

## Задача А. Быстрое прибавление

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 4 секунды  
Ограничение по памяти: 256 мегабайт

Есть массив целых чисел длины  $n = 2^{24}$ , изначально заполненных нулями. Вам нужно сперва обработать  $m$  случайных запросов вида “прибавление на отрезке”. Затем обработать  $q$  случайных запросов вида “сумма на отрезке”.

### Формат входных данных

На первой строке числа  $m, q$  ( $1 \leq m, q \leq 2^{24}$ ). На второй строке пара целых чисел  $a, b$  от 1 до  $10^9$ , используемая в генераторе случайных чисел.

```
0. unsigned int a, b; // даны во входных данных
1. unsigned int cur = 0; // беззнаковое 32-битное число
2. unsigned int nextRand() {
3.     cur = cur * a + b; // вычисляется с переполнениями
4.     return cur >> 8; // число от 0 до  $2^{24} - 1$ .
5. }
```

Каждый запрос первого вида генерируется следующим образом:

```
1. add = nextRand(); // число, которое нужно прибавить
2. l = nextRand();
3. r = nextRand();
4. if (l > r) swap(l, r); // получили отрезок [l..r]
```

Каждый запрос второго вида генерируется следующим образом:

```
1. l = nextRand();
2. r = nextRand();
3. if (l > r) swap(l, r); // получили отрезок [l..r]
```

Сперва генерируются запросы первого вида, затем второго.

### Формат выходных данных

Выведите сумму ответов на все запросы второго типа по модулю  $2^{32}$ .

### Примеры

стандартный ввод	стандартный вывод
5 5 13 239	811747796

### Замечание

Последовательность запросов в тесте из примера:

```
[13..170] += 0
[28886..375523] += 2221
[2940943..13131777] += 4881801
[2025901..10480279] += 4677840
[4943766..6833065] += 9559505
get sum [13412991..13937319]
get sum [1871500..6596736]
get sum [7552290..14293694]
get sum [1268651..16492476]
get sum [2210673..13075602]
```

## Задача В. Разрезанные таблицы

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

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

стандартный ввод	стандартный вывод
10 8 12345 3 9	5 3 1565158

## Задача С. Прямоугольники

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

Есть таблица  $T$  размера  $N \times M$ . Элементами таблицы являются прямоугольники  $T_{ij}$ , где  $0 \leq i < N$  и  $0 \leq j < M$ . Прямоугольник  $T_{ij}$  задаётся четвёркой чисел  $(x_1^{ij}, y_1^{ij}, x_2^{ij}, y_2^{ij})$ , где  $(x_1^{ij}, y_1^{ij})$  и  $(x_2^{ij}, y_2^{ij})$  — координаты противоположных углов прямоугольника. Стороны прямоугольника параллельны осям координат.

Далее вам поступают запросы. Каждый запрос состоит из четырёх чисел:  $(r_1, c_1, r_2, c_2)$ . Ответом на такой запрос является площадь фигуры, являющейся пересечением всех прямоугольников  $T_{ij}$  таких, что  $\min(r_1, r_2) \leq i \leq \max(r_1, r_2)$  и  $\min(c_1, c_2) \leq j \leq \max(c_1, c_2)$ . Запросов очень много, поэтому мы просим вас вывести сумму ответов на все запросы по модулю  $10^9 + 7$ .

### Формат входных данных

В первой строке записаны два целых числа  $N$  и  $M$  — размеры таблицы  $T$  ( $1 \leq N, M \leq 127$ ). Далее в  $N$  строках описывается таблица  $T$ : в  $(i+1)$ -й строке  $(j+1)$ -я четвёрка чисел  $x_1^{ij} y_1^{ij} x_2^{ij} y_2^{ij}$  описывает прямоугольник  $T_{ij}$ . Гарантируется, что  $|x_k^{ij}|, |y_k^{ij}| \leq 10^6$ .

Дальше в отдельной строке записано четыре числа. Первое из них, число  $Q$  — количество запросов ( $1 \leq Q \leq 5 \cdot 10^6$ ). Следующие три числа — это  $A, B, v_0$  ( $0 \leq A, B, v_0 < 10^9 + 7$ ). При помощи этих чисел генерируется бесконечная последовательность  $\{v_i\}$  по правилу  $v_i = (A \cdot v_{i-1} + B) \bmod (10^9 + 7)$ .

После этого  $k$ -й запрос (запросы нумеруются с единицы) задаётся следующей четвёркой чисел:  $(v_{4k-3} \bmod N, v_{4k-2} \bmod M, v_{4k-1} \bmod N, v_{4k} \bmod M)$ .

### Формат выходных данных

Выведите сумму ответов на все запросы по модулю  $10^9 + 7$ .

### Примеры

стандартный ввод	стандартный вывод
2 2 0 0 2 2 1 1 3 3 0 3 2 1 1 2 3 0 1 5000000003 4 2	1
3 2 8 -1 -7 6 6 8 9 10 -4 -10 4 9 -3 -8 6 9 -2 -9 3 8 -5 7 7 3 5 303164476 273973578 65779139	85

### Замечание

В первом примере запрос имеет вид  $(1, 0, 0, 1)$ , то есть это запрос ко всей таблице. Пересечением всех прямоугольников является квадрат с углами в точках  $(1, 1)$  и  $(2, 2)$ . Его площадь равна 1.

Во втором примере запросы имеют вид  $(0, 1, 1, 1)$ ,  $(1, 0, 2, 0)$ ,  $(0, 0, 2, 1)$ ,  $(0, 1, 1, 1)$ ,  $(0, 1, 0, 0)$ . На второй запрос ответ — 85, на остальные — 0.

## Задача D. RMQ

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

### Формат входных данных

В первой строке находится число  $n$  — размер массива. ( $1 \leq n \leq 500000$ ) Во второй строке находится  $n$  чисел  $a_i$  — элементы массива. Далее содержится описание операций, их количество не превышает 1000000. В каждой строке находится одна из следующих операций:

- **set**  $i$   $x$  — установить  $a[i]$  в  $x$ .
- **min**  $i$   $j$  — вывести значение минимального элемента в массиве на отрезке с  $i$  по  $j$ , гарантируется, что  $(1 \leq i \leq j \leq n)$ .

В массив помещаются только целые числа, не превышающие по модулю  $10^9$ .

### Формат выходных данных

Выведите последовательно результат выполнения всех операций **min**. Следуйте формату выходного файла из примера.

### Пример

стандартный ввод	стандартный вывод
5	2
1 2 3 4 5	1
min 2 5	1
min 1 5	2
min 1 4	2
min 2 4	2
set 1 10	3
set 2 3	3
set 5 2	
min 2 5	
min 1 5	
min 1 4	
min 2 4	

## Задача E. RSQ

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

### Формат входных данных

В первой строке находится число  $n$  — размер массива. ( $1 \leq n \leq 500000$ ) Во второй строке находится  $n$  чисел  $a_i$  — элементы массива. Далее содержится описание операций, их количество не превышает 1000000. В каждой строке находится одна из следующих операций:

- **set**  $i$   $x$  — установить  $a[i]$  в  $x$ .
- **sum**  $i$   $j$  — вывести значение суммы элементов в массиве на отрезке с  $i$  по  $j$ , гарантируется, что  $(1 \leq i \leq j \leq n)$ .

Все числа во входном файле и результаты выполнения всех операций не превышают по модулю  $10^{18}$

### Формат выходных данных

Выведите последовательно результат выполнения всех операций **sum**. Следуйте формату выходного файла из примера.

### Пример

стандартный ввод	стандартный вывод
5	14
1 2 3 4 5	15
sum 2 5	10
sum 1 5	9
sum 1 4	12
sum 2 4	22
set 1 10	20
set 2 3	10
set 5 2	
sum 2 5	
sum 1 5	
sum 1 4	
sum 2 4	

## Задача F. Криптография

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

Задано  $n$  матриц  $A_1, A_2, \dots, A_n$  размера  $2 \times 2$ . Необходимо для нескольких запросов вычислить произведение матриц  $A_i, A_{i+1}, \dots, A_j$ . Все вычисления производятся по модулю  $r$ .

### Формат входных данных

Первая строка входного файла содержит числа  $r$  ( $1 \leq r \leq 10\,000$ ),  $n$  ( $1 \leq n \leq 200\,000$ ) и  $m$  ( $1 \leq m \leq 200\,000$ ). Следующие  $n$  блоков по две строки содержащие по два числа в строке — описания матриц. Затем следуют  $m$  пар целых чисел от 1 до  $n$ , запросы на произведение на отрезке.

### Формат выходных данных

Выведите  $m$  блоков по две строки, по два числа в каждой — произведения на отрезках. Разделяйте блоки пустой строкой. Все вычисления производятся по модулю  $r$ .

### Пример

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

## Задача G. RMQ

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

### Формат входных данных

В первой строке находится число  $n$  — размер массива. ( $1 \leq n \leq 100000$ ) Во второй строке находится  $n$  чисел  $a_i$  — элементы массива. Далее содержится описание операций, их количество не превышает 200000. В каждой строке находится одна из следующих операций:

- **set**  $i\ j\ x$  — установить все  $a[k]$ ,  $i \leq k \leq j$  в  $x$ .
- **add**  $i\ j\ x$  — увеличить все  $a[k]$ ,  $i \leq k \leq j$  на  $x$ .
- **min**  $i\ j$  — вывести значение минимального элемента в массиве на отрезке с  $i$  по  $j$ , гарантируется, что  $(1 \leq i \leq j \leq n)$ .

Все числа во входном файле и результаты выполнения всех операций не превышают по модулю  $10^{18}$

### Формат выходных данных

Выведите последовательно результат выполнения всех операций **min**. Следуйте формату выходного файла из примера.

### Пример

стандартный ввод	стандартный вывод
5	2
1 2 3 4 5	1
min 2 5	1
min 1 5	2
min 1 4	5
min 2 4	5
set 1 3 10	8
add 2 4 4	8
min 2 5	
min 1 5	
min 1 4	
min 2 4	

## Задача Н. Парковка

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

На кольцевой парковке есть  $n$  мест пронумерованных от 1 до  $n$ . Есть два вида событий прибытие машину на парковку и отъезд машины с парковки. Если машина приезжает на парковку, а её место занято, то она едет далее по кругу и встаёт на первое свободное место.

### Формат входных данных

В первой строке входного файла находится два числа  $n$  и  $m$  — размер парковки и количество запросов ( $1 \leq n, m \leq 100000$ ). В следующих  $m$  строках находятся события. Каждая из этих строк имеет следующий вид:

- `enter  $x$`  — приехала машина, которая хочет встать на место  $x$ . Для каждой такой команды выведите какое место займёт эта машина.
- `exit  $x$`  — уехала машина занимавшая место  $x$ . Гарантируется, что на этом месте была машина.

### Формат выходных данных

Выведите последовательно результаты выполнения всех операций `enter`.

### Пример

стандартный ввод	стандартный вывод
3 5	1
enter 1	2
enter 1	3
exit 1	1
enter 2	
enter 2	



## Задача I. Функция на отрезке

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 3 seconds  
Ограничение по памяти: 256 megabytes

Вам задан массив натуральных чисел  $a_1, a_2, \dots, a_n$ .

Вам нужно ответить на несколько запросов посчитать функцию

$$f(l, r) = \sum_{x \in \mathbb{N}} K_x^2 \cdot x,$$

где  $K_x$  — число вхождений числа  $x$  в отрезок  $a[l \dots r]$ .

### Формат входных данных

В первой строке заданы числа  $n$  и  $m$  — размер массива и число запросов ( $1 \leq n, m \leq 200\,000$ ). Во второй строке находится  $n$  чисел  $a_i$  — элементы массива ( $1 \leq a_i \leq 10^6$ ). Далее содержится описание запросов: в каждой строке заданы два натуральных числа  $l$  и  $r$  — отрезок, на котором нужно посчитать функцию  $f$  ( $1 \leq l \leq r \leq n$ ).

### Формат выходных данных

Выведите  $m$  строк — ответы на запросы.

### Примеры

стандартный ввод	стандартный вывод
3 2 4 7 4 2 3 1 3	11 23
8 4 3 3 6 6 3 7 3 3 2 3 1 4 2 2 1 8	9 36 3 106

## Задача J. Окна

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

На экране расположены прямоугольные окна, каким-то образом перекрывающиеся (со сторонами, параллельными осям координат). Вам необходимо найти точку, которая покрыта наибольшим числом из них.

### Формат входных данных

В первой строке входного файла записано число окон  $n$  ( $1 \leq n \leq 50\,000$ ). Следующие  $n$  строк содержат координаты окон  $x_{(1,i)} y_{(1,i)} x_{(2,i)} y_{(2,i)}$ , где  $(x_{(1,i)}, y_{(1,i)})$  — координаты левого верхнего угла  $i$ -го окна, а  $(x_{(2,i)}, y_{(2,i)})$  — правого нижнего (на экране компьютера  $y$  растет сверху вниз, а  $x$  — слева направо). Все координаты — целые числа, по модулю не превосходящие  $10^6$ .

### Формат выходных данных

В первой строке выходного файла выведите максимальное число окон, покрывающих какую-либо из точек в данной конфигурации. Во второй строке выведите два целых числа, разделенных пробелом — координаты точки, покрытой максимальным числом окон. Окна считаются замкнутыми, т. е. покрывающими свои граничные точки.

### Примеры

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

## Задача К. LCA offline

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 64 мегабайт

Изначально имеется дерево состоящее только из корня (вершина с номером 1). Требуется научиться отвечать на следующие запросы:

- ADD  $a\ b$  — подвесить вершину  $b$  за вершину  $a$  (гарантируется, что вершина  $a$  уже существует).
- GET  $a\ b$  — вернуть LCA вершин  $a$  и  $b$ .

Все номера вершин от 1 до  $N$ .

В каждый момент времени у нас есть одно дерево.

### Формат входных данных

В первой строке входного файла содержится число  $k$  — количество запросов. Следующие  $k$  строк содержат сами запросы. Гарантируется, что число запросов каждого из типов не превосходит 500 000.

### Формат выходных данных

Для каждого запроса типа GET выведите в отдельную строку одно целое число — ответ на соответствующий запрос.

### Примеры

стандартный ввод	стандартный вывод
9	1
ADD 1 2	1
ADD 1 3	1
ADD 2 4	2
GET 1 3	5
GET 2 3	
GET 3 4	
ADD 2 5	
GET 4 5	
GET 5 5	

## Задача L. Двумерные запросы

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

Вам задан массив размера  $2^{17}$ . Требуется ответить на запросы: сколько есть элементов  $f[i]$  таких, что  $l \leq i \leq r$  и  $x \leq f[i] \leq y$ .

### Формат входных данных

На первой строке число  $q$  ( $1 \leq q \leq 2^{17}$ ). На второй строке пара целых чисел  $a, b$  от 1 до  $10^9$ , используемая в генераторе случайных чисел.

```
0. unsigned int a, b; // даны во входных данных
1. unsigned int cur = 0; // беззнаковое 32-битное число
2. unsigned int nextRand17() {
3.     cur = cur * a + b; // вычисляется с переполнениями
4.     return cur >> 15; // число от 0 до  $2^{17} - 1$ .
5. }
6. unsigned int nextRand24() {
7.     cur = cur * a + b; // вычисляется с переполнениями
8.     return cur >> 8; // число от 0 до  $2^{24} - 1$ .
9. }
```

Сначала массив генерируется следующим образом:

```
1. for (int i = 0; i < 1 << 17; i++)
2.     f[i] = nextRand24();
```

Потом генерируются запросы следующим образом:

```
1. l = nextRand17();
2. r = nextRand17();
3. if (l > r) swap(l, r); // получили отрезок [l..r]
4. x = nextRand24();
5. y = nextRand24();
6. if (x > y) swap(x, y); // получили отрезок [x..y]
7. b += c; // c -- ответ на данный запрос, для ответа на запросы в online
```

### Формат выходных данных

Выведите сумму ответов на все запросы второго типа по модулю  $2^{32}$ .

### Примеры

стандартный ввод	стандартный вывод
5 13 239	111139

## Задача М. LCA Problem Revisited

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 4 секунды  
Ограничение по памяти: 64 мегабайта

Задано подвешенное дерево, содержащее  $n$  ( $1 \leq n \leq 100\,000$ ) вершин, пронумерованных от 0 до  $n-1$ . Требуется ответить на  $m$  ( $1 \leq m \leq 10\,000\,000$ ) 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:  $n$  и  $m$ . Корень дерева имеет номер 0. Вторая строка содержит  $n-1$  целых чисел,  $i$ -е из этих чисел равно номеру родителя вершины  $i$ . 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 вершин — ответов на все запросы.

### Примеры

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

## Задача N. 3D сумма

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 8 секунд  
Ограничение по памяти: 512 мегабайт

Задан трехмерный массив  $a$  размера  $n_x \times n_y \times n_z$  ( $n_x n_y n_z \leq 10^6$ ). Изначально он заполнен 0. Массив индексируется с 0.

Ответьте на два вида запросов в online:

1. **a**  $x_1 \ y_1 \ z_1 \ x_2 \ y_2 \ z_2 \ v$  — прибавить ко всем элементам  $a_{x,y,z}$  таким, что  $x_1 \leq x < x_2$ ,  $y_1 \leq y < y_2$  и  $z_1 \leq z < z_2$ ;
2. **s**  $x_1 \ y_1 \ z_1 \ x_2 \ y_2 \ z_2$  — вычислить сумму всех  $a_{x,y,z}$  таких, что  $x_1 \leq x < x_2$ ,  $y_1 \leq y < y_2$  и  $z_1 \leq z < z_2$ .

Все вычисления проводить по модулю  $2^{32}$ .

### Формат входных данных

В первой строке заданы три целых числа  $n_x, n_y$  и  $n_z$  ( $1 \leq n_x, n_y, n_z \leq 10^6$ ;  $n_x n_y n_z \leq 10^6$ ).

Во второй строке задано целое число  $q$  — число запросов ( $1 \leq q \leq 10^5$ ).

В следующих  $q$  строках заданы запросы, в каждом запросе  $0 \leq x_1 < x_2 \leq n_x$ ,  $0 \leq y_1 < y_2 \leq n_y$ ,  $0 \leq z_1 < z_2 \leq n_z$ ,  $0 \leq v \leq 10^9$ .

### Формат выходных данных

Выведите ответ: по одному целому числу на каждый запрос.

### Система оценки

Подзадача 1 (60 баллов):  $n_y = n_z = 1$

Подзадача 2 (30 баллов):  $n_z = 1$

Подзадача 3 (10 баллов): нет дополнительных ограничений

### Пример

стандартный ввод	стандартный вывод
3 2 4	0
5	20
a 0 0 0 1 2 3 5	42
s 1 1 1 2 2 2	
a 1 1 2 3 2 4 3	
s 0 0 0 2 2 2	
s 0 0 0 3 2 4	

## Задача О. Самое дешевое ребро

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 4 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 у двух вершин минимум среди стоимостей ребер пути между ними”.

### 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 записаны два целых числа  $x$  и  $y$ . Число  $x$  на строке  $i$  означает, что  $x$  — предок вершины  $i$ ,  $y$  задает стоимость ребра ( $x < i$ ;  $|y| \leq 10^6$ ).

Далее заданы  $m$  ( $0 \leq m \leq 5 \cdot 10^5$ ) запросов вида  $(x, y)$  — найти минимум на пути из  $x$  в  $y$  ( $1 \leq x, y \leq n$ ;  $x \neq y$ ).

### Формат выходных данных

Выведите ответы на запросы.

### Примеры

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

## Задача Р. Прибавление на пути

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 4 секунды  
Ограничение по памяти: 256 мегабайт

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

### Формат входных данных

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

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

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

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  $u$  ( $1 \leq v, u \leq n$ ;  $1 \leq d \leq 10^9$ );
- `? v` — вывести значение в вершине  $v$  ( $1 \leq v \leq n$ ).

### Формат выходных данных

Выведите ответы на все запросы.

### Примеры

стандартный ввод	стандартный вывод
5	1
1 2	3
1 3	1
3 4	
3 5	
5	
+ 2 5 1	
? 3	
+ 1 1 2	
? 1	
? 3	



## Задача Q. Пещеры и туннели

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

После посадки на Марс учёные нашли странную систему пещер, соединённых туннелями. И учёные начали исследовать эту систему, используя управляемых роботов. Было обнаружено, что существует ровно один путь между каждой парой пещер. Но потом учёные обнаружили специфическую проблему. Иногда в пещерах происходят небольшие взрывы. Они вызывают выброс радиоактивных изотопов и увеличивают уровень радиации в пещере. К сожалению, роботы плохо выдерживают радиацию. Но для исследования они должны переместиться из одной пещеры в другую. Учёные поместили в каждую пещеру сенсор для мониторинга уровня радиации. Теперь они каждый раз при движении робота хотят знать максимальный уровень радиации, с которым придётся столкнуться роботу во время его перемещения. Как вы уже догадались, программу, которая это делает, будете писать вы.

### Формат входных данных

Первая строка входного файла содержит одно целое число  $N$  ( $1 \leq N \leq 100\,000$ ) — количество пещер. Следующие  $N - 1$  строк описывают туннели. Каждая из этих строк содержит два целых числа —  $a_i$  и  $b_i$  ( $1 \leq a_i, b_i \leq N$ ), описывающие туннель из пещеры с номером  $a_i$  в пещеру с номером  $b_i$ . Следующая строка содержит целое число  $Q$  ( $1 \leq Q \leq 100\,000$ ), означающее количество запросов. Далее идут  $Q$  запросов, по одному на строку. Каждый запрос имеет вид « $C\ U\ V$ », где  $C$  — символ «I» либо «G», означающие тип запроса (кавычки только для ясности). В случае запроса «I» уровень радиации в  $U$ -й пещере ( $1 \leq U \leq N$ ) увеличивается на  $V$  ( $0 \leq V \leq 10\,000$ ). В случае запроса «G» ваша программа должна вывести максимальный уровень радиации на пути между пещерами с номерами  $U$  и  $V$  ( $1 \leq U, V \leq N$ ) после всех увеличений радиации (запросов «I»), указанных ранее. Предполагается, что изначальный уровень радиации равен 0 во всех пещерах, и он никогда не уменьшается со временем (потому что период полураспада изотопов много больше времени наблюдения).

### Формат выходных данных

Для каждого запроса «G» выведите одну строку, содержащую максимальный уровень радиации.

### Примеры

стандартный ввод	стандартный вывод
4	1
1 2	0
2 3	1
2 4	3
6	
I 1 1	
G 1 1	
G 3 4	
I 2 3	
G 1 1	
G 3 4	

## Задача R. Комментарии

Имя входного файла:	стандартный ввод
Имя выходного файла:	стандартный вывод
Ограничение по времени:	1 second
Ограничение по памяти:	256 megabytes

Петя очень любит писать программы в одну строку. Он даже придумал специальный язык программирования OneLine, в котором программа обязательно должна состоять из одной строки.

В этом языке, как и в любом другом, есть комментарии. Они оформляются следующим образом:

```
/* комментарий */
```

Компилятор OneLine распознает комментарии слева направо жадным образом. То есть он движется по тексту программы до самого левого вхождения сочетания символов «/\*» (если такое есть). Пусть символы «/» и «\*» находятся на позициях  $i$  и  $i + 1$  соответственно, тогда далее компилятор движется начиная с позиции  $i + 2$  до ближайшего вхождения «\*/» (если такое есть). Если сочетание «\*/» находится, то все символы от «/\*» до «\*/», включая границы, считаются комментарием, и поиск продолжается со следующего после «\*/» символа. Если же сочетание «\*/» не находится, то комментарием считаются все символы от «/\*» до конца строки.

Петя хочет написать среду для разработки на OneLine, для этого ему нужно быстро анализировать изменяющуюся программу. Для начала он хочет научиться после изменений быстро понимать, где комментарий, а где — нет.

Для простоты заменим все символы программы кроме «\*» и «/» на «.». Вам нужно обрабатывать два вида запросов:

- 1  $i$   $c$  — заменить символ с индексом  $i$  на  $c$  (символы нумеруются от 1 до  $n$ ).
- 2  $i$  — проверить, является ли символ с индексом  $i$  частью комментария.

### Формат входных данных

Первая строка содержит число  $n$ , длину программы ( $1 \leq n \leq 200\,000$ ). Вторая строка содержит текущее состояние программы. Программа состоит только из символов «\*», «/» и «.». Третья строка содержит число  $q$ , число запросов ( $1 \leq q \leq 200\,000$ ), следующие  $q$  строк содержат запросы.

### Формат выходных данных

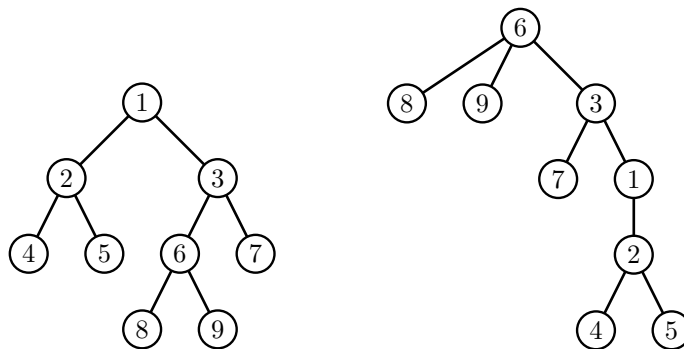
Для каждого запроса второго типа выведите «Yes» если символ является частью комментария и «No», если нет.

стандартный ввод	стандартный вывод
7 ./*/*/. 13 2 1 2 2 2 3 2 4 2 5 2 6 2 7 1 4 * 1 5 / 2 4 2 5 2 6 2 7	No Yes Yes Yes Yes Yes No Yes Yes No No      
5 ./*.. 8 2 1 2 2 2 3 2 4 2 5 1 5 * 2 4 2 5	No Yes Yes Yes Yes Yes     
5 .//*. 10 2 1 2 2 2 3 2 4 2 5 1 3 . 2 2 2 3 2 4 2 5	No No Yes Yes Yes No No No No    

## Задача S. Dynamic LCA

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 3 seconds  
Ограничение по памяти: 256 megabytes

Наименьший общий предок в дереве для вершин  $u$  и  $v$  — вершина  $lca(u, v)$ , которая является предком и  $u$ , и  $v$  и при этом имеет максимальную глубину. Например,  $lca(8, 7) = 3$  в дереве на картинке слева. В дереве можно поменять корень.  $chroot(u)$  делает корнем вершину  $u$ . Например, после операции  $chroot(6)$ , как видно на картинке справа,  $lca(8, 7) = 6$ .



Вам дано дерево с корнем в вершине 1. Напишите программу, отвечающую на запросы  $lca(u, v)$  и  $chroot(u)$ .

### Формат входных данных

Во входных данных дано несколько тестов.

Каждый тест начинается с количества вершин в дереве  $n$  ( $1 \leq n \leq 100\,000$ ). Следующие  $n - 1$  строк содержат пары целых чисел от 1 до  $n$  — рёбра дерева. Далее число запросов  $m$  ( $1 \leq m \leq 200\,000$ ). Каждая из следующих  $m$  строк имеет вид “?  $u$   $v$ ” для запроса  $lca(u, v)$  или “!  $u$ ” для запроса  $chroot(u)$ .

Последний тест имеет  $n = 0$ , его обрабатывать не нужно.

Сумма  $n$  по всем тестам не более 100 000. Сумма  $m$  по всем тестам не более 200 000.

### Формат выходных данных

Для каждой операции вида “?  $u$   $v$ ” выведите  $lca(u, v)$ .

### Примеры

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