Задача A. RMQ

Имя входного файла: `rmq.in`
Имя выходного файла: `rmq.out`

Формат входного файла

В первой строке находится число n — размер массива. ($1 \le n \le 500000$) 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 ($1 \le i \le j \le n$).

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

Формат выходного файла

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

Пример

rmq.in	rmq.out
5 1 2 3 4 5 min 2 5 min 1 5 min 1 4 min 2 4 set 1 10 set 2 3 set 5 2 min 2 5 min 1 5 min 1 4 min 2 4	2 1 1 2 2 2 3 3

Задача B. RSQ

Имя входного файла: `rsq.in`
Имя выходного файла: `rsq.out`

Формат входного файла

В первой строке находится число n — размер массива. ($1 \le n \le 500000$) 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 ($1 \le i \le j \le n$).

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

Формат выходного файла

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

Пример

rsq.in	rsq.out
5 1 2 3 4 5 sum 2 5 sum 1 5 sum 1 4 sum 2 4 set 1 10 set 2 3 set 5 2 sum 2 5 sum 1 5 sum 1 4 sum 2 4	14 15 10 9 12 22 20 10

Задача C. Двоичное дерево поиска

Имя входного файла: `bst.in`
Имя выходного файла: `bst.out`

Формат входного файла

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 есть в дереве, 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», если нет «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

В дерево помещаются и извлекаются только целые числа, не превышающие по модулю 10^9.

Формат выходного файла

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

Пример

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

Задача D. Криптография

Имя входного файла: `crypto.in`
Имя выходного файла: `crypto.out`

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

Формат входного файла

Первая строка входного файла содержит числа r ($1 \le r \le 10\,000$), n ($1 \le n \le 200\,000$) и m ($1 \le m \le 200\,000$). 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 выходного файла

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

crypto.in	crypto.out
3 4 4 0 1 0 0 2 1 1 2 0 0 0 2 1 0 0 2 1 4 2 3 1 3 2 2	0 2 0 0 0 2 0 1 0 1 0 0 2 1 1 2

Задача E. RMQ 2

Имя входного файла: `rmq2.in`
Имя выходного файла: `rmq2.out`

Формат входного файла

В первой строке находится число n — размер массива. ($1 \le n \le 100000$) 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 $a[k]$, $i \le k \le j$ в x.
- `add` i j x — увеличить все $a[k]$, $i \le k \le j$ на x.
- `min` i j — вывести значение минимального элемента в массиве на отрезке с i по j, гарантируется, что $(1 \le i \le j \le n)$.

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

Формат выходного файла

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

Пример

rmq2.in	rmq2.out
5 1 2 3 4 5 min 2 5 min 1 5 min 1 4 min 2 4 set 1 3 10 add 2 4 4 min 2 5 min 1 5 min 1 4 min 2 4	2 1 1 2 5 5 8 8

Задача F. Парковка

Имя входного файла: `parking.in`
Имя выходного файла: `parking.out`

На кольцевой парковке есть n мест пронумерованых от 1 до n. Есть два вида событий: прибытие машины 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 входного файла

В первой строке входного файла находится два числа n и m — размер парковки и количество запросов($1 \le n, m \le 100000$). 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 выходного файла

Выведите последовательно результаты выполнения всех операций `enter`.

Пример

parking.in	parking.out
3 5 enter 1 enter 1 exit 1 enter 2 enter 2	1 2 3 1

Задача G. Река

Имя входного файла: `river.in`
Имя выходного файла: `river.in`

Во Флатландии протекает богатая рыбой река Большой Флат. Много лет назад река была поделена между n рыболовными предприятиями, каждое из которых получило непрерывный отрезок реки. При этом i-е предприятие, если рассматривать их по порядку, начиная от истока, изначально получило отрезок реки длиной a_i.

С тех пор с рыболовными предприятиями во Флатландии k раз происходили различные события. Каждое из событий было одного из двух типов: банкротство некоторого предприятия или разделение некоторого предприятия на два.

При некоторых событиях отрезок реки, принадлежащий предприятию, с которым это событие происходит, делится на две части. Каждый такой отрезок имеет длину большую или равную 2. Деление происходит по следующему правилу. Если отрезок имеет четную длину, то он делится на две равные части. Иначе он делится на две части, длины которых различаются ровно на единицу, при этом часть, которая ближе к истоку реки, имеет меньшую длину.

При банкротстве предприятия происходит следующее. Отрезок реки, принадлежавший обанкротившемуся предприятию, переходит к его соседям. Если у обанкротившегося предприятия один сосед, то этому соседу целиком передается отрезок реки обанкротившегося предприятия. Если же соседей двое, то отрезок реки делится на две части описанным выше способом, после чего каждый из соседей присоединяет к своему отрезку ближайшую к нему часть.

При разделении предприятия отрезок реки, принадлежавший разделяемому предприятию, всегда делится на две части описанным выше способом. Разделившееся предприятие ликвидируется, и образуются два новых предприятия. Таким образом, после каждого события каждое предприятие владеет некоторым отрезком реки.

Министерство финансов Флатландии предлагает ввести налог на рыболовные предприятия, пропорциональный квадрату длины отрезка реки, принадлежащего соответствующему предприятию. Чтобы проанализировать, как будет работать этот налог, министр хочет по имеющимся данным узнать, как изменялась величина, равная сумме квадратов длин отрезков реки, принадлежащих предприятиям, после каждого произошедшего события.

Требуется написать программу, которая по заданному начальному разделению реки между предприятиями и списку событий, происходивших с предприятиями, определит, чему равна сумма квадратов длин отрезков реки, принадлежащих предприятиям, в начальный момент времени и после каждого события.

Формат входного файла

Первая строка входного файла содержит число: n — исходное количество предприятий ($2 \le n \le 100000$).

Вторая строка входного файла содержит n целых чисел a_1, a_2, ... , a_n — длины исходных отрезков реки.

Третья строка входного файла содержит целое число k — количество событий, происходивших с предприятиями ($1 \le k \le 100000$).

Последующие k строк содержат описания событий, i-я строка содержит два целых числа: e_i и v_i — тип события и номер предприятия, с которым оно произошло. Значение $e_i = 1$ означает, что предприятие, которое после всех предыдущих событий является v_i-м по порядку, если считать с единицы от истока реки, обанкротилось, а значение $e_i = 2$ означает, что это предприятие разделилось на два. Гарантируется, что значение v_i не превышает текущее количество предприятий. Гарантируется, что если отрезок предприятия при банкротстве или разделении требуется поделить на две части, то он имеет длину большую или равную 2. Гарантируется, что если на реке осталось единственное предприятие, оно не банкротится.

Формат выходного файла

Выходной файл должен содержать $(k + 1)$ целых чисел, по одному в строке. Первая строка

должна содержать исходную сумму квадратов длин отрезков реки, а каждая из последующих k строк — сумму квадратов длин отрезков реки после очередного события.

Пример

river.in	river.in
4 0 3 5 5 4 5 1 1 2 1 1 3 2 2 1 3	75 105 73 101 83 113

Задача I. Конфетки

Имя входного файла: `candies.in`
Имя выходного файла: `candies.out`

У Кролика день рождения! Он пригласил в гости n гостей. Чтобы гостям не было грустно и скучно, Кролик купил n коробок конфет. Кролик любит разнообразие, поэтому конфеты были разные. В i-й коробке лежало a_i конфет.

В назначенный день с самого утра к Кролику начали приходить гости. Каждый гость характеризуется своей наглостью b_i. Это означает, что, зайдя домой к Кролику и увидев коробки конфет, он брал из каждой коробки, в которой не меньше, чем b_i, конфет, по одной и съедал её. Например, у Винни-Пуха вполне могла была быть наглость один. Это значит, что он бы съел по конфете из каждой коробки.

Вечером, когда гости разошлись, Кролику стало интересно, кто съел сколько конфет. Помогите ему определить это.

Формат входного файла

В первой строке задано целое число n ($1 \le n \le 100\,000$) — количество коробок конфет. В следующей строке задано n натуральных чисел a_i ($1 \le a_i \le 10^9$) — сколько конфет в каждой коробке.

Далее, в следующей строке задано число число m ($1 \le m \le 100\,000$) — количество гостей. В четвёртой и последней строке задано m чисел b_i ($1 \le b_i \le 10^9$) — наглости гостей.

Формат выходного файла

В выходной файл выведите n строк, i-ая из которых должна содержать количество конфет съеденных i-ым гостем.

Примеры

candies.in	candies.out
3 3 1 1 2 1 1	3 1

Задача J. Атомы

Имя входного файла: `atoms.in`
Имя выходного файла: `atoms.out`

В лаборатории аномальных материалов антинаучно-исследовательского комплекса «Black Mesa» проводят эксперименты с недавно разработанным графитовым наностержнем. Графитовый наностержень представляет собой n последовательно соединенных атомов углерода, находящихся на одной прямой. Каждый атом имеет определенный заряд.

Для проведения эксперимента, стержень располагают вертикально. Пронумеруем атомы от 1 до n снизу вверх. Между двумя атомами образуется сильная связь, если это соседние атомы и верхний из них имеет заряд ровно на один больше, чем нижний. Иными словами, атомы a и b соединены сильной связью, если $a = b + 1$ и $q_a = q_b + 1$, где q_i — заряд i-го атома. Цепочкой атомов назовем несколько последовательных атомов, соединенных сильными связями.

Вчера был проведен очередной эксперимент. Перед началом эксперимента каждому атому установили определенный заряд: i-му атому установили заряд q_i.

Во время эксперимента ученые проводили действия двух типов:

- у всех атомов с номерами от l_i до r_i, включительно, заряд изменяли на величину d_i;
- временно разрушали все сильные связи атомов, кроме тех, которые соединяют атомы с номерами от l_i до r_i, включительно, и измеряли длину самой длинной цепочки атомов среди оставшихся сильных связей. Затем восстанавливали все временно разрушенные связи.

Было произведено m действий, однако выяснилось, что в результате побочного эффекта эксперимента запись результатов измерений оказалась утеряна. Для продолжения работы с графитовым наностержнем необходимо восстановить результаты вчерашних измерений. К счастью, сохранился план действий, произведенных во время эксперимента. Помогите ученым продолжить исследования, восстановите результаты измерений.

Формат входного файла

В первой строке находится одно целое число n ($1 \le n \le 100\,000$) — количество атомов в наностержне. Во второй строке находятся n чисел q_i ($|q_i| \le 10^9$) — начальный заряд i-го атома. В третьей строке находится одно целое число m ($0 \le m \le 100\,000$) — количество действий в эксперименте. В следующих m строках содержится описание эксперимента.

Если строка начинается с символа «+», очередное действие — изменение заряда атомов. В таком случае, далее в этой строке находятся три целых числа: l_i, r_i и d_i ($1 \le l_i \le r_i \le n$, $|d_i| \le 10^9$), которые характеризуют это действие.

Если строка начинается с символа «?», очередное действие — второго типа. В таком случае, далее в этой строке находятся два целых числа: l_i и r_i ($1 \le l_i \le r_i \le n$), которые характеризуют это действие.

Формат выходного файла

Для каждого действия второго типа выведите в новой строке одно число — длину наибольшей цепочки.

Пример

atoms.in	atoms.out
6 2 3 4 3 4 4 5 ? 1 6 + 6 6 1 ? 2 6 + 4 6 2 ? 1 5	3 3 5

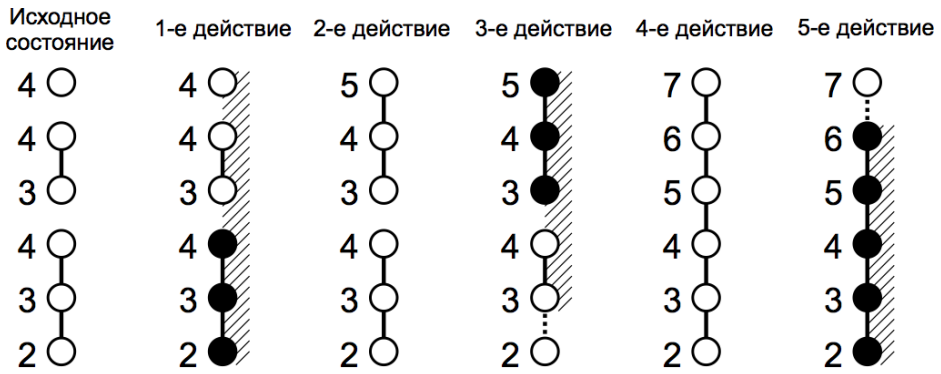

Иллюстрация к примеру. Пунктиром выделены сильные связи, которые разрушаются на время действия второго типа. Для каждого действия второго типа выделены отрезок запроса и самая длинная цепочка.

Задача K. Маршрутное такси

Имя входного файла: `taxibus.in`
Имя выходного файла: `taxibus.out`

Жители Флатландии обычно передвигаются по городу на таком виде общественного транспорта, как маршрутное такси — он быстрый, удобный и относительно недорогой.

К сожалению, есть одна проблема — производители автобусов для маршрутных такси спроектировали их так, что пассажиры испытывают большие неудобства, пробираясь друг мимо друга к свободным местам в автобусе.

Спиридон — водитель маршрутного такси, который уже давно работает на своем маршруте и знает всех пассажиров, проезжающих на его автобусе изо дня в день. Поэтому, он решил спланировать рассадку пассажиров таким образом, чтобы минимизировать количество прохождений пассажиров друг мимо друга.

Во время поездки пассажиры не перемещаются по автобусу, то есть занимают одно и то же место все время.

Формат входного файла

Первая строка входного файла содержит n ($1 \le n \le 100000$) — количество пассажиров автобуса.

Следующие n строк содержат по паре чисел a_i, b_i ($1 \le a_i < b_i \le 2n$)— номера остановок, на которых входят входит и выходит i-ый пассажир.

На каждой остановке входит или выходит не более одного человека.

Формат выходного файла

В первую строку входного файла выведите одно число — минимальное количество прохождений людей друг мимо друга.

Во вторую строку выведите n чисел, i-ое из которых — номер места, на которое должен сесть i-ый пассажир. Места нумеруются целыми числами от 1 до 100000.

Примеры

taxibus.in	taxibus.out
2 1 4 2 3	0 2 1
5 1 8 3 6 2 4 9 10 5 7	2 10 2 4 1 1

Задача L. Разрезание графа

Имя входного файла: `cutting.in`
Имя выходного файла: `cutting.out`

Дан неориентированный граф. Над ним в заданном порядке производят операции следующих двух типов:

- `cut` — разрезать граф, то есть удалить из него ребро;
- `ask` — проверить, лежат ли две вершины графа в одной компоненте связности.

Известно, что после выполнения всех операций типа `cut` рёбер в графе не осталось. Найдите результат выполнения каждой из операций типа `ask`.

Формат входного файла

Первая строка входного файла содержит три целых числа, разделённые пробелами — количество вершин графа n, количество рёбер m и количество операций k ($1 \leqslant n \leqslant 50\,000$, $0 \leqslant m \leqslant 100\,000$, $m \leqslant k \leqslant 150\,000$).

Следующие m строк задают рёбра графа; i-ая из этих строк содержит два числа u_i и v_i ($1 \leqslant u_i, v_i \leqslant n$), разделённые пробелами — номера концов i-го ребра. Вершины нумеруются с единицы; граф не содержит петель и кратных рёбер.

Далее следуют k строк, описывающих операции. Операция типа `cut` задаётся строкой "`cut` u v" ($1 \leqslant u, v \leqslant n$), которая означает, что из графа удаляют ребро между вершинами u и v. Операция типа `ask` задаётся строкой "`ask` u v" ($1 \leqslant u, v \leqslant n$), которая означает, что необходимо узнать, лежат ли в данный момент вершины u и v в одной компоненте связности. Гарантируется, что каждое ребро графа встретится в операциях типа `cut` ровно один раз.

Формат выходного файла

Для каждой операции `ask` во входном файле выведите на отдельной строке слово "`YES`", если две указанные вершины лежат в одной компоненте связности, и "`NO`" в противном случае. Порядок ответов должен соответствовать порядку операций `ask` во входном файле.

Пример

cutting.in	cutting.out
3 3 7 1 2 2 3 3 1 ask 3 3 cut 1 2 ask 1 2 cut 1 3 ask 2 1 cut 2 3 ask 3 1	YES YES NO NO

Задача M. Ребра добавляются, граф растет

Имя входного файла: `addedge.in`
Имя выходного файла: `addedge.out`

В неориентированный граф последовательно добавляются новые ребра. Изначально граф пустой. После каждого добавления нужно говорить, является ли текущий граф двудольным.

Формат входного файла

На первой строке n — количество вершин, m — количество операций «добавить ребро». Следующие m строк содержат пары чисел от 1 до n — описание добавляемых ребер.

Формат выходного файла

Выведите в строчку m нулей и единиц. i-й символ должен быть равен единице, если граф, состоящий из первых i ребер, является двудольным.

Примеры

addedge.in	addedge.out
3 3 1 2 2 3 3 1	110