

Задача А. Двоичное дерево поиска

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

Реализуйте сбалансированное двоичное дерево поиска.

Формат входных данных

Входной файл содержит описание операций с деревом, их количество не превышает 100000. В каждой строке находится одна из следующих операций:

- **insert** x — добавить в дерево ключ x . Если ключ x в дереве уже есть, то ничего делать не надо.
- **delete** x — удалить из дерева ключ x . Если ключа x в дереве нет, то ничего делать не надо.
- **exists** x — если ключ x есть в дереве, выведите «**true**», иначе «**false**»
- **next** x — выведите минимальный элемент в дереве, строго больший x , или «**none**», если такого нет.
- **prev** x — выведите максимальный элемент в дереве, строго меньший x , или «**none**», если такого нет.

Все числа во входном файле целые и по модулю не превышают 10^9 .

Формат выходных данных

Выведите последовательно результат выполнения всех операций **exists**, **next**, **prev**. Следуйте формату выходного файла из примера.

Примеры

bst.in	bst.out
insert 2	true
insert 5	false
insert 3	5
exists 2	3
exists 4	none
next 4	3
prev 4	
delete 5	
next 4	
prev 4	

Задача В. Переворот

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

Дан массив. Надо научиться обрабатывать два типа запросов.

- 1 L R - перевернуть отрезок [L, R]
- 2 L R - найти минимум на отрезке [L, R]

Формат входных данных

Первая строка файла содержит два числа n, m . ($1 \leq n, m \leq 10^5$) Во второй строке находится n чисел a_i ($1 \leq a_i \leq 10^9$)- исходный массив. Остальные m строк содержат запросы, в формате описанном в условии. Для чисел L,R выполняется ограничение ($1 \leq L \leq R \leq n$).

Формат выходных данных

На каждый запрос типа 2, во входной файл выведите ответ на него, в отдельной строке.

Примеры

reverse.in	reverse.out
10 7	3
5 3 2 3 12 6 7 5 10 12	2
2 4 9	2
1 4 6	2
2 1 8	
1 1 8	
1 8 9	
2 1 7	
2 3 6	

Задача С. Динамический Лес

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

Вам нужно научиться обрабатывать 3 типа запросов:

1. Добавить ребро в граф (`link`).
2. Удалить ребро из графа (`cut`).
3. По двум вершинам a и b вернуть длину пути между ними (или -1 , если они лежат в разных компонентах связности) (`get`).

Изначально граф пустой (содержит N вершин, не содержит ребер). Гарантируется, что в любой момент времени граф является лесом. При добавлении ребра гарантируется, что его сейчас в графе нет. При удалении ребра гарантируется, что оно уже добавлено.

Формат входных данных

Числа N и M ($1 \leq N \leq 10^5 + 1$, $1 \leq M \leq 10^5$) — количество вершин в дереве и, соответственно, запросов. Далее M строк, в каждой строке команда (`link` или `cut`, или `get`) и 2 числа от 1 до N — номера вершин в запросе.

Формат выходных данных

В выходной файл для каждого запроса `get` выведите одно число — расстояние между вершинами, или -1 , если они лежат в разных компонентах связности.

Примеры

linkcut.in	linkcut.out
3 7 get 1 2 link 1 2 get 1 2 cut 1 2 get 1 2 link 1 2 get 1 2	-1 1 -1 1
5 10 link 1 2 link 2 3 link 4 3 cut 3 4 get 1 2 get 1 3 get 1 4 get 2 3 get 2 4 get 3 4	1 2 -1 1 -1 -1

Задача D. Кратчайший путь

Имя входного файла:	<code>shortest.in</code>
Имя выходного файла:	<code>shortest.out</code>
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

В некотором царстве, в некотором государстве есть несколько городов, соединённых двусторонними дорогами. При этом страна разбита на провинции, внутри каждой провинции можно добраться от любого города до любого другого, а между провинциями пути не существует. Внутри каждой провинции есть один выделенный город, который называется столицей этой провинции.

Компания «AloeVera» занимается пассажирскими перевозками в этом государстве. Компания перевозит людей от одного города до другого по кратчайшему пути. Однако, дорожная сеть постоянно развивается, поэтому периодически в государстве появляются новые дороги. Строительство новой дороги — очень важное событие, поэтому при появлении новой дороги все стремятся проехать именно по ней. Более точно, при добавлении новой дороги между городами u и v существует две возможности:

1. Города u и v находятся в разных провинциях. Тогда после добавления этой дороги эти две провинции объединяются, при этом столицей становится столица первой провинции (той, в которой до добавления находилась вершина u).
2. Города u и v находятся в одной провинции. Тогда происходит перераспределение транспортных потоков. Услышав об открытии новой дороги, все продолжают добираться из столицы в другие города провинции по кратчайшему пути, но в случае, если таких путей существует несколько, то предпочтение отдается тому из них, на котором находится новая построенная дорога. После этого, если найдутся дороги, по которым никто ездить не будет, они навсегда удаляются из дорожной сети. Будьте внимательны: может так случиться, что по новой дороге никто ездить не будет, и тогда её не нужно добавлять вообще.

Иногда в государстве некоторые дороги приходят в негодность. При этом, если после удаления этой дороги некоторая часть провинции становится недостижимой из столицы, то она становится независимой провинцией, и её столицей назначается город, находящийся в этой провинции и являющийся концом удалённой дороги.

Основная же задача компании — сообщать людям о кратчайшем расстоянии между двумя какими-то городами. Ваша задача состоит в том, чтобы автоматизировать работу «AloeVera». При этом люди, сообщающие о разрушении дорог, иногда запаздывают или оперируют неверной информацией, и сообщают о разрушении дороги, которая уже была разрушена или вообще никогда не существовала.

Для решения поставленной задачи компания «AloeVera» использует следующий интерфейс:

- 1 u v — запрос об удалении дороги между городами u и v . Будьте внимательны, эта дорога могла быть разрушена ранее, или вообще никогда не существовать!
- 2 u v w — запрос о добавлении новой дороги между городами u и v , время проезда по которой будет равно w ($1 \leq w \leq 10^4$).
- 3 u v — запрос кратчайшего расстояния от города u до города v . В случае, если города находятся в разных провинциях, длина полагается равной -1 .

Во всех запросах номера городов корректны, то есть $1 \leq u, v \leq N$, где N обозначает общее число городов в государстве.

Формат входных данных

В первой строке входного файла содержится число N — количество городов в государстве. В следующей строке содержится число M — количество запросов. Далее в каждой строке содержится описание запроса. Описание соответствует условию.

$1 \leq N \leq 50\,000$, $1 \leq M \leq 100\,001$.

Формат выходных данных

Для каждого запроса типа 3 выведите единственное число: ответ на запрос.

Примеры

shortest.in	shortest.out
5 8 2 1 3 10 2 1 2 3 1 1 2 3 1 3 2 1 3 9 2 2 3 1 3 1 5 3 1 3	10 -1 9
5 10 2 2 3 2 2 3 1 2 2 1 5 2 2 5 4 2 2 2 3 2 2 1 5 2 1 3 1 2 3 1 1 2 3 1 2 3 3 1	1