Содержание

04.Base [3/3] 3

Задача 04A. Сумма простая [1.5 sec, 256 mb] 3

Задача 04B. Одномерный почтальон [0.7 sec, 256 mb] 4

Задача 04C. Одномерный финансист [0.2 sec, 256 mb] 5

04.Advanced [3/4] 6

Задача 04D. Быстрое прибавление [4.5 sec, 256 mb] 6

Задача 04E. Мега-инверсии [0.2 sec, 256 mb] 7

Задача 04F. Умножение чисел [0.8 sec, 256 mb] 8

Задача 04G. Ближайшие точки [1 sec, 256 mb] 9

04.Hard [0/1] 10

Задача 04H. Точки в пространстве [0.8 sec, 256 mb] 10

Общая информация:

Вход в контест: http://contest.yandex.ru/contest/2836/
Дедлайн на задачи: 9 дней, до 2016-10-01 23:59.
К каждой главе есть более простые задачи (base), посложнее (advanced), и сложные (hard).
В скобках к каждой главе написано сколько любых задач из этой главы нужно сдать.
Сайт курса: https://compscicenter.ru/courses/algorithms-1/2016-autumn/
Семинары ведут Сергей Копелиович (burunduk30@gmail.com, vk.com/burunduk1) и Алексей Кладов (aleksey.kladov@gmail.com).

В каждом условии указан таймлимит для `C/C++`.
Таймлиминт для `Java` примерно в 2-3 раза больше.
Таймлиминт для `Python` примерно в 6 раз больше.

C++:

Быстрый ввод-вывод.
http://acm.math.spbu.ru/~sk1/algo/input-output/fread_write_export.cpp.html Более подробно про ввод-вывод.
http://acm.math.spbu.ru/~sk1/algo/input-output/cpp_common.html

В некоторых задачах нужен STL, который активно использует динамическую память (set-ы, map-ы) переопределение стандартного аллокатора ускорит вашу программу:
http://acm.math.spbu.ru/~sk1/algo/memory.cpp.html

Java:

Быстрый ввод-вывод.
http://acm.math.spbu.ru/~sk1/algo/input-output/java/java_common.html

04.Base [3/3]

Задача 04A. Сумма простая [1.5 sec, 256 mb]

Вам нужно научиться отвечать на запрос "сумма чисел на отрезке".
Массив не меняется. Запросов много. Отвечать на 1 запрос следует за $O(1)$.

Формат входных данных

Размер массива — n и числа x, y, a_0, порождающие массив a: $a_i = (x \cdot a_{i-1} + y) \bmod 2^{16}$ Далее следуют количество запросов m и числа z, t, b_0, порождающие массив b: $b_i = (z \cdot b_{i-1} + t) \bmod 2^{30}$, $c_i = b_i \bmod n$. i-й запрос — найти сумму на отрезке от $min(c_{2i}, c_{2i+1})$ до $max(c_{2i}, c_{2i+1})$ в массиве a.

Ограничения: $1 \leqslant n \leqslant 10^7, 0 \leqslant m \leqslant 10^7$. Все числа целые от 0 до 2^{16}. t может быть -1.

Формат выходных данных

Выведите сумму всех сумм.

Пример

sum0.in	sum0.out
3 1 2 3 3 1 -1 4	23

Замечание

$a = \{3, 5, 7\}, b = \{4, 3, 2, 1, 0, 2^{30} - 1\}, c = \{1, 0, 2, 1, 0, 0\}$,
запросы $= \{[0, 1], [1, 2], [0, 0]\}$, суммы $= \{8, 12, 3\}$.

Заметим, что вместо того, чтобы брать по модулю 2^{30},
достаточно всё считать в типе `int` и оставлять младшие 30 бит.

В `Java` есть знаковый сдвиг `>>` и беззнаковый `>>>`.
Одна из стандартных ошибок – переполнение типа. Проверьте, что у вас везде `int64`.

Задача 04B. Одномерный почтальон [0.7 sec, 256 mb]

В деревне Печалька живут n человек, их домики расположены ровно на оси абсцисс. Домик i-го человека находится в точке x_i. В деревню приехал и хочет там поселиться почтальон. Координату своего домика y он хочет выбрать так, чтобы суммарное расстояние от него до всех жителей деревни было минимально возможным. То есть

$$\sum_{i=1}^{n} |y - x_i| \to \min$$

Вам дан массив x из n случайных целых чисел. Найдите точку y.

Формат входных данных

На первой строке число n ($1 \leqslant n \leqslant 10^7$). На второй строке пара целых чисел a, b от 1 до 10^9, используемая в генераторе случайных чисел.

```
1. unsigned int cur = 0; // беззнаковое 32-битное число
2. unsigned int nextRand24() {
3.     cur = cur * a + b; // вычисляется с переполнениями
4.     return cur >> 8; // число от 0 до 2^24 - 1.
5. }
6. unsigned int nextRand32() {
7.     unsigned int a = nextRand24(), b = nextRand24();
8.     return (a << 8) ^ b; // число от 0 до 2^32 - 1.
9. }
```

Элементы массива генерируются последовательно. `x`$_i$ `= nextRand32();`

Формат выходных данных

Выведите одно число — минимальное суммарное расстояние от точки y до всех домиков.

Примеры

postman.in	postman.out
6 239 13	8510257371

Замечание

Сгенерированный массив: $12, 130926, 3941054950, 2013898548, 197852696, 2753287507$.

Предполагается решение за линейное время. Без сортировки.
Поиск k-й порядковой статистики: neerc.ifmo.ru/wiki

Задача 04C. Одномерный финансист [0.2 sec, 256 mb]

В деревне Печалька живут n человек, их домики расположены ровно на оси абсцисс. Домик i-го человека находится в точке x_i. В деревню недавно заселился почтальон. Почтальон построил себе домик в такой точке y, что суммарное расстояние от него до всех жителей деревни было минимально возможным. А теперь в деревню приехал финансовый аналитик, который привык не только оптимизировать результат, но и оценивать риски. Посмотрев на опыт почтальона, аналитик заметил, что несмотря на то, что сумма минимальна, есть домики очень далеко от дома почтальона. Финансист учел это и свой дом хочет построить в такой точке z, что

$$\sum_{i=1}^{n}(z - x_i)^2 \to \min$$

С почтальоном финансист не дружит, поэтому расстояние до y в сумме не учитывается. Вам дан массив x из n случайных целых чисел. Найдите точку z.

Формат входных данных

На первой строке число n ($1 \leqslant n \leqslant 10^7$). На второй строке пара целых чисел a, b от 1 до 10^9, используемая в генераторе случайных чисел.

```
1. unsigned int cur = 0; // беззнаковое 32-битное число
2. unsigned int nextRand24() {
3.     cur = cur * a + b; // вычисляется с переполнениями
4.     return cur >> 8; // число от 0 до 2^24 - 1.
5. }
6. unsigned int nextRand32() {
7.     unsigned int a = nextRand24(), b = nextRand24();
8.     return (a << 8) ^ b; // число от 0 до 2^32 - 1.
9. }
```

Элементы массива генерируются последовательно. `x`$_i$ `= nextRand32();`

Формат выходных данных

Выведите координату домика финансиста в виде несократимой дроби с положительным знаменателем.

Примеры

finansist.in	finansist.out
6 230 10	3368129374/3

Замечание

Сгенерированный массив: $9, 1004452, 2338007883, 149525792, 917993446, 3329727166$.

Это очень простая задача. Она разбиралась на паре.

04.Advanced [3/4]

Задача 04D. Быстрое прибавление [4.5 sec, 256 mb]

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

fastadd.in	fastadd.out
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]
```

Есть простое решение за линейное время, нужны только частичные суммы.

Задача 04E. Мега-инверсии [0.2 sec, 256 mb]

Инверсией в перестановке $p_1, p_2, ..., p_N$ называется пара (i, j) такая, что $i < j$ и $p_i > p_j$. Назовем мега-инверсией в перестановке $p_1, p_2, ..., p_N$ 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 ($1 \leqslant N \leqslant 100\,000$). Следующие N чисел описывают перестановку: $p_1, p_2, ..., p_N$ ($1 \leqslant p_i \leqslant N$), 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 $p_1, p_2, ..., p_N$.

Примеры

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

Замечание

Есть несложное решение за $O(n \log n)$ с использованием `MergeSort`.

Задача 04F. Умножение чисел [0.8 sec, 256 mb]

Требуется перемножить два целых неотрицательных числа.

Формат входных данных

В двух строках даны два целых неотрицательных числа в 10-чной системе счисления. Максимальная длина числа $= 2^{18}$.

Формат выходных данных

Выведите в выходной файл произведение.

Пример

mul.in	mul.out
13 100	1300

Замечание

Решение за $\mathcal{O}(n^2)$ на C++ работает 1.35 секунд... Поэтому TL именно такой.

С другой стороны авторское решение алгоритмом Карацубы работает 0.17 секунд ;-)

Задача 04G. Ближайшие точки [1 sec, 256 mb]

Дано несколько точек на плоскости. Выведите наименьшее расстояние, которое достигается между какими-то двумя из них.

Формат входных данных

В первой строке задано число N ($2 \leqslant N \leqslant 200\,000$) — количество точек.

Следующие N строк содержат координаты точек (целые числа от -10^9 до 10^9).

Формат выходных данных

Выведите единственное вещественное число — минимальное расстояние между какими-то двумя из этих точек. Ответ будет считаться корректным, если абсолютная погрешность ответа не будет превышать 10^{-6}.

Примеры

closest.in	closest.out
2 0 0 3 4	5.0
2 7 7 7 7	0.0
4 0 0 5 6 3 4 7 2	2.8284271247461903

Замечание

Лучше писать решение за $O(n \log n)$. Решение за $\mathcal{O}(n \log^2 n)$ также должно получать `OK`. Помните, `sqrt()` – крайне медленная функция!

04.Hard [0/1]

Задача 04H. Точки в пространстве [0.8 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 ($2 \leqslant n \leqslant 50\,000$). 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

points3d.in	points3d.out
5 1 1 0 1 0 1 0 1 1 0 0 0 2 2 2	1.4142135624 4 3

Замечание

Есть решение за $\mathcal{O}(n \log n)$. Используйте метод “разделяй и властвуй”.