

Задача А. Фаброзавры-дизайнеры

Имя входного файла: `fabro.in`
Имя выходного файла: `fabro.out`
Ограничение по времени: 3 секунды
Ограничение по памяти: 64 мегабайта

Фаброзавры известны своим тонким художественным вкусом и увлечением ландшафтным дизайном. Они живут около очень живописной реки и то и дело перестраивают тропинку, идущую вдоль реки: либо насыпают дополнительной земли, либо срывают то, что есть. Для того, чтобы упростить эти работы, они поделили всю тропинку на горизонтальные участки, пронумерованные от 1 до N , и их переделки устроены всегда одинаково: они выбирают часть дороги от L -ого до R -ого участка (включительно) и изменяют (увеличивают или уменьшают) высоту на всех этих участках на одну и ту же величину (если до начала переделки высоты были разными, то и после переделки они останутся разными).

Поскольку, как уже говорилось, у фаброзавров тонкий художественный вкус, каждый из них считает, что их река лучше всего выглядит с определенной высоты. Поэтому им хочется знать, есть ли поблизости от их дома место на тропинке, где высота на их взгляд оптимальна. Помогите им в этом разобраться.

Формат входных данных

Первая строка входного файла содержит два числа N и M — длину дороги и количество запросов соответственно ($1 \leq N, M \leq 10^5$). На второй строке содержатся N чисел, разделенных пробелами — начальные высоты соответствующих частей дороги; высоты не превосходят 10^4 по модулю. В следующих M строках содержатся запросы по одному на строке.

Запрос $+ L R X$ означает, что высоту частей дороги от L -ой до R -ой (включительно) нужно изменить на X . При этом $1 \leq L \leq R \leq N$, а $|X| \leq 10^4$.

Запрос $? L R X$ означает, что нужно проверить, есть ли между L -ым и R -ым участками (включая эти участки) участок, где дорога проходит точно на высоте X . Гарантируется, что $1 \leq L \leq R \leq N$, а $|X| \leq 10^9$.

Формат выходных данных

На каждый запрос второго типа нужно вывести в выходной файл на отдельной строке одно слово «YES» (без кавычек), если нужный участок существует, и «NO» в противном случае.

Примеры

<code>fabro.in</code>	<code>fabro.out</code>
10 5	NO
0 1 1 3 3 3 2 0 0 1	YES
? 3 5 2	YES
+ 1 4 1	
? 3 5 2	
+ 7 10 2	
? 9 10 3	

Задача В. Варенье

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

Мальш и Карлсон решили пойти на прогулку. Они знают, что прогулка будет совсем скучной, если перед ней не опустошить несколько банок варенья.

Мальш достал из кладовки N банок варенья и выставил их в ряд. В банке номер i содержится ровно a_i грамм варенья. Карлсон немного подумал и решил, что в некоторых банках недостаточно варенья, и что в банке номер i должно быть хотя бы b_i грамм варенья.

Выходить из этой ситуации Карлсон хочет в M этапов. На каждом этапе он выбирает числа l , r , x и y , а затем выполняет следующие операции: в банку номер l он добавляет x грамм варенья, в банку номер $l + 1 - x + y$ грамм варенья, в банку номер $l + 2 - x + 2 \cdot y$, и так далее. В банку номер r наш герой добавит $x + y \cdot (r - l)$ грамм варенья.

Мальшу хочется определить для каждой банки i наименьший номер операции, после которой в ней станет хотя бы b_i грамм варенья. Помогите Мальшу: найдите соответствующее число для каждой банки.

Формат входных данных

В первой строке входного файла задано одно число N ($1 \leq n \leq 10^5$) — количество банок. Во второй строке заданы N чисел a_i ($0 \leq a_i \leq 2 \cdot 10^9$) — изначальное количество варенья в банке номер i . В третьей строке заданы N чисел b_i ($0 \leq b_i \leq 2 \cdot 10^9$) — минимальное количество варенья, которое должно быть в банке номер i .

В четвертой строке задано M ($0 \leq M \leq 10^5$) — число этапов добавления варенья в банки, которые выполнит Карлсон. В следующих M строках описаны сами этапы в хронологическом порядке. Каждый этап задан четырьмя числами l , r , x и y ($1 \leq l \leq r \leq N$, $0 \leq x, y \leq 3 \cdot 10^5$).

Формат выходных данных

Выведите N чисел в одной строке, разделенные пробелом. Число номер i должно быть равно нулю, если в банке номер i изначально было достаточно варенья, номеру этапа, после которого в ней станет хотя бы b_i варенья, или -1 , если даже после выполнения всех этапов, в этой банке будет недостаточно варенья. Этапы нумеруются с единицы.

Примеры

jam.in	jam.out
5	1 2 0 3 -1
5 4 4 2 1	
7 7 4 7 7	
3	
1 2 2 0	
2 5 1 1	
3 4 2 2	

Задача С. Инверсии на отрезке

Имя входного файла: `riq.in`
Имя выходного файла: `riq.out`
Ограничение по времени: 4 секунды
Ограничение по памяти: 64 мегабайта

Дана перестановка из n элементов.

Ответьте на m запросов про число инверсий для подотрезка перестановки от l до r .

Инверсией называется пара индексов i, j такая, что $i < j$ и $a_i > a_j$, где a_i - это i -ый элемент перестановки.

Формат входных данных

В первой строке задано число n ($1 \leq n \leq 10^5$).

Во второй строке задана перестановка из n элементов (элементы перестановки — попарно различные целые числа от 1 до n).

В третьей строке задано число m ($1 \leq m \leq 10^5$).

В последующих m строках содержится по два числа, l и r , границы запроса ($1 \leq l, r \leq n$).

Формат выходных данных

Выведите m строк, ответы на данные запросы.

Примеры

<code>riq.in</code>	<code>riq.out</code>
5	2
4 5 2 3 1	2
3	8
1 3	
3 5	
1 5	
6	1
5 2 4 3 1 6	4
4	8
4 6	8
2 5	
1 5	
1 6	

Задача D. Мощные юнги

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

Имеется список из n юнг, для каждого из которых известен его рост a_1, a_2, \dots, a_n . Рассмотрим некоторый его подсписок a_l, a_{l+1}, \dots, a_r , где $1 \leq l \leq r \leq n$, и для каждого натурального числа s обозначим через K_s число юнг с ростом s в этом подсписке. Назовем *мощностью* подсписка сумму произведений $K_s \cdot K_s \cdot s$ по всем различным натуральным s . Так как количество различных чисел в массиве конечно, сумма содержит лишь конечное число ненулевых слагаемых.

Необходимо вычислить мощности каждого из t заданных подсписков.

Формат входных данных

Первая строка содержит два целых числа n и t ($1 \leq n, t \leq 200000$) — длина списка и количество запросов соответственно.

Вторая строка содержит n натуральных чисел a_i ($1 \leq a_i \leq 10^6$) — рост юнг.

Следующие t строк содержат по два натуральных числа l и r ($1 \leq l \leq r \leq n$) — индексы левого и правого концов соответствующего подсписка.

Формат выходных данных

Выведите t строк, где i -ая строка содержит единственное натуральное число — мощность подсписка i -го запроса.

Примеры

<code>power.in</code>	<code>power.out</code>
3 2	3
1 2 1	6
1 2	
1 3	
8 3	20
1 1 2 2 1 3 1 1	20
2 7	20
1 6	
2 7	

Задача Е. Нестабильность сети

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

Вася — системный администратор в большой компании под названием Глюкософт. В сети компании n компьютеров, и какие-то пары компьютеров напрямую соединены сетевыми кабелями, всего таких соединений ровно m , при чем никакой кабель не соединяет компьютер с самим собой, и между любой парой компьютеров не больше одного кабеля.

На каждом компьютере в Глюкософте установлена специальная программа, поддерживаемая разработчиками Глюкософта, под названием СетеБаг. Новые версии этой программы выходят почти каждый день, но система обновлений в компании совершенно непостижима. Так, обновление версии программы происходит не синхронно для всех компьютеров. Это происходит следующим образом — выбирается один компьютер, и на него устанавливается какая-то версия СетеБага. Поскольку система обновлений настроена очень странным образом, вполне возможно, что на компьютер установят не последнюю версию СетеБага, а какую-то другую, например, ту что уже установлена на нем, или даже какую-то более раннюю.

После нескольких месяцев работы, Вася обнаружил, что наиболее частая причина нестабильной работы сети в несовместимости версий СетеБага двух компьютеров в Глюкософте. Если два компьютера, напрямую соединенных кабелем, имеют установленный Сетебаг разных версий, то соединение между этими компьютерами *нестабильно*: попытка передачи данных между ними может вызвать отказ системы. Чем больше нестабильных соединений между компьютерами, тем больше вероятность отказа системы. С другой стороны, обмен данными между компьютерами с одинаковой версией СетеБага обычно не приводит ни к каким ошибкам.

Задача Васи — предотвратить и исправить ошибки в сети Глюкософта. Сделать он это может только если он находится на рабочем месте целый день. Вася хочет быть на работе, когда вероятность отказа системы достаточно высока. Он хочет распланировать свое расписание на следующий год, но оценка вероятностей отказа системы — довольно сложная задача.

Помогите Васе найти число нестабильных соединений между компьютерами после каждого обновления версий СетеБага.

Формат входных данных

В первой строке даны два целых числа n, m — количество компьютеров и соединений между ними ($1 \leq n, m \leq 10^5$).

Во второй строке даны n целых чисел v_1, v_2, \dots, v_n — версии СетеБага, изначально установленные на компьютеры Глюкософта.

В следующих m строках даны пары целых чисел a_i, b_i — номера компьютеров, соединенных i -м кабелем ($1 \leq a_i, b_i \leq n, a_i \neq b_i$). Гарантируется, что никакие два компьютера не соединены больше, чем одним кабелем.

В следующей строке дано целое число q — количество запланированных обновлений версий СетеБага ($1 \leq q \leq 10^5$).

В следующих q строках даны пары целых чисел c_i, v_i — номер компьютера, на котором обновляется версия СетеБага, и новая версия СетеБага, которая установится на компьютер ($1 \leq c_i \leq n, 1 \leq v_i' \leq 10^5$). Все обновления даны в хронологическом порядке, и никакие два обновления не происходят одновременно.

Формат выходных данных

Выведите для каждого запроса изменения одно целое число — количество нестабильных соединений сразу после очередного обновления версии СетеБага.

Примеры

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

Задача F. Substring Query

Имя входного файла: `str-qry.in`
Имя выходного файла: `str-qry.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 512 мегабайт

У Пупы есть n строк S_1, S_2, \dots, S_n . Однажды его друг Лупа пришёл в бухгалтерию и попросил его ответить на q вопросов: сколько строк среди $S_{l_i}, S_{l_i+1}, \dots, S_{r_i}$ содержат P_i как подстроку?

Но в бухгалтерии все перепутали, и ответить на запросы за Пупу придется Вам.

Формат входных данных

На первой строке 2 целых числа n, q ($1 \leq n, q \leq 200\,000$). Каждая из следующих n строк содержит ровно одну строку S_i ($|S_1| + |S_2| + \dots + |S_n| \leq 200\,000$). Последние q строк содержат запросы. Каждый запрос задаётся двумя целыми числами l_i, r_i и строкой P_i . ($1 \leq l_i \leq r_i \leq n, |P_1| + |P_2| + \dots + |P_n| \leq 200\,000$) Все строки состоят из букв «a» и «b».

Формат выходных данных

На каждый вопрос выведите одно число — ответ на вопрос.

Примеры

<code>str-qry.in</code>	<code>str-qry.out</code>
4 2	2
a	2
b	
ab	
bab	
1 3 a	
1 4 ab	

Задача G. Сычи

Имя входного файла: owls.in
Имя выходного файла: owls.out
Ограничение по времени: 5 секунд
Ограничение по памяти: 256 мегабайт

Мурка, не ходи, там сыч
На подушке вышит,
Мурка серый, не мурлычь,
Дедушка услышит.
Няня, не горит свеча,
И скребутся мыши.
Я боюсь того сыча,
Для чего он вышит?

Анна Ахматова

Есть n сычей. Известны пары друзей. Три сыча образуют компанию, если все три попарно дружат. Требуется найти количество компаний сычей.

Формат входных данных

Первая строка входных данных содержит два целых числа n и m — количество сычей и дружеских связей соответственно ($1 \leq n, m \leq 3 \cdot 10^5$). Каждая из следующих m строк содержит по два целых числа от 1 до n — номера сычей, которые дружат. Гарантируется, что никакая пара не указана два раза и сыч не дружит сам с собой.

Формат выходных данных

В единственной строке выходного файла выведите количество компаний сычей.

Примеры

owls.in	owls.out
6 6 1 2 2 3 3 1 4 2 3 4 5 1	2

Задача Н. Безумие и Отвага

Имя входного файла: `heroes.in`
Имя выходного файла: `heroes.out`
Ограничение по времени: 10 секунд
Ограничение по памяти: 512 мегабайт

Многие из нас с детства мечтали создавать компьютерные игры, а для некоторых это даже стало причиной, по которой они начали изучать информатику и программирование. Мишина мечта сбылась, и теперь он работает в известной и уважаемой корпорации «Метель», выпустившей в своё время такие шедевры, как «Искусство войны» и «Звёздное ремесло».

Недавно Миша присоединился к проекту новой ролевой игры «Безумие и отвага». Её ключевой особенностью является возможность на каждом из уровней заново выбирать персонажа для его прохождения.

Перед стартом очередного уровня игроку доступны N героев. Каждый герой характеризуется силой атаки a_i и запасом здоровья b_i . Уровень представляет собой длинную пещеру, содержащую M монстров. Каждый монстр также имеет свою силу атаки c_i и запас здоровья d_i . Зайдя в пещеру, герой сначала сражается с первым монстром, затем, если остаётся жив, сражается со вторым и так далее, пока не погибнет или не дойдёт до конца. Количество жизней героя не восстанавливается между боями, то есть каждую следующую драку он начинает с меньшим запасом здоровья, чем предыдущую.

Бой между монстром и героем состоит в одновременном обмене ударами. Каждый из них, нанося удар, уменьшает запас здоровья противника на величину, равную силе своей атаки. Как только запас здоровья кого-либо из сражающихся становится неположительным, он умирает, и бой прекращается. Обратите внимание, что при такой схеме боя возможна ситуация, когда оба противника погибнут одновременно.

Компания планирует распространять игру бесплатно, получая доход за счёт продажи разнообразных бонусов, реализовать один из которых и поручено Мише. Данный бонус позволяет игроку узнать, сколько монстров убьёт каждый из героев, если игрок выберет именно его для прохождения данного уровня. Так как монстров и героев может быть очень много, Миша столкнулся со сложностями при вычислении необходимых значений и обратился за помощью к вам.

Формат входных данных

В первой строке ввода записаны два целых числа N и M — количество доступных игроку героев и количество монстров в пещере соответственно ($1 \leq N, M \leq 200\,000$).

Следующие N строк описывают героев. Каждая из них содержит два целых числа a_i и b_i , задающих силу атаки и запас здоровья i -го героя ($1 \leq a_i, b_i \leq 10^9$).

Далее следуют M строк, описывающих находящиеся в пещере монстров. Каждое описание состоит из двух целых чисел c_i и d_i , обозначающих параметры i -го монстра ($1 \leq c_i, d_i \leq 200\,000$). Порядок расположения монстров в пещере совпадает с порядком их описания, то есть первым необходимо убить монстра, описанного в строке $N + 2$, а последним — в строке $N + M + 1$.

Формат выходных данных

Выведите N чисел по одному в строке. i -я строка должна содержать ответ для i -го героя.

Примеры

<code>heroes.in</code>	<code>heroes.out</code>
5 3	0
1 2	1
2 2	2
10 10	3
100 10	3
1 100	
2 2	
7 2	
3 20	

Замечание

Бой между первым героем и первым монстром в пещере продлится один ход, после которого герой погибнет, а монстр останется в живых.

Параметры второго героя совпадают с параметрами первого монстра, поэтому они убьют друг друга на первом же ходу боя. Ответ для данного героя равен одному.

Если игрок выберет для прохождения уровня третьего героя, то после боя с первым монстром его запас здоровья будет равен восьми, а после боя со вторым — единице. Для убийства третьего монстра ему необходимо сделать два удара, но он умрёт после первой же его атаки.

У четвёртого героя столько же жизней, сколько и у третьего, но сила атаки гораздо больше, поэтому он пройдёт уровень полностью, хотя и погибнет в последней драке.

Пятый герой обладает минимально возможной силой атаки, но при этом у него большой запас здоровья, поэтому он сможет пройти весь уровень и остаться в живых. После первого боя его запас здоровья будет равен 96, после второго — 82, а в конце игры останется только 22.