

Содержание

Must have	2
Задача 7A. Количество инверсий [0.5 sec, 256 mb]	2
Задача 7B. Сила с тобой, Люк [5.5 sec, 256 mb]	3
Обязательные задачи	4
Задача 7BV. Сила с тобой, Люк [2.2 sec, 256 mb]	4
Задача 7C. Мега-инверсии [0.2 sec, 256 mb]	5
Задача 7D. Миллиардеры [0.8 sec, 256 mb]	6
Задача 7E. Гонка с дозарядкой [2 sec, 256 mb]	8
Дополнительные задачи	9
Задача 7F. Точки в пространстве [0.7 sec, 256 mb]	9
Задача 7G. К минимумов на отрезке [3.5 sec, 256 mb]	10
Задача 7H. Самая дальняя [1.5 sec, 256 mb]	11

Обратите внимание, входные данные лежат в **стандартном потоке ввода** (он же stdin), вывести ответ нужно в **стандартный поток вывода** (он же stdout).

Обратите внимание на GNU C++ компиляторы с суффиксом inc.

Подни можно пользоваться **дополнительной библиотекой** (optimization.h).

То есть, использовать быстрый ввод-вывод: **пример про числа и строки**.

И быструю аллокацию памяти (ускоряет vector-set-map-весь-STL): **пример**.

Для тех, кто хочет разобраться, как всё это работает.

Короткая версия быстрого ввода-вывода (**тык**) и короткая версия аллокатора (**тык**).

Must have

Задача 7А. Количество инверсий [0.5 sec, 256 mb]

Дан массив случайных целых чисел, нужно найти количество инверсий.

Формат входных данных

На первой строке числа n ($1 \leq n \leq 1\,000\,000$) — размер массива и m ($1 \leq m \leq 2^{24}$ числа в массиве от 0 до $m - 1$). На второй строке пара целых чисел a, b от 1 до 10^9 , используемая в генераторе случайных чисел.

```
1. uint32_t cur = 0; // беззнаковое 32-битное число
2. uint32_t nextRand24() {
3.     cur = cur * a + b; // вычисляется с переполнениями
4.     return cur >> 8; // число от 0 до  $2^{24} - 1$ .
5. }
```

Элементы массива генерируются последовательно. $x_i = \text{nextRand24()} \% m$;

Формат выходных данных

Выведите количество инверсий

Примеры

stdin	stdout
20 5 19 18	63

Замечание

Сгенерированный массив: 01142210424031343330.

Подсказка по решению

Напишите merge sort ;-)

Запрещается пользоваться деревьями Фенвика, отрезков, treap и т.д.

Задача 7B. Сила с тобой, Люк [5.5 сек, 256 mb]

Дан массив a из n чисел, нужно научиться обрабатывать запросы двух типов.

- `change(i, y)` — сделать a_i равным y .
- `intget ()` — вернуть индекс i такой, что a_i встречается в массиве наименьшее возможное число раз. Если таких i несколько, вернуть минимально возможный.

Все индексы, встречающиеся в задаче, нумеруются с нуля.

Формат входных данных

На первой строке размер массива n .

На второй строке сам массив — n целых чисел от 0 до 10^9-1 .

На третьей строке число запросов q .

Следующие q строк содержат запросы в формате «?» (`get`) и «= i y » (`change`).

Ограничения: $n, q \leq 300\,000$.

Формат выходных данных

На каждый запрос `get` выведите на отдельной строке ответ.

Примеры

stdin	stdout
4	0
1 2 3 4	2
5	0
?	
= 0 2	
?	
= 3 3	
?	
6	2
1 1 2 1 1 0	2
3	
?	
= 5 2	
?	

Замечание

Это задача про STL!

Мы проходим структуры данных. Очень важно уметь пользоваться стандартными.

В C++: STL есть много полезного. Вот напоминание того, чем уже пора уметь пользоваться.

<http://acm.math.spbu.ru/~sk1/examples/c++/stl/sample06-data-structures.cpp.html>

По этой задаче можно получить ОК за 0.750 сек, используя только контейнеры C++: STL.

Подсказка по решению

Поскольку 8.5 мегабайт нельзя прочитать из файла мгновенно (как и записать 1 мегабайт данных), используйте максимально быстрые ввод вывод.

Обязательные задачи

Задача 7BB. Сила с тобой, Люк [2.2 sec, 256 mb]

Ровно та же задача. С TL пожестче.

Задача 7С. Мега-инверсии [0.2 сек, 256 mb]

Инверсией в перестановке p_1, p_2, \dots, p_N называется пара (i, j) такая, что $i < j$ и $p_i > p_j$. Назовем мега-инверсией в перестановке p_1, p_2, \dots, p_N тройку (i, j, k) такую, что $i < j < k$ и $p_i > p_j > p_k$. Придумайте алгоритм для быстрого подсчета количества мега-инверсий в перестановке.

Формат входных данных

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

Формат выходных данных

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

Примеры

stdin	stdout
4 4 3 2 1	4

Замечание

Разобрана на практике почти полностью ; -)

Задача 7D. Миллиардеры [0.8 sec, 256 mb]

Возможно, вы знаете, что из всех городов мира больше всего миллиардеров живёт в Москве. Но, поскольку работа миллиардера подразумевает частые перемещения по всему свету, в определённые дни какой-то другой город может занимать первую строчку в таком рейтинге. Ваши приятели из ФСБ, ФБР, MI5 и Шин Бет скинули вам списки перемещений всех миллиардеров за последнее время. Ваш работодатель просит посчитать, сколько дней в течение этого периода каждый из городов мира был первым по общей сумме денег миллиардеров, находящихся в нём.

Формат входных данных

В первой строке записано число n — количество миллиардеров ($1 \leq n \leq 10\,000$). Каждая из следующих n строк содержит данные на определённого человека: его имя, название города, где он находился в первый день данного периода, и размер состояния. В следующей строке записаны два числа: m — количество дней, о которых есть данные ($1 \leq m \leq 50\,000$), k — количество зарегистрированных перемещений миллиардеров ($0 \leq k \leq 50\,000$). Следующие k строк содержат список перемещений в формате: номер дня (от 1 до $m-1$), имя человека, название города назначения. Вы можете считать, что миллиардеры путешествуют не чаще одного раза в день, и что они отбывают поздно вечером и прибывают в город назначения рано утром следующего дня. Список упорядочен по возрастанию номера дня. Все имена и названия городов состоят не более чем из 20 латинских букв, регистр букв имеет значение. Состояния миллиардеров лежат в пределах от 1 до 100 миллиардов.

Формат выходных данных

В каждой строке должно содержаться название города и, через пробел, количество дней, в течение которых этот город лидировал по общему состоянию миллиардеров, находящихся в нём. Если таких дней не было, пропустите этот город. Города должны быть отсортированы по алфавиту (используйте обычный порядок символов: ABC...Zabc...z).

Примеры

stdin	stdout
5	Anadyr 5
Abramovich London 15000000000	London 14
Deripaska Moscow 10000000000	Moscow 1
Potantin Moscow 5000000000	
Berezovsky London 2500000000	
Khodorkovsky Chita 1000000000	
25 9	
1 Abramovich Anadyr	
5 Potantin Courchevel	
10 Abramovich Moscow	
11 Abramovich London	
11 Deripaska StPetersburg	
15 Potantin Norilsk	
20 Berezovsky Tbilisi	
21 Potantin StPetersburg	
22 Berezovsky London	

Замечание

Если упорядочить события по времени, то

London : 1
Anadyr : 5
Moscow : 1
London : 13

что означает, что сперва один день Лондон был на первом месте и так далее...

Подсказка по решению

Это задача на STL. Не бойтесь им пользоваться!

`unordered_map<string, int>` поможет (кстати, он сам хеширует строки).

Задача 7E. Гонка с дозарядкой [2 sec, 256 mb]

Есть $Y + 1$ гоночная трасса. i -я трасса – горизонтальный отрезок $(0, i) - (X, i)$. Есть n заправок. j -я заправка представляет собой вертикальный отрезок $(x_j, y_{j1}) - (x_j, y_{j2})$. Проезжая по i -й трассе, машина начинает в точке $(0, i)$ и движется прямолинейно равномерно к точке (X, i) , тратя на каждую единицу расстояния одну единицу бензина. Если в какой-то момент машина проезжает заправку (точка-машина лежит на отрезке-заправке), то бак машины мгновенно заполняется до максимума. Если в какой-то момент бензин закончился, а машина не находится в точке заправки или точке (X, i) , трасса считается не пройденной. Для каждого i от 0 до Y определите, какой минимальный объём бака должна иметь машина, чтобы пройти i -ю трассу. Машина начинает с полным баком.

Формат входных данных

На первой строке целые числа n, Y, X ($1 \leq n, Y, X \leq 200\,000$).

Следующие n строк содержат по три целых числа x_i, y_{i1}, y_{i2} – описания заправок ($0 < x_i < X, 0 \leq y_{i1} < y_{i2} \leq Y$).

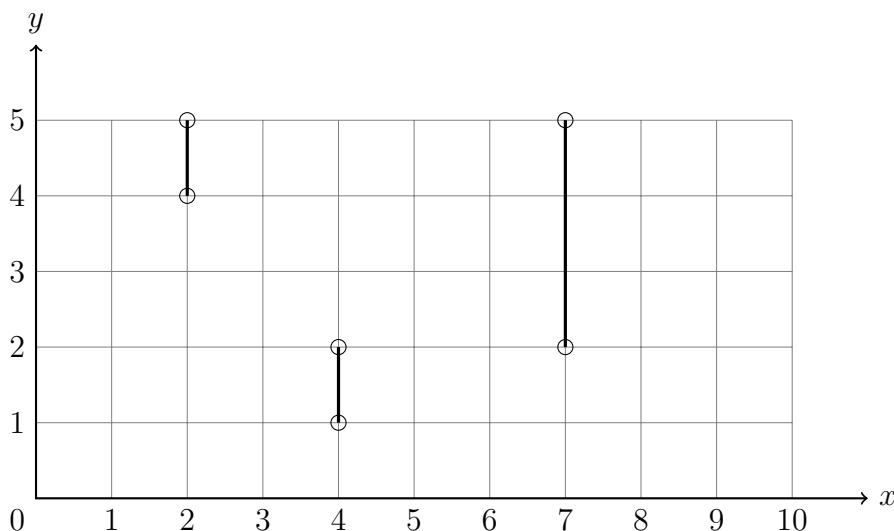
Формат выходных данных

Выведите $Y + 1$ целое число – ответы для всех трасс.

Примеры

stdin	stdout
3 5 10	10
4 1 2	6
7 2 5	4
2 4 5	7
	5
	5

Иллюстрация



Замечание

Запрещается пользоваться структурами данных не из STL. Есть простое решение, использующее только `set<int>`.

Это не простая задача. Главная идея для решения: сканирующая (заметаящая) прямая и события. Заметьте, что сканировать события можно в разных направлениях: по x и по y .

Дополнительные задачи

Задача 7F. Точки в пространстве [0.7 sec, 256 mb]

В пространстве заданы n точек. Вас очень интересует одна величина — минимальное из попарных расстояний между точками. Именно её вы и должны найти.

Формат входных данных

Первая строка ввода содержит единственное число n — количество точек ($2 \leq n \leq 50\,000$). Следующие n строк содержат по три целых числа каждая — координаты точек в пространстве. Гарантируется, что все точки различны. Координаты не превышают 10^6 по абсолютной величине.

Формат выходных данных

В первой строке выведите единственное вещественное число d — минимальное расстояние — с точностью не менее 5 знаков. Во второй строке выведите пару целых чисел — номера точек, расстояние между которыми совпадает с ответом. Если таких пар несколько, выведите любую пару.

Пример

stdin	stdout
5	1.4142135624
1 1 0	4 3
1 0 1	
0 1 1	
0 0 0	
2 2 2	

Замечание

Есть решение за $\mathcal{O}(n \log n)$. Решение за $\mathcal{O}(n \log^2 n)$ также получит ОК.

Задача 7G. К минимумов на отрезке [3.5 сек, 256 mb]

Дан массив a из n целых чисел и q запросов вида «вывести k первых чисел в отсортированной версии отрезка $[l \dots r]$ нашего массива».

Пример: $n = 7$, $a = [6, 1, 5, 2, 4, 3, 1]$, $l = 2$, $r = 4$, $k = 2$. Отрезок $[l \dots r] = [1, 5, 2]$. Его отсортированная версия = $[1, 2, 5]$. Первые 2 числа = $[1, 2]$.

Формат входных данных

На первой строке число n ($1 \leq n \leq 100\,000$).

На второй строке массив a (n целых чисел от 1 до 10^9).

На третьей строке количество запросов q ($1 \leq q \leq 100\,000$).

Следующие q строк содержат тройки чисел $l_i r_i k_i$

$1 \leq l_i \leq r_i \leq n$, $1 \leq k_i \leq \min(r_i - l_i + 1, 10)$

Формат выходных данных

Для каждого из q запросов выведите ответ (k_i чисел) на отдельной строке. Числа внутри одного запроса нужно выводить в порядке возрастания. Для лучшего понимания условия и формата данных смотрите пример.

Тесты в этой задаче состоят из двух групп: $n, q \leq 100\,000$ $l_i \leq l_{i+1}, r_i \leq r_{i+1}$.
 $n, q \leq 30\,000$ l_i и r_i произвольны.

Пример

stdin	stdout
7	1 1 2 3 4 5 6
6 1 5 2 4 3 1	1 2
4	2
1 7 7	1 3
2 4 2	
3 5 1	
5 7 2	

Задача 7H. Самая дальняя [1.5 sec, 256 mb]

Даны N точек на плоскости, нужно уметь обрабатывать следующие запросы:

- `get a b` — возвращает максимум по всем точкам величины $ax + by$.
- `add x y` — добавить точку в множество.

Формат входных данных

Число N ($1 \leq N \leq 10^5$) и N точек. Далее число M ($1 \leq M \leq 10^5$ — количество запросов и собственно запросы. Формат запросов можно посмотреть в примере. Все координаты точек и числа a, b — целые числа, по модулю не превосходящие 10^9 .

Формат выходных данных

На каждый запрос вида `get` выведите одно целое число — максимум величины $ax + by$.

Пример

stdin	stdout
3	1
0 0	0
1 0	1
0 1	1
10	4
get 1 1	4
get -1 -1	1
get 1 -1	1
get -1 1	
add 2 2	
add -2 -2	
get 1 1	
get -1 -1	
get 1 -1	
get -1 1	

Замечание

Это сложная задача. Беритесь за неё только если уверены, что можете построить за $\mathcal{O}(n \log n)$ выпуклую оболочку n точек.