Задача A. Хорошие дни

Имя входного файла:	`feelgood.in`
Имя выходного файла:	`feelgood.out`
Ограничение по времени:	2 секунды
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 входного файла

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 выходного файла

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 B. Мега-инверсии

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

Инверсией в перестановке $p_1, p_2, \ldots, p_N$ называется пара (i, j) такая, что $i < j$ и $p_i > p_j$. Назовём *мега-инверсией* в перестановке $p_1, p_2, \ldots, p_N$ тройку (i, j, k) такую, что $i < j < k$ и $p_i > p_j > p_k$. Напишите алгоритм для быстрого подсчёта количества мега-инверсий в перестановке.

Формат входного файла

Первая строка входного файла содержит целое число N ($1 \leqslant N \leqslant 100\,000$). Следующие N чисел описывают перестановку: $p_1, p_2, \ldots, p_N$ ($1 \leqslant p_i \leqslant N$), все p_i попарно различны. Числа разделяются переводами строк.

Формат выходного файла

Единственная строка выходного файла должна содержать одно число, равное количеству мега-инверсий в перестановке $p_1, p_2, \ldots, p_N$.

Примеры

mega.in	mega.out
4 4 3 2 1	4

Задача C. Окна

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

В первой строке входного файла записано число окон n ($1 \leqslant n \leqslant 50\,000$). 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 10^6.

Формат выходного файла

В первой строке выходного файла выведите максимальное число окон, покрывающих какую-либо из точек в данной конфигурации. Во второй строке выведите два целых числа, разделенных пробелом — координаты точки, покрытой максимальным числом окон. Окна считаются замкнутыми, т. е. покрывающими свои граничные точки.

Примеры

windows.in	windows.out
2 0 0 3 3 1 1 4 4	2 1 3
1 0 0 1 1	1 0 1
4 0 0 1 1 0 1 1 2 1 0 2 1 1 1 2 2	4 1 1
5 0 0 1 1 0 1 1 2 0 0 2 2 1 0 2 1 1 1 2 2	5 1 1

Задача D. Четвертый этаж

Имя входного файла:	`floor4.in`
Имя выходного файла:	`floor4.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Знаете ли вы, почему четвертый этаж заперт и там не останавливается лифт? Потому что на самом деле четвертый, запертый, этаж, где не останавливается лифт, содержит бесконечное количество комнат, пронумерованных натуральными числами. На этот этаж регулярно приезжают дети, каждый из которых заранее выбрал, в какую комнату он хочет заселиться. Если выбранная комната оказывается свободна, то ребенок занимает ее, в противном случае он занимает первую свободную комнату с бóльшим номером.

Кроме того, некоторые дети уезжают в середине смены. Сразу после отъезда ребенка его комната становится доступна для заселения следующего.

Промоделируйте работу преподавателей, ответственных за четвертый этаж и научитесь быстро сообщать приезжающим детям, какую комнату им следует занимать.

Формат входного файла

Первая строка входного файла содержит натуральное число n — количество прибытий и отъездов, происходящих в течение смены ($1 \leqslant n \leqslant 100\,000$).

Следующие n строк содержат информацию об ЛКШатах. Число $a > 0$ обозначает, что приехал школьник, желающий занять комнату номер a ($1 \leqslant a \leqslant 100\,000$). Число $a < 0$ обозначает, что из комнаты номер $|a|$ уехал школьник. (Гарантируется, что эта комната не была пуста).

Формат выходного файла

Для каждого приезжающего школьника выведите одно натуральное число — номер комнаты, в которую он поселится.

Примеры

floor4.in	floor4.out
6 5 5 5 -6 5 5	5 6 7 6 8

Задача E. Найм на работу

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

Вам необходимо нанять работников для строительного проекта. Заявление о приёме на работу подали N кандидатов, пронумерованных от 1 до N включительно. Каждый кандидат с номером k требует, чтобы в случае приёма его на работу ему платили не менее чем S_k долларов. Также для каждого кандидата с номером k известен его уровень квалификации Q_k. Положение о строительной деятельности требует, чтобы вы платили работникам пропорционально их уровню квалификации относительно друг друга. Например, если вы нанимаете двух работников A и B таких что $Q_A = 3Q_B$, то вы обязаны платить работнику A ровно в три раза больше, чем вы платите работнику B. Вам разрешается платить работникам нецелое число денег. Более того, разрешается даже платить количество денег, которое не может быть записано с помощью конечного числа десятичных цифр, такое как треть или шестую долю доллара.

В вашем распоряжении есть W долларов, и вы хотите нанять как можно больше рабочих. Вы решаете кого нанимать и сколько им платить, но вы должны удовлетворить как требованиям работников о минимальном жаловании, так и требованиям положения о строительной деятельности. Естественно, что вам требуется уложиться в бюджет, равный W долларам.

Для данного строительного проекта уровень квалификации работников не имеет значения. Вы заинтересованы только в том, чтобы нанять как можно больше работников независимо от их уровня квалификации. Однако, если есть несколько способов достичь цели, то вы хотите выбрать такой, чтобы общая сумма денег, которую вы заплатите работникам, была как можно меньше. Если и этого можно достичь несколькими способами, то нет никакого различия между этими способами, и вас удовлетворит любой из них.

Напишите программу, которая по заданным требованиям к жалованию и уровням квалификации кандидатов, а также количеству денег, которое у вас есть, определяет, каких кандидатов вам требуется нанять. Вы должны нанять как можно больше из них и при этом потратить как можно меньше денег, соблюдая требования положения о строительной деятельности, описанные выше.

Формат входного файла

Ваша программа должна читать из стандартного потока ввода следующие данные:

- Первая строка входного файла содержит два целых числа N и W, разделённые пробелом ($1 \leqslant N \leqslant 500\,000$, $1 \leqslant W \leqslant 10\,000\,000\,000$).
- Следующие N строк описывают кандидатов, по одному кандидату на каждую строку.

k-я строка из них описывает кандидата с номером k и содержит два целых числа S_k и Q_k, разделённых пробелом ($1 \leqslant Q_k,\ S_k \leqslant 20\,000$).

Максимальное значение W не может быть представлено 32-битным типом данных. Вам необходимо использовать 64-битный тип данных, такой как `long long` в C/C++ или `int64` в Pascal, чтобы значение W можно было сохранить в одной переменной.

Формат выходного файла

Ваша программа должна вывести в стандартный поток вывода следующие данные:

- Первая строка должна содержать одно целое число H – количество работников, которых вы принимаете на работу.
- Следующие H строк должны содержать список номеров кандидатов в произвольном порядке, которых вы выбрали для найма на работу (различные целые числа от 1 до N), по одному в каждой строке.

Примеры

hiring.in	hiring.out
4 100 5 1000 10 100 8 10 20 1	2 2 3

Задача F. Без сказок

Имя входного файла:	`minsumseg.in`
Имя выходного файла:	`minsumseg.out`
Ограничение по времени:	5 секунд
Ограничение по памяти:	64 мегабайта

У этой задачи нет легенды. Вам дана последовательность из N целых чисел и M запросов одного из двух типов:

- *change* ps, val — заменить число стоящее на позиции ps числом val.
- *get* l, r — найти подотрезок отрезка $[l, r]$ с максимальной суммой.

Обратите внимание на факт, что по определению, пустой отрезок является подотрезком любого отрезка.

Формат входного файла

Первая строка содержит два целых положительных числа N и M не превосходящих 300 000. Следующая строка содержит N целых чисел - изначальную последовательность. Следующие M строк содержат запросы в формате описанном в условии. Гарантируется, что все запросы корректны и все значения в последовательности в любой момент не превосходят по модулю 10^9. Используется индексация от 1.

Формат выходного файла

Для каждого запроса *get* выведите одно число — сумму чисел на подотрезке являющемся ответом на данный запрос.

Примеры

minsumseg.in	minsumseg.out
4 2 -5 2 -1 2 get 1 2 get 1 4	2 3