Задача A. Словарь

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

Кхамул решил составить толковый словарь орочей речи. Он последовательно приставал к N оркам, каждый говорил ему слово по-орочьи, и Кхамул записывал это слово. Некоторые из записанных слов совпали. Определений к словам Кхамул решил не писать: все равно каждый орк знает их значение. Таким образом, все, что вам осталось (Кхамулу эта затея уже надоела) — отсортировать список слов в лексикографическом порядке и, при наличии одинаковых слов, оставить из них только одно.

Формат входных данных

В первой строчке находится целое число N — число слов ($1 \leqslant N \leqslant 100$). В следующих N строчках находятся слова орочьего языка, состоящие только из больших букв. Длины слов не превышают 100.

Формат выходных данных

Выведите готовый словарь Кхамула — N неповторяющихся орочьих слов в лексикографическом порядке.

Пример

dictionary.in	dictionary.out
5 C C A B A	A B C

Задача B. Минимум и максимум

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

Пусть есть множество целых чисел. Необходимо реализовать структуру данных для их хранения, поддерживающую следующие операции: `GetMin` — извлечение минимума, `GetMax` — извлечение максимума, `Insert(N)` — добавление числа в множество.

Формат входных данных

В первой строке входного файла записано одно целое число N ($1 \leqslant N \leqslant 100\,000$) — число запросов к структуре. Затем в N строках следуют запросы по одному в строке: `GetMin`, `GetMax`, `Insert(A)` — извлечение минимума, максимума и добавление числа A ($1 \leqslant A \leqslant 2^{31} - 1$). Запросы корректны, то есть нет операций извлечения для пустого множества.

Формат выходных данных

Для каждого запроса `GetMin` или `GetMax` выведите то число, которое было извлечено.

Пример

minmax.in	minmax.out
10 Insert(100) Insert(99) Insert(1) Insert(2) GetMin GetMax Insert(1) GetMin GetMin GetMax	1 100 1 2 99

Задача C. Англо-латинский словарь

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

Однажды, разбирая старые книги на чердаке, школьник Вася нашёл англо-латинский словарь. Английский он к тому времени знал в совершенстве, и его мечтой было изучить латынь. Поэтому попавшийся словарь был как раз кстати.

К сожалению, для полноценного изучения языка недостаточно только одного словаря: кроме англо-латинского необходим латинско-английский. За неимением лучшего он решил сделать второй словарь из первого.

Как известно, словарь состоит из переводимых слов, к каждому из которых приводится несколько слов-переводов. Для каждого латинского слова, встречающегося где-либо в словаре, Вася предлагает найти все его переводы (то есть все английские слова, для которых наше латинское встречалось в его списке переводов), и считать их и только их переводами этого латинского слова.

Помогите Васе выполнить работу по созданию латинско-английского словаря из англо-латинского.

Формат входных данных

Во входном файле содержатся несколько описаний английских слов. Каждое описание содержится в отдельной строке, в которой записано сначала английское слово, затем отделённый пробелами дефис (символ номер 45), затем разделённые запятыми с пробелами переводы этого английского слова на латинский. Переводы отсортированы в лексикографическом порядке. Порядок следования английских слов в словаре также лексикографический.

Все слова состоят только из маленьких латинских букв, длина каждого слова не превосходит 15 символов. Общее количество слов на входе не превышает 100 000.

Формат выходных данных

Программа должна вывести количество латинских слов в словаре k. В следующих k строках программа должна вывести латинско-английский словарь, соответствующий входному словарю, в точности соблюдая формат входных данных. В частности, первым должен идти перевод лексикографически минимального латинского слова, далее — второго в этом порядке и т. д. Внутри перевода английские слова должны быть также отсортированы лексикографически.

Пример

dictionary.in	dictionary.out
apple - malum, pomum, popula fruit - baca, bacca, popum punishment - malum, multa	7 baca - fruit bacca - fruit malum - apple, punishment multa - punishment pomum - apple popula - apple popum - fruit

Задача D. Частотный анализ

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Вам дан текст. Мы не спрашиваем вас, что хотел сказать автор; в течение отведенного вам времени выпишите все слова из текста в порядке убывания их частоты.

Формат входных данных

Во входном файле содержится исходный текст. Текст состоит не более чем из 100 000 слов, разделённых пробелами и переводами строк. Все слова состоят из строчных латинских букв. Соседние слова разделены ровно одним пробельным символом. Длина любого слова не превышает 20 символов.

Формат выходных данных

Выведите все слова, встречающиеся в тексте, по одному на каждую строку. Слова должны быть отсортированы по убыванию их количества в тексте, а при равенстве — по алфавиту.

Примеры

стандартный ввод	стандартный вывод
hi hi what is your name my name is bond james bond my name is damme van damme claude van damme jean claude van damme	damme is name van bond claude hi my james jean what your
oh you touch my tralala mmm my ding ding dong	ding my dong mmm oh touch tralala you

Задача E. Монополия

Имя входного файла:	`monopoly.in`
Имя выходного файла:	`monopoly.out`
Ограничение по времени:	15 секунд
Ограничение по памяти:	256 мегабайт

В новом варианте игры "Монополия" появилась возможность объединять несколько предприятий в одно для увеличения приносимого ими дохода. При это 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 обойдется 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 входных данных

В единственной строке входного файла записано N натуральных чисел ($2 \leqslant N \leqslant 200\,000$), каждое из которых не превосходит 10000 — стоимости Колиных предприятий.

Формат выходных данных

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

Примеры

monopoly.in	monopoly.out
10 11 12 13	4.6000000000
1 1	0.1000000000

Задача F. Электрички

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

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 нем довольно долго, в частности, электричка, которая прибывает раньше другой, отправляться обратно может значительно позднее.

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

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

Формат входных данных

В первой строке вводится число K — количество тупиков ($1 \leqslant K \leqslant 20000$). 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 100000, для каждой фигурирующей электрички присутствуют оба события.

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

Формат выходных данных

Выведите по одному числу на каждую электричку — номер тупика, куда её поставят по прибытии. Если тупиков не достаточно для того, чтобы организовать движение электричек согласно расписанию, выведите число 0 и 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 G. Снеговики

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

Для того, чтобы слепить снеговика, необходимо три снежных кома разного размера. В вашем распоряжении есть n снежных комов, которые представляют собой шары с радиусами r_1, r_2, ..., r_n. Снеговика можно слепить из любых трех комов, радиусы которых попарно различны. Например, из комов с радиусами 1, 2 и 3 можно слепить снеговика, а из комов с радиусами 2, 2, 3 или 2, 2, 2 — нельзя. Определите, какое наибольшее количество снеговиков можно слепить из данных комов.

Формат входных данных

В первой строке входных данных задано целое число n ($1 \leqslant n \leqslant 10^5$) — количество комов. В следующей строке заданы n целых чисел — радиусы комов r_1, r_2, ..., r_n ($1 \leqslant r_i \leqslant 10^9$). Радиусы комов могут совпадать.

Формат выходных данных

В первой строке выведите одно целое число k — наибольшее количество снеговиков. Следующие k строк должны содержать описания снеговиков. Описание каждого снеговика должно состоять из трех чисел, разделенных пробелами — радиуса большого кома, радиуса среднего кома и радиуса маленького кома. Снеговиков разрешается выводить в любом порядке. Если решений несколько, выведите любое.

Примеры

snowmen.in	snowmen.out
7 1 2 3 4 5 6 7	2 7 6 5 4 3 2
3 2 2 3	0

Задача H. ACM Марафон

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

64 мегабайта

Школьник Вася Иванов так сильно боялся идти на командную олимпиаду, что ему приснился кошмар: в ЛКШ вместо обычной олимпиады устраивали обязательный ACM-марафон. Это почти обычная командная олимпиада, отличается она только продолжительностью. Марафон длится ровно 24 часа, то есть если он начался в 00:00:00 то в 23:59:59 команда еще может сдать решение, а в 00:00:00 следующего дня — уже нет.

Как и в обычном турнире ACM, побеждает команда, решившая наибольшее число задач, а при равном количестве решенных задач лучше результат у той команды, у которой меньше штрафное время. Изначально штрафное время каждой команды равно нулю. За каждую правильно сданную задачу к штрафному времени команды прибавляют время в минутах, округленное вниз, прошедшее с начала соревнования до момента сдачи задачи. Кроме того, если зачтённой попытке предшествовало несколько неудачных попыток сдать ту же задачу, то за каждую из них к штрафному времени прибавляют двадцать минут. За неудачные попытки сдать задачу, которую команде в итоге так и не удалось решить, штрафного времени не начисляется. Так же посылки с результатом “Compilation error” и “Code style violation” не считаются неудачными, то есть за них не начисляются штрафные минуты.

Вам требуется написать программу, которая подсчитает результаты марафона.

Формат входных данных

В первой строке входного файла находится время начала олимпиады в формате $hh : mm : ss$, где двухразрядное целое число hh ($0 \leqslant hh \leqslant 23$) означает час, а двухразрядные целые числа mm и ss ($0 \leqslant mm, ss \leqslant 59$) — минуты и секунды соответственно.

Во второй строке находится единственное целое число n ($1 \leqslant n \leqslant 1000$) — количество посылок за олимпиаду.

Далее следуют n строк с описаниями посылок. В начале каждой из них в двойных кавычках записано название команды, сделавшей посылку. Название может состоять из строчных и заглавных латинских букв, пробелов и цифр от 1 до 9. Длина названия — не меньше одного символа и не больше 255. После названия команды написано время посылки в том же формате, что и время начала контеста.

Далее через пробел идет заглавная латинская буква — номер задачи. Последние два символа в строке — результат посылки. Результат посылки может быть одни из следующих:
OK — OK
WA — Wrong answer
PE — Presentation error
TL — Time limit
ML — Memory limit
CE — Compillation error
CS — Code style violation

Формат выходных данных

Выходной файл должен содержать итоговую таблицу результатов — по строке на каждую команду. Строки должны идти в порядке уменьшения результата, если у нескольких команд результаты равны, то порядок команд определяется названием — раньше идет та, название которой лексикографически меньше.

Каждая строка должна начинаться с места команды в итоговом зачете. Место команды — это $k + 1$, где k — число команд, имеющих строго лучший результат. Далее через пробел идет название

команды в двойных кавычках, а за ним через пробел два числа — количество решенных задач и штрафное время.

Примеры

contest.in	contest.out
00:00:00 5 "Super team" 00:00:23 A WA "Mega team" 00:10:21 A WA "Super team" 00:20:23 A OK "Mega team" 00:30:23 A OK "Mega team" 00:40:23 B OK	1 "Mega team" 2 90 2 "Super team" 1 40
01:00:00 3 "Team1" 01:10:00 A WA "Team1" 01:20:00 A OK "Team2" 01:40:00 B OK	1 "Team1" 1 40 1 "Team2" 1 40

Задача I. Четвертый этаж

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

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 входных данных

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 выходных данных

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

Пример

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