Задача A. Быстрое прибавление

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	5 секунд
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 ($1 \leqslant m, q \leqslant 2^{24}$). 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 B. Разреженные таблицы

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

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 ($1 \leqslant n \leqslant 10^5$, $1 \leqslant m \leqslant 10^7$) и a_1 ($0 \leqslant a_1 < 16\,714\,589$) — количество элементов в массиве, количество запросов и первый элемент массива соответственно. Вторая строка содержит два натуральных числа u_1 и v_1 ($1 \leqslant u_1, v_1 \leqslant n$) — первый запрос.

Элементы $a_2, a_3, \ldots, a_n$ 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:

$$u_{i+1} = \big((17 \cdot u_i + 751 + ans_i + 2i) \bmod n\big) + 1,$$
$$v_{i+1} = \big((13 \cdot v_i + 593 + ans_i + 5i) \bmod n\big) + 1,$$

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

sparse.in	sparse.out
10 8 12345 3 9	5 3 1565158

Замечание

Пояснение к тесту из примера: запросы и результаты.

a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}
12345	305498	7048017	11694653	1565158	2591019	9471233	570265	13137658	1325095

#	u	v	ans
1	3	9	570265
2	10	1	12345
3	1	2	12345
4	10	10	1325095
5	5	9	570265
6	2	1	12345
7	3	2	305498
8	5	3	1565158

Задача C. 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 ($1 \leqslant N, M \leqslant 10^5$). 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 $1 \leqslant L \leqslant R \leqslant N$, а $|X| \leqslant 10^4$.

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 $1 \leqslant L \leqslant R \leqslant N$, а $|X| \leqslant 10^9$.

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

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

Задача D. Варенье

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

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

Малыш 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.

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

Формат входных данных

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

В четвертой строке задано M ($0 \leqslant M \leqslant 10^5$) — 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 ($1 \leqslant l \leqslant r \leqslant N$, $0 \leqslant x, y \leqslant 3\cdot 10^5$).

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 5 4 4 2 1 7 7 4 7 7 3 1 2 2 0 2 5 1 1 3 4 2 2	1 2 0 3 -1

Задача E. Друзья и последовательности

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

Майк и !Майк соперничают еще со школьных лет, они противоположны во всем что делают, кроме программирования. Сегодня у них возникла проблема, которую сами друзья сами решить не могут, но вместе с вами — кто знает?

Каждый из них знает две последовательности n чисел a и b. По запросу в виде пары целых чисел (l, r) Майк может сразу сообщить значение $\max\limits_{i=l}^{r} a_i$, а !Майк — значение $\min\limits_{i=l}^{r} b_i$.

Предположим, что робот задает им каждый из возможных различных запросов в виде пары целых чисел (l, r) $(1 \leqslant l \leqslant r \leqslant n)$ (то есть он сделает ровно $n(n+1)/2$ запросов) и считает, сколько раз их ответы на один и тот же запрос совпадают, то есть для скольких пар выполняется $\max\limits_{i=l}^{r} a_i = \min\limits_{i=l}^{r} b_i$.

Сколько случаев совпадения посчитает робот?

Формат входных данных

В первой строке содержится единственное целое число n $(1 \leqslant n \leqslant 200\,000)$.

Во второй строке содержатся n целых чисел $a_1, a_2, \ldots, a_n$ $(-10^9 \leqslant a_i \leqslant 10^9)$ — элементы последовательности a.

В третьей строке содержатся n целых чисел $b_1, b_2, \ldots, b_n$ $(-10^9 \leqslant b_i \leqslant 10^9)$ — элементы последовательности b.

Формат выходных данных

Выведите одно целое число — количество совпадений ответов, которые посчитает робот, то есть для скольких пар выполняется $\max\limits_{i=l}^{r} a_i = \min\limits_{i=l}^{r} b_i$.

Примеры

`friends.in`	`friends.out`
6 1 2 3 2 1 4 6 7 1 2 3 2	2
3 3 3 3 1 1 1	0

Замечание

Эта задача взята с Codeforces Round 361. Оригинал: http://codeforces.com/contest/689/problem/D

Задача F. 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 ($1 \leqslant N, M \leqslant 200\,000$).

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 ($1 \leqslant a_i, b_i \leqslant 10^9$).

Далее следуют M строк, описывающих находящихся в пещере монстров. Каждое описание состоит из двух целых чисел c_i и d_i, обозначающих параметры i-го монстра ($1 \leqslant c_i, d_i \leqslant 200\,000$). 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

heroes.in	heroes.out
5 3 1 2 2 2 10 10 100 10 1 100 2 2 7 2 3 20	0 1 2 3 3

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 первому 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.

Задача G. Река

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

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 входных данных

Первая строка входного файла содержит два целых числа: n и p — исходное количество предприятий ($2 \leqslant n \leqslant 100000$) и номер подзадачи ($0 \leqslant p \leqslant 4$) (считайте его просто так).

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

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

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 выходных данных

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.

Примеры

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

Задача H. Жесть

Имя входного файла:	`sqrtrev.in`
Имя выходного файла:	`sqrtrev.out`
Ограничение по времени:	10 секунд
Ограничение по памяти:	256 мегабайт

Дам массив из N чисел. Нужно уметь обрабатывать 3 типа запросов:

- `get(L, R, x)` — сказать, сколько элементов отрезка массива $[L..R]$ не меньше x.
- `set(L, R, x)` — присвоить всем элементам массива на отрезке $[L..R]$ значение x.
- `reverse(L, R)` — перевернуть отрезок массива $[L..R]$.

Формат входных данных

Число N ($1 \leqslant N \leqslant 10^5$) и массив из N чисел. Далее число запросов M ($1 \leqslant M \leqslant 10^5$) и M запросов. Формат описания запросов предлагается понять из примера. Для всех отрезков верно $1 \leqslant L \leqslant R \leqslant N$. Исходные числа в массиве и числа x в запросах — целые от 0 до 10^9.

Формат выходных данных

Для каждого запроса типа `get` нужно вывести ответ.

Примеры

sqrtrev.in	sqrtrev.out
5 1 2 3 4 5 6 get 1 5 3 set 2 4 2 get 1 5 3 reverse 1 2 get 2 5 2 get 1 1 2	3 1 3 1