Содержание

12.Base [2/2] 3

Задача 12A. Декодирование [0.5 sec, 256 mb] 3

Задача 12B. Снеговики [0.5 sec (1.5 sec), 256 mb] 4

12.Advanced [2/4] 5

Задача 12C. Persistant Array [0.5 sec (1 sec), 256 mb] 5

Задача 12D. CHM [0.7 sec (1.5 sec), 256 mb] 6

Задача 12E. Больше чёрного! [0.5 sec (1 sec), 256 mb] 7

Задача 12F. Ребра добавляются, граф растет [2.5 sec (5 sec), 256 mb] 8

12.Hard [0/2] 9

Задача 12G. Приключение [0.8 sec (1.6 sec), 256 mb] 9

Задача 12H. Менеджер памяти [3.5 sec (6 sec), 256 mb] 10

Общая информация:

Вход в контест: `http://contest.yandex.ru/contest/956/`
Дедлайн на задачи 17-го декабря в 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: `http://compscicenter.ru/courses/algorithms-1/2014-autumn/`
Семинары ведет Сергей Владимирович Копелиович,
контакты: burunduk30@gmail.com, vk.com/burunduk1

В каждом условии 2 таймлимита: для `C/C++` и для `Java, Python`.

12.Base [2/2]

Задача 12A. Декодирование [0.5 sec, 256 mb]

В этой задаче по набору беспрефиксных кодов букв и строке, закодированной с помощью этих кодов так, как это описано в задаче **“Кодирование”**, нужно восстановить исходную строку.

Формат входных данных

В первой строке входного файла заданы два целых числа k и l через пробел — количество различных букв, встречающихся в строке, и размер получившейся закодированной строки, соответственно. В следующих k строках записаны коды букв в формате “`<letter>: <code>`”. Ни один код не является префиксом другого. Буквы могут быть перечислены в любом порядке. В качестве букв могут встречаться лишь строчные буквы латинского алфавита; каждая из этих букв встречается в строке хотя бы один раз. Наконец, в последней строке записана закодированная строка.

Исходная строка и коды всех букв непусты. Заданный код таков, что закодированная строка имеет минимальный возможный размер.

Формат выходных данных

В первой строке выходного файла выведите строку s. Она должна состоять из строчных букв латинского алфавита. Гарантируется, что длина правильного ответа не превосходит 100 000 символов.

Примеры

decoding.in	decoding.out
1 1 a: 0 0	a
4 14 a: 0 b: 10 c: 110 d: 111 01001100100111	abacabad

Задача 12B. Снеговики [0.5 sec (1.5 sec), 256 mb]

Зима. 2012 год. На фоне грядущего Апокалипсиса и конца света незамеченной прошла новость об очередном прорыве в областях клонирования и снеговиков: клонирования снеговиков. Вы конечно знаете, но мы вам напомним, что снеговик состоит из нуля или более вертикально поставленных друг на друга шаров, а клонирование — это процесс создания идентичной копии (клона).

В местечке Местячково учитель Андрей Сергеевич Учитель купил через интернет-магазин «Интернет-магазин аппаратов клонирования» аппарат для клонирования снеговиков. Теперь дети могут играть и даже играют во дворе в следующую игру. Время от времени один из них выбирает понравившегося снеговика, клонирует его и:

- либо добавляет ему сверху один шар;
- либо удаляет из него верхний шар (если снеговик не пустой).

Учитель Андрей Сергеевич Учитель записал последовательность действий и теперь хочет узнать суммарную массу всех построенных снеговиков.

Формат входных данных

Первая строка содержит количество действий n ($1 \leqslant n \leqslant 200\,000$). В строке номер $i + 1$ содержится описание действия i:

- `t m` — клонировать снеговика номер t ($0 \leqslant t < i$) и добавить сверху шар массой m ($0 < m \leqslant 1000$);
- `t ∅` — клонировать снеговика номер t ($0 \leqslant t < i$) и удалить верхний шар. Гарантируется, что снеговик t не пустой.

В результате действия i, описанного в строке $i + 1$ создается снеговик номер i. Изначально имеется пустой снеговик с номером ноль.

Все числа во входном файле целые.

Формат выходных данных

Выведите суммарную массу построенных снеговиков.

Примеры

snowmen.in	snowmen.out
8 0 1 1 5 2 4 3 2 4 3 5 0 6 6 1 0	74

12.Advanced [2/4]

Задача 12C. Persistant Array [0.5 sec (1 sec), 256 mb]

Дан массив (вернее, первая, начальная его версия).

Нужно уметь отвечать на два запроса:

∘ $a_i[j] = x$ — создать из i-й версии новую, в которой j-й элемент равен x, а остальные элементы такие же, как в i-й версии.

∘ `get` $a_i[j]$ — сказать, чему равен j-й элемент в i-й версии.

Формат входных данных

Количество чисел в массиве N ($1 \leqslant N \leqslant 10^5$) и N элементов массива. Далее количество запросов M ($1 \leqslant M \leqslant 10^5$) и M запросов. Формат описания запросов можно посмотреть в примере. Если уже существует K версий, новая версия получает номер $K + 1$. И исходные, и новые элементы массива — целые числа от 0 до 10^9. Элементы в массиве нумеруются числами от 1 до N.

Формат выходных данных

На каждый запрос типа `get` вывести соответствующий элемент нужного массива.

Пример

parray.in	parray.out
6 1 2 3 4 5 6 11 create 1 6 10 create 2 5 8 create 1 5 30 get 1 6 get 1 5 get 2 6 get 2 5 get 3 6 get 3 5 get 4 6 get 4 5	6 5 10 5 10 8 6 30

Задача 12D. CHM [0.7 sec (1.5 sec), 256 mb]

Ваша задача — реализовать **Persistent Disjoint-Set-Union**. Что это значит?
Про **Disjoint-Set-Union**:
Изначально у вас есть n элементов. Нужно научиться отвечать на 2 типа запросов.

- `+ a b` — объединить множества, в которых лежат вершины a и b
- `? a b` — сказать, лежат ли вершины a и b сейчас в одном множестве

Про **Persistent**:
Теперь у нас будет несколько копий (версий) структуры данных **Disjoint-Set-Union**.
Запросы будут выглядеть так:

- `+ i a b` — запрос к i-й структуре, объединить множества, в которых лежат вершины a и b. При этом i-я структура остается не изменной, создается новая версия, ей присваивается новый номер (какой? читайте дальше)
- `? i a b` — запрос к i-й структуре, сказать, лежат ли вершины a и b сейчас в одном множестве

Формат входных данных

На первой строке 2 числа N ($1 \leqslant N \leqslant 10^5$) и K ($0 \leqslant K \leqslant 10^5$) — число элементов и число запросов. Изначально все элементы находятся в различных множествах. Эта изначальная копия (версия) структуры имеет номер 0.

Далее следуют K строк, на каждой описание очередного запроса. Формат запросов описан выше. Запросы нумеруются целыми числами от 1 до K.

При обработки j-го запроса вида `+ i a b`, новая версия получит номер j.

Формат выходных данных

Для каждого запроса вида `? i a b` на отдельной строке нужно вывести `YES` или `NO`.

Пример

snm.in	snm.out
4 7 + 0 1 2 ? 0 1 2 ? 1 1 2 + 1 2 3 ? 4 3 1 ? 0 4 4 ? 4 1 4	NO YES YES YES NO

Задача 12E. Больше чёрного! [0.5 sec (1 sec), 256 mb]

Дана прямоугольная доска $w \times h$, состоящая из квадратов 1×1. Некоторые из квадратов отсутствуют. Нужно покрасить оставшиеся квадраты в чёрный и белый так, чтобы число чёрных было как можно больше, и чтобы никакие два одноцветных квадрата не имели общей стороны.

Формат входных данных

В первой строке ввода записаны числа w и h — размеры доски ($1 \leqslant w, h \leqslant 50$). В следующих h строках записано по w символов: «.» означает, что соответствующий квадрат отсутствует, «#» — обратное.

Формат выходных данных

Выведите доску, раскрашенную указанным образом. Среди всех оптимальных решений выведите лексикографически минимальное. Чёрный квадрат обозначается символом «b», белый — «w».

Примеры

black.in	black.out
3 3 .#. ### .#.	.b. bwb .b.
6 6 #.#.#. .#.#.# #.#.#. .#.#.# #.#.#. .#.#.#	b.b.b. .b.b.b b.b.b. .b.b.b b.b.b. .b.b.b
6 1 ######	bwbwbw
3 4 .#. .#. ### .#.	.w. .b. bwb .b.

Задача 12F. Ребра добавляются, граф растет [2.5 sec (5 sec), 256 mb]

В неориентированный граф последовательно добавляются новые ребра. Изначально граф пустой. После каждого добавления нужно говорить, является ли текущий граф двудольным.

Формат входных данных

На первой строке n — количество вершин, m — количество операций «добавить ребро». Следующие m строк содержат пары чисел от 1 до n — описание добавляемых ребер.

Формат выходных данных

Выведите в строчку m нулей и единиц. i-й символ должен быть равен единице, если граф, состоящий из первых i ребер, является двудольным.

Система оценки

Подзадача 1 (25 баллов) $1 \leqslant n, m \leqslant 1\,000$.
Подзадача 2 (50 баллов) $1 \leqslant n, m \leqslant 50\,000$.
Подзадача 3 (25 баллов) $1 \leqslant n, m \leqslant 300\,000$.

Примеры

addedge.in	addedge.out
3 3 1 2 2 3 3 1	110

12.Hard [0/2]

Задача 12G. Приключение [0.8 sec (1.6 sec), 256 mb]

Теплым весенним днем группа из N школьников-программистов гуляла в окрестностях города Кисловодска. К несчастью, они набрели на большую и довольно глубокую яму. Как это случилось — непонятно, но вся компания оказалась в этой яме.

Глубина ямы равна H. Каждый школьник знает свой рост по плечи h_i и длину своих рук l_i. Таким образом, если он, стоя на дне ямы, поднимет руки, то его ладони окажутся на высоте $h_i + l_i$ от уровня дна ямы. Школьники могут, вставая друг другу на плечи, образовывать вертикальную колонну. При этом любой школьник может встать на плечи любого другого школьника. Если под школьником i стоят школьники $j_1, j_2, \ldots, j_k$, то он может дотянуться до уровня $h_{j1} + h_{j2} + \ldots + h_{jk} + h_i + l_i$.

Если школьник может дотянуться до края ямы (то есть $h_{j1} + h_{j2} + \ldots + h_{jk} + h_i + l_i \geq H$), то он может выбраться из нее. Выбравшиеся из ямы школьники не могут помочь оставшимся. Найдите наибольшее количество школьников, которые смогут выбраться из ямы до прибытия помощи, и перечислите их номера.

Формат входных данных

В первой строке входного файла записано натуральное число N — количество школьников, попавших в яму. Далее в N строках указаны по два целых числа: рост i-го школьника по плечи h_i и длина его рук l_i. В последней строке указано целое число — глубина ямы H.

Формат выходных данных

В первой строке выведите K — максимальное количество школьников, которые смогут выбраться из ямы. Если $K > 0$, то во второй строке выведите их номера в том порядке, в котором они вылезают из ямы. Школьники нумеруются с единицы в том порядке, в котором они заданы во входном файле. Если существует несколько решений, выведите любое.

Система оценки

Подзадача 1 (50 баллов) $n \leqslant 2\,000$ $1 \leqslant l_i, h_i, H \leqslant 10^5$
Подзадача 2 (50 баллов) $n \leqslant 100\,000$ $1 \leqslant l_i, h_i, H \leqslant 10^9$

Пример

advent.in	advent.out
2 10 4 5 2 20	0
6 6 7 3 1 8 5 8 5 4 2 10 5 30	4 5 2 4 1

Задача 12H. Менеджер памяти [3.5 sec (6 sec), 256 mb]

Одно из главных нововведений новейшей операционной системы Indows 7 — новый менеджер памяти. Он работает с массивом длины N и позволяет выполнять три самые современные операции:

- `copy(a, b, l)` — скопировать отрезок длины $[a, a + l - 1]$ в $[b, b + l - 1]$
- `sum(l, r)` — посчитать сумму элементов массива на отрезке $[l, r]$
- `print(l, r)` — напечатать элементы с l по r, включительно

Вы являетесь разработчиком своей операционной системы, и Вы, безусловно, не можете обойтись без инновационных технологий. Вам необходимо реализовать точно такой же менеджер памяти.

Формат входных данных

Первая строка входного файла содержит целое число N $(1 \leqslant N \leqslant 1\,000\,000)$ — размер массива, с которым будет работать Ваш менеджер памяти.

Во второй строке содержатся четыре числа $1 \leqslant X_1, A, B, M \leqslant 10^9 + 10$. С помощью них можно сгенерировать исходный массив чисел $X_1, X_2, \cdots, X_N$. $X_{i+1} = (A * X_i + B) \mod M$

Следующая строка входного файла содержит целое число K $(1 \leqslant K \leqslant 200\,000)$ — количество запросов, которые необходимо выполнить Вашему менеджеру памяти.

Далее в K строках содержится описание запросов. Запросы заданы в формате:

- `cpy` a b l — для операции `copy`
- `sum` l r — для операции `sum` $(l \leqslant r)$
- `out` l r — для операции `print` $(l \leqslant r)$

Гарантируется, что суммарная длина запросов `print` не превышает 3 000. Также гарантируется, что все запросы корректны.

Формат выходных данных

Для каждого запроса *sum* или *print* выведите в выходной файл на отдельной строке результат запроса.

Пример

memory.in	memory.out
6 1 4 5 7 8 out 1 6 sum 1 6 cpy 1 3 2 out 3 4 sum 3 4 cpy 1 2 4 out 1 6 sum 1 6	1 2 6 1 2 6 18 1 2 3 1 1 2 1 2 6 13