Задача A. Быстрая сортировка

Имя входного файла:	`qsort.in`
Имя выходного файла:	`qsort.out`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мегабайта

Отсортируйте данную последовательность, используя алгоритм быстрой сортировки Хоара.

Формат входного файла

В единственной строке входного файла содержится последовательность, содержащая не более чем 10 000 целых чисел.

Формат выходного файла

В единственной строке выходного файла выведите последовательность в неубывающем порядке.

Пример

qsort.in	qsort.out
4 1 4 8 6 6 5	1 4 4 5 6 6 8

Задача B. Электрички

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

На вокзале есть K тупиков, куда прибывают электрички. Этот вокзал является их конечной станцией, поэтому электрички, прибыв, некоторое время стоят на вокзале, а потом отправляются в новый рейс (в ту сторону, откуда прибыли).

Дано расписание движения, в котором указаны события прибытия и отбытия для каждой из электричек в хронологическом порядке. Поскольку вокзал — конечная станция, то электричка может стоять на нём довольно долго, в частности, электричка, которая прибывает раньше другой, отправляться обратно может значительно позднее.

Тупики пронумерованы числами от 1 до K. Когда электричка прибывает, её ставят в свободный тупик с минимальным номером.

Напишите программу, которая по данному расписанию для каждой электрички определит номер тупика, куда прибудет эта электричка.

Формат входного файла

В первой строке вводится число K — количество тупиков ($1 \leqslant K \leqslant 20\,000$). Далее следуют строки, описывающие события прибытия/отбытия электричек. Каждая электричка задаётся своей противоположной конечной станцией — строкой длины не более 15 из латинских букв и знаков подчёркивания. Событие `+city` означает, что прибывает электричка из города `city`, событие `-city` — что эта электричка отправляется обратно. Общее количество электричек, фигурирующих в условии, не более 20 000, для каждой фигурирующей электрички присутствуют оба события.

Считается, что в нулевой момент времени все тупики на вокзале свободны.

Формат выходного файла

Выведите по одному числу на каждую электричку — номер тупика, куда её поставят по прибытии. Если тупиков недостаточно для того, чтобы организовать движение электричек согласно расписанию, выведите два числа: первое должно равняться нулю, а второе — содержать город первой из электричек, которая не сможет прибыть на вокзал.

Примеры

trains.in	trains.out
3 +bologoe +moscow -bologoe +stpetersburg -stpetersburg +samara +saratov -moscow -samara -saratov	bologoe 1 moscow 2 stpetersburg 1 samara 1 saratov 3
2 +kostroma +sudislavl +newvasyuki -sudislavl -kostroma -newvasyuki	0 newvasyuki

Задача C. Очередь с приоритетами

Имя входного файла:	`priority-queue.in`
Имя выходного файла:	`priority-queue.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256мегабайта

Реализуйте структуру данных «очередь с приоритетами», поддерживающую следующие операции:

1. Добавление элемента в очередь.
2. Удаление из очереди элемента с наибольшим приоритетом.

3. Изменение приоритета для произвольного элемента, находящегося в очереди.

Формат входного файла

Программа получает на вход последовательность команд, по одной команде в каждой строке. Общее число команд не превосходит 30 000. Команда может иметь один из следующих форматов:

`ADD id priority` — добавить в очередь новый элемент с идентификатором `id` и приоритетом `priority`. Гарантируется, что в очереди нет элемента с таким идентификатором.

`POP` — удалить из очереди элемент с наибольшим значением приоритета. Если таких элементов несколько, то удаляется один (любой) из них. Гарантируется, что очередь не пуста.

`CHANGE id new_priority` — изменить значение приоритета элемента с идентификатором `id` на значение `new_priority`. Гарантируется, что в очереди есть элемент с таким идентификатором.

Идентификаторы элементов — строки, состоящие из строчных латинских букв длиной не более 10 символов.

В самом начале очередь пуста.

Формат выходного файла

Для каждой команды типа `POP` выведите идентификатор удаленного элемента и — через пробел — значение его приоритета.

Пример

priority-queue.in	priority-queue.out
ADD one 1 ADD two 2 ADD three 3 POP CHANGE one 5 POP POP	three 3 one 5 two 2

Задача D. Монополия

Имя входного файла:	`monopoly.in`
Имя выходного файла:	`monopoly.out`
Ограничение по времени:	20 секунд
Ограничение по памяти:	256 мегабайта

В новом варианте игры «Монополия» появилась возможность объединять несколько предприятий в одно для увеличения приносимого ими дохода. При этом в игре действуют следующие правила:

1. За один ход можно объединить ровно два предприятия в одно. При этом стоимость нового предприятия равна сумме стоимостей двух предприятий до объединения.
2. За совершение операции по объединению предприятий необходимо заплатить налог в размере 5 % от стоимости объединяемых предприятий.

Коля уже заработал в игре много денег и теперь хочет объединить все свои предприятия в одно. Он заметил, что общая сумма уплаченного налога зависит от того, в каком порядке будут объединяться предприятия. Например, пусть у Коли есть четыре предприятия стоимостью 10, 11, 12 и 13. Если Коля сначала объединит предприятия 10 и 11 (это обойдётся ему в \$1.05), потом результат — с 12 (\$1.65), и затем — с 13 (\$2.3), то всего он заплатит \$5. Если же сначала отдельно объединить 10 и 11 (\$1.05), потом — 12 и 13 (\$1.25) и, наконец, объединить два полученных предприятия (\$2.3), то в итоге он заплатит лишь \$4.6.

Помогите Коле определить минимальную сумму денег, необходимую для объединения всех его предприятий в одно.

Формат входного файла

В единственной строке входного файла записано N натуральных чисел ($2 \leqslant N \leqslant 200\,000$), каждое из которых не превосходит 10 000 — стоимости Колиных предприятий.

Формат выходного файла

В выходной файл выведите минимальную сумму денег необходимую для объединения всех Колиных предприятий в одно.

Примеры

monopoly.in	monopoly.out
10 11 12 13	4.60
1 1	0.10