Задача A. RSQ

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

Формат входных данных

В первой строке находится число n — размер массива. ($1 \leqslant n \leqslant 500,000$) Во второй строке находится n чисел a_i — элементы массива. Далее содержится описание операций, их количество не превышает $1,000,000$. 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 $(1 \leqslant i \leqslant j \leqslant n)$.

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

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 B. RMQ2

Имя входного файла:	`rmq2.in`
Имя выходного файла:	`rmq2.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Формат входных данных

В первой строке находится число n — размер массива. ($1 \leqslant n \leqslant 10^5$) Во второй строке находится n чисел a_i — элементы массива. Далее содержится описание операций, их количество не превышает $2 \cdot 10^5$. В каждой строке находится одна из следующих операций:

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

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

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 C. Знакочередование

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

Реализуйте структуру данных из n элементов $a_1, a_2 \dots a_n$, поддерживающую следующие операции:

- присвоить элементу a_i значение j;
- найти знакочередующуюся сумму на отрезке от l до r включительно $(a_l - a_{l+1} + a_{l+2} - \dots \pm a_r)$.

Формат входных данных

В первой строке входного файла содержится натуральное число n ($1 \leqslant n \leqslant 10^5$) — длина массива. Во второй строке записаны начальные значения элементов (неотрицательные целые числа, не превосходящие 10^4).

В третьей строке находится натуральное число m ($1 \leqslant m \leqslant 10^5$) — количество операций. В последующих m строках записаны операции:

- операция первого типа задается тремя числами `0 i j` ($1 \leqslant i \leqslant n$, $1 \leqslant j \leqslant 10^4$).
- операция второго типа задается тремя числами `1 l r` ($1 \leqslant l \leqslant r \leqslant n$).

Формат выходных данных

Для каждой операции второго типа выведите на отдельной строке соответствующую знакочередующуюся сумму.

Пример

signchange.in	signchange.out
3 1 2 3 5 1 1 2 1 1 3 1 2 3 0 2 1 1 1 3	-1 2 -1 3

Задача D. Хорошие дни

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

Билл разрабатывает новую математическую теорию, описывающую человеческие эмоции. Его последние исследования посвящены изучению того, насколько хорошие и плохие дни влияют на воспоминания людей о различных периодах жизни.

Недавно Билл придумал методику, которая описывает, насколько хорошим или плохим был день человеческой жизни с помощью сопоставления дню некоторого неотрицательного целого числа. Билл называет это число эмоциональной значимостью этого дня. Чем больше это число, тем лучше этот день. Билл полагает, что значимость некоторого периода человеческой жизни равна сумме эмоциональных значимостей каждого из дней периода, помноженной на минимум эмоциональных значимостей дней этого периода. Эта методика отражает то, что период, который в среднем может быть весьма неплох, бывает испорчен одним плохим днем.

Теперь Билл хочет проанализировать свою собственную жизнь и найти в ней период максимальной значимости. Помогите ему это сделать.

Формат входных данных

Первая строка входного файла содержит число n — количество дней в жизни Билла, которые он хочет исследовать ($1 \leqslant n \leqslant 100\,000$). Оставшаяся часть файла содержит n целых чисел $a_1, a_2, \ldots, a_n$, все в пределах от 0 до 10^6 — эмоциональные значимости дней. Числа во входном файле разделяются пробелами и переводами строки.

Формат выходных данных

В первой строке выходного файла выведите максимальную значимость периода жизни Билла. Во второй строке выведите два числа l и r, означающие, что значимость периода с l-го по r-й день (включительно) в жизни Билла была максимально возможной.

Примеры

feelgood.in	feelgood.out
6 3 1 6 4 5 2	60 3 5

Задача E. RMQ наоборот

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

Рассмотрим массив $a[1..n]$. Пусть $Q(i, j)$ — ответ на запрос о нахождении минимума среди чисел $a[i], \ldots, a[j]$. Вам даны несколько запросов и ответы на них. Восстановите исходный массив.

Формат входных данных

Первая строка входного файла содержит число n — размер массива, и m — число запросов ($1 \leqslant n, m \leqslant 100\,000$). Следующие m строк содержат по три целых числа i, j и q, означающих, что $Q(i, j) = q$ ($1 \leqslant i \leqslant j \leqslant n$, $-2^{31} \leqslant q \leqslant 2^{31} - 1$).

Формат выходных данных

Если искомого массива не существует, выведите строку «`inconsistent`».

В противном случае в первую строку выходного файла выведите «`consistent`». Во вторую строку выходного файла выведите элементы массива. Элементами массива должны быть целые числа в интервале от -2^{31} до $2^{31} - 1$ включительно. Если решений несколько, выведите любое.

Примеры

rmq.in	rmq.out
3 2 1 2 1 2 3 2	consistent 1 2 2
3 3 1 2 1 1 1 2 2 3 2	inconsistent

Задача F. 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 ($1 \leqslant n \leqslant 50000$). 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 $2 \cdot 10^5$.

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, разделенные 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

Задача G. Криптография

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

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 входных данных

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

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 H. Художник

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

Итальянский художник-абстракционист Ф. Мандарино увлекся рисованием одномерных черно-белых картин. Он пытается найти оптимальное местоположение и количество черных участков картины. Для этого он проводит на прямой белые и черные отрезки, и после каждой из таких операций хочет знать количество черных отрезков на получившейся картине и их суммарную длину.

Изначально прямая — белая. Ваша задача — написать программу, которая после каждой из таких операций выводит в выходной файл интересующие художника данные.

Формат входных данных

В первой строке входного файла содержится общее количество нарисованных отрезков ($1 \leqslant n \leqslant 100\,000$). В последующих n строках содержится описание операций. Каждая операция описывается строкой вида $c\ x\ l$, где c — цвет отрезка (`W` для белых отрезков, `B` для черных), а сам отрезок имеет вид $[x; x + l]$, причем координаты обоих концов — целые числа, не превосходящие по модулю 500 000. Длина задается положительным целым числом.

Формат выходных данных

После выполнения каждой из операций необходимо вывести в выходной файл на отдельной строке количество черных отрезков на картине и их суммарную длину, разделенные одним пробелом.

Пример

painter.in	painter.out
7 W 2 3 B 2 2 B 4 2 B 3 2 B 7 2 W 3 1 W 0 10	0 0 1 2 1 4 1 4 2 6 3 5 0 0

Задача I. Стена

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

Джан-Джи строит стену из кирпичей одинакового размера. Стена состоит из n столбцов кирпичей, пронумерованных слева направо от 0 до $(n-1)$. Высотой столбца называется количество кирпичей в нем. У столбцов могут быть разные высоты.

Джан-Джи строит стену так. Сначала ни в одном из столбцов нет кирпичей. Далее Джан-Джи выполняет k действий, каждое из которых может быть действием *добавления* или *удаления* кирпичей. Строительство считается законченным, когда выполнены все k действий. Перед каждым действием Джан-Джи выбирает интервал из последовательно стоящих столбцов и высоту h. После этого он выполняет одно из следующих действий:

- действие *добавления*: Джан-Джи добавляет кирпичи в столбцы из выбранного интервала, высота которых меньше чем h, так, чтобы она стала равной h. Со столбцами, высота которых не меньше, чем h, он ничего не делает;
- действие *удаления*: Джан-Джи убирает кирпичи из столбцов из выбранного интервала, высота которых больше чем h, так, чтобы она стала равной h. Со столбцами, высота которых не больше, чем h, он ничего не делает.

Требуется определить конечную форму стены.

Формат входных данных

Первая строка содержит два натуральных числа n и k ($1 \leqslant n \leqslant 2\,000\,000$, $1 \leqslant k \leqslant 500\,000$) — количество столбцов и количество действий.

Следующие k строк содержат описания действий. В каждой строке записано четыре числа. Первое из них, число t обозначает тип действия: 1, если это действие добавления кирпичей, и 2, если это действие удаления кирпичей. Следующие два числа l и r ($0 \leqslant l \leqslant r \leqslant n-1$) задают интервал действия: действие начинается со столбца l и заканчивается столбцом r. Четвертое число h ($0 \leqslant h \leqslant 100\,000$) — высота действия.

Формат выходных данных

Выведите n строк: в i-й строке количество количество кирпичей в $(i-1)$-м столбце после того, как все действия будут выполнены.

Система оценки

Подзадача	Баллы	n	k	Комментарий
1	8	$1 \leqslant n \leqslant 10\,000$	$1 \leqslant k \leqslant 5\,000$	нет дополнительных ограничений
2	24	$1 \leqslant n \leqslant 100\,000$	$1 \leqslant k \leqslant 500\,000$	все действия добавления будут до действий удаления
3	29	$1 \leqslant n \leqslant 100\,000$	$1 \leqslant k \leqslant 500\,000$	нет дополнительных ограничений
4	39	$1 \leqslant n \leqslant 2\,000\,000$	$1 \leqslant k \leqslant 500\,000$	нет дополнительных ограничений

Примеры

стандартный ввод	стандартный вывод
10 3 1 3 4 91220 1 5 9 48623 2 3 5 39412	0 0 0 39412 39412 39412 48623 48623 48623 48623
10 6 1 1 8 4 2 4 9 1 2 3 6 5 1 0 5 3 1 2 2 5 2 6 7 0	3 4 5 4 3 3 0 0 1 0

Задача J. 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 $l \leqslant i \leqslant r$ и $x \leqslant f[i] \leqslant y$.

Формат входных данных

На первой строке число q ($1 \leqslant q \leqslant 2^{17}$). 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