины v вычислить количество различных цветов, встречающихся в поддереве с корнем v.

Входные данные:
В первой строке входного файла задано число n. Последующие n строк описывают вершины, по одной в строке. Описание очередной вершины i имеет вид p_i ci, где p_i — номер родителя вершины i, а c_i — цвет вершины i ($1 \leq c_i \leq n$). Для корня дерева $p_i = 0$

Выходные данные:
Выведите n чисел, обозначающих количества различных цветов в поддеревьях с корнями в вершинах $1, \ldots, n$

Пример:

входные данные:

```
5
2 1
3 2
0 3
3 3
2 1
```

выходные данные:

```
1 2 3 1 1
```

H. Прибавление на пути

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: стандартный ввод

output: стандартный вывод

Задано дерево. В каждой вершине есть значение, изначально все значения равны нулю. Требуется обработать запрос прибавления на пути и запрос значения в вершине.

Входные данные:
В первой строке задано целое число n — число вершин в дереве ($1 \le n \le 3 \cdot 10^5$).

В следующих $n-1$ строках заданы ребра дерева: по два целых числа v и u в строке — номера вершин, соединенных ребром ($1 \le v, u \le n$).

В следующей строке задано целое число m — число запросов ($1 \le m \le 5 \cdot 10^5$).

Следующие m строк содержат запросы в одном из двух форматов:

- $+\ v\ u\ d$ — прибавить число d во все значения в вершинах на пути от v до u ($1 \le v, u \le n; 1 \le d \le 10^9$);
- $?\ v$ — вывести значение в вершине v ($1 \le v \le n$).

Выходные данные:
Выведите ответы на все запросы.

Примеры:
входные данные:

```
5
1 2
1 3
3 4
3 5
5
+ 2 5 1
? 3
+ 1 1 2
? 1
? 3
```

выходные данные:

```
1
3
1
```

I. Чип и Дейл в лабиринте

time limit per test: 1 seconds

memory limit per test: 256 megabytes

input: стандартный ввод

output: стандартный вывод

Чип и Дейл спешат на помощь! Но внимательные зрители знают, что помощь как правило нужна самим Чипу и Дейлу, поэтому сегодня вам надо будет сыграть роль сообразительной Гаечки. Итак, Чип и Дейл снова попали в лапы к Толстопузу. Кот очень не любит грызунов и поэтому приготовил им изощренное испытание. Он собирается поместить их в лабиринт и посмотреть смогут ли они из него выбраться. Лабиринт представляет собой дерево, в котором каждое ребро имеет одно направление. Гаечка подслушала разговор Толстопуза со своими сообщниками и теперь знает несколько возможных вариантов: в какую точку лабиринта поместят её друзей, и где будет выход. Для каждого такого варианта она хочет понять, смогут ли Чип и Дейл найти выход, или нет.

Входные данные:
В первой строке записано число n ($n \leq 10^5$) — количество вершин дерева. В следующих $n-1$ строках описаны ребра дерева. В $(i+1)$-й строке записано номера вершин a_i, b_i, означающие, что в дереве есть ребро из вершины a_i в вершину b_i.

Далее на отдельной строке записано число m ($m \leq 10^5$) — количество запросов. После этого идут m строк с описанием запросов, в $(n+1+i)$-й строке записаны через пробел числа x_i и y_i.

Выходные данные:
Для каждого запроса на отдельной строке требуется вывести «Yes», если в графе есть путь между вершинами x_i и y_i, и «No» иначе

Пример:
входные данные:

```
4
1 2
3 1
4 1
6
1 2
3 2
2 3
4 2
4 3
2 1
```

выходные данные:

```
Yes
Yes
No
Yes
No
No
```

J. В бухгалтерии опять всё перепутали

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: стандартный ввод

output: стандартный вывод

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.

Входные данные:
В первой строке входных данных даны 2 числа n и m ($1 \le n, m \le 10^5$) — 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 ($1 \le c_i \le 10^9$) — зарплата i-го сотрудника в нулевой день.

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

Выходные данные:
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:
входные данные:

```
3 3
0 0
1 1 1
0 0
2 1
1 2
```

выходные данные:

```
1 4 4 28
```

входные данные:

```
4 3
0 1 1
1 2 1 1
0 1
1 3
2 3
```

выходные данные:

```
1 6 31 100
```

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