Задача A. Цифровой корень

Имя входного файла:	`dig-root.in`
Имя выходного файла:	`dig-root.out`
Ограничение по времени:	0.5 секунда
Ограничение по памяти:	64 мегабайта

Цифровым корнем числа n называется следующее число: берется сумма цифр числа n, затем сумма цифр у получившегося числа и так далее пока не получится однозначное число.

Ваша задача — отсортировать данный массив по возрастанию цифровых корней его элементов. Если цифровые корни двух чисел равны, то раньше должно идти меньшее число.

Формат входных данных

В первой строке файла через пробел введены элементы массива. Длина массива не превосходит 200, каждое число положительно и не превосходит 10^9.

Формат выходных данных

Массив, отсортированный в порядке возрастания цифрового корня.

Примеры

dig-root.in	dig-root.out
15 14 13 12 11 10 9 8 7	10 11 12 13 14 15 7 8 9
80 61 51 41 22 1	1 22 41 51 61 80

Замечание

Требуется в решении написать вспомогательную функцию `digital_root(number)`, вычисляющую и возвращающую цифровой корень числа. Эту функцию необходимо использовать в сортировке по ключу в качестве ключа.

Задача B. Постфиксная запись

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	1 second
Ограничение по памяти:	64 megabytes

В постфиксной записи (или обратной польской записи) операция записывается после двух операндов. Например, сумма двух чисел A и B записывается как $A\ B\ +$. Запись $B\ C\ +\ D\ *$ обозначает привычное нам $(B + C) * D$, а запись $A\ B\ C\ +\ D\ *\ +$ означает $A + (B + C) * D$. Достоинство постфиксной записи в том, что она не требует скобок и дополнительных соглашений о приоритете операторов для своего чтения.

Дано выражение в обратной польской записи. Определите его значение.

Формат входных данных

В единственной строке записано выражение в постфиксной записи, содержащее однозначные числа и операции $+$, $-$, $*$. Строка содержит не более 100 чисел и операций.

Формат выходных данных

Необходимо вывести значение записанного выражения. Гарантируется, что результат выражения, а также результаты всех промежуточных вычислений по модулю меньше 2^{31}.

Примеры

stdin	stdout
8 9 + 1 7 - *	-102

Задача C. Права доступа

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

В файловую систему одного суперкомпьютера проник вирус, который сломал контроль за правами доступа к файлам. Для каждого файла N_i известно, с какими действиями можно к нему обращаться:

- запись (`W`),
- чтение (`R`),
- запуск (`X`).

Вам требуется восстановить контроль над правами доступа к файлам (ваша программа для каждого запроса должна будет возвращать «`OK`» если над файлом выполняется допустимая операция, или же «`Access denied`», если операция недопустима).

Формат входных данных

В первой строке входного файла содержится число N ($1 \leqslant N \leqslant 10\,000$) — количество файлов, содержащихся в данной файловой системе.

В следующих N строчках содержатся имена файлов, состоящие из маленьких латинских букв, цифр, точек и символов подчёркивания, и допустимых с ними операций, разделенные пробелами. Длина имени файла не превышает 15 символов.

Далее указано число M ($1 \leqslant M \leqslant 50\,000$) — количество запросов к файлам.

В последних M строках указан запрос вида «Операция Файл». К одному и тому же файлу может быть применено любое количество запросов.

Формат выходных данных

Для каждого из M запросов нужно вывести в отдельной строке «`Access denied`» или «`OK`».

Примеры

access.in	access.out
4 helloworld.exe R X pinglog W R nya R goodluck X W R 5 read nya write helloworld.exe execute nya read pinglog write pinglog	OK Access denied Access denied OK OK

Задача D. 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 E. Монополия

Имя входного файла:	`monopoly.in`
Имя выходного файла:	`monopoly.out`
Ограничение по времени:	2 секунды
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. Марсианская парикмахерская

Имя входного файла:	`saloon.in`
Имя выходного файла:	`saloon.out`
Ограничение по времени:	1 second
Ограничение по памяти:	64 megabytes

Пока я не поспал, «сегодня» не наступило

мистер Грин

В парикмахерской работает один мастер. Он тратит на одного клиента ровно 20 минут, а затем сразу переходит к следующему, если в очереди кто-то есть, либо ожидает, когда придет следующий клиент.

Даны времена прихода клиентов в парикмахерскую (в том порядке, в котором они приходили).

Также у каждого клиента есть характеристика, называемая *степенью нетерпения*. Она показывает, сколько человек может максимально находиться в очереди перед клиентом, чтобы он дождался своей очереди и не ушел раньше. Если в момент прихода клиента в очереди находится больше людей, чем степень его нетерпения, то он решает не ждать своей очереди и уходит. Клиент, который обслуживается в данный момент, также считается находящимся в очереди.

Требуется для каждого клиента указать время его выхода из парикмахерской.

Формат входных данных

В первой строке вводится натуральное число N, не превышающее 10^5 — количество клиентов.

В следующих N строках вводятся времена прихода клиентов — по два числа, обозначающие часы и минуты (часы — от 0 до 16 000 000, минуты — от 0 до 59) и степень его нетерпения (неотрицательное целое число не большее 10^5) — максимальное количество человек, которое он готов ждать впереди себя в очереди. Времена указаны в порядке возрастания (все времена различны).

Если для каких-то клиентов время окончания обслуживания одного клиента и время прихода другого совпадают, то можно считать, что в начале заканчивается обслуживание первого клиента, а потом приходит второй клиент.

Формат выходных данных

В выходной файл выведите N пар чисел: времена выхода из парикмахерской 1-го, 2-го, ..., N-го клиента (часы и минуты). Если на момент прихода клиента человек в очереди больше, чем степень его нетерпения, то нужно считать, что время его ухода равно времени прихода.

Примеры

saloon.in	saloon.out
3 10 0 0 10 1 1 10 2 1	10 20 10 40 10 2
5 1 0 100 2 0 0 2 1 0 2 2 3 2 3 0	1 20 2 20 2 1 2 40 2 3

Задача G. Гемоглобин

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	3 секунды
Ограничение по памяти:	64 мегабайта

Каждый день к Грегори Хаусу приходит много больных, и у каждого измеряется уровень гемоглобина в крови. Данные по всем пациентам заносятся в базу данных.

Но волчанка попадается один раз на миллион, а работать с остальными неинтересно. Чтобы Хаус не выгонял больных, Кадди иногда запрашивает статистику по k последним больным: ей хочется знать сумму их уровня гемоглобина.

Также Хаус — мизантроп: он смотрит уровень гемоглобина больного, который поступил к нему позже всех, и, видя, что это точно не волчанка, выписывает его из больницы и удаляет информацию о нем из базы.

Автоматизацию процесса Хаус поручил Чейзу. Но Чейз почему-то не справился с этой задачей и попросил вас ему помочь.

Формат входных данных

Первой строкой входного файла задано число n ($1 \leqslant n \leqslant 100\,000$) — число обращений к базе данных. Запросы к базе выглядят следующим образом: `«+x»` ($1 \leqslant x \leqslant 10^9$) — добавить пациента с уровнем гемоглобина x в базу, `«-»` — удалить последнего пациента из базы, `«?k»` ($1 \leqslant k \leqslant 100\,000$) — вывести суммарный гемоглобин последних k пациентов. Гарантируется, что k не превосходит число элементов в базе. Также гарантируется, что запросов на удаление к пустой базе не поступает. Перед началом работы база данных пуста.

Формат выходных данных

Для каждого запроса `«-»` вывести уровень гемоглобина в крови пациента, а для каждого запроса `«?k»` — суммарный гемоглобин у последних k поступивших пациентов. Ответы выводите в порядке поступления запросов.

Примеры

стандартный ввод	стандартный вывод
7 +1 +2 +3 ?2 - - ?1	5 3 2 1

Задача H. Число

Имя входного файла:	`number.in`
Имя выходного файла:	`number.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Вася написал на длинной полоске бумаги большое число и решил похвастаться своему старшему брату Пете этим достижением. Но только он вышел из комнаты, чтобы позвать брата, как его сестра Катя вбежала в комнату и разрезала полоску бумаги на несколько частей. В результате на каждой части оказалось одна или несколько идущих подряд цифр.

Теперь Вася не может вспомнить, какое именно число он написал. Только помнит, что оно было очень большое. Чтобы утешить младшего брата, Петя решил выяснить, какое максимальное число могло быть написано на полоске бумаги перед разрезанием.

Помогите ему!

Формат входных данных

Входной файл содержит одну или более строк, каждая из которых содержит последовательность цифр. Количество строк во входном файле не превышает 100, каждая строка содержит от 1 до 100 цифр. Гарантируется, что хотя бы в одной строке первая цифра отлична от нуля.

Формат выходных данных

Выведите в выходной файл одну строку — максимальное число, которое могло быть написано на полоске перед разрезанием.

Примеры

number.in	number.out
2 20 004 66	66220004
3	3

Задача I. 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 J. ACM Марафон

Имя входного файла:	`contest.in`
Имя выходного файла:	`contest.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	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 K. Звезда в Отеле

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Как известно, в Отеле скучно не бывает: то фуршет, то номерки какие-то загадочные. А вот сейчас, для гостей опять подготовили нечто невероятное: приезжает Звезда.

Звезды — народ причудливый, вот например, Звезда, приехавшая в Отель, боится больших скоплений людей, поэтому поклонников на автограф-сессии будет принимать в специальном кабинете и только по одному, все же остальные гости будут ожидать в зале. И все бы хорошо, человечество давно придумало очереди, только вот во время сессии проходит фуршет, поэтому все разбредутся по залу и никакой очереди в ее привычном понимании устроить не получится и САО (служба администрирования очередей) решила обратиться к вам.

Вас просят написать систему учета людей, которая должна уметь выводить номер текущего первого в очереди гостя, добавлять людей в очередь и удалять из нее. Но мало того, что неразбериха с фуршетом, САО ещё и по неизвестной нам причине поощряет использование знакомств и связей в личных корыстных целях (в народе «блат»): если в зал приходит человек x, то он хочет найти своего друга y и, если y находится в зале, то x в очереди становится прямо за ним, иначе x помещается системой в конец очереди. Также может произойти такое, что гостю надоело ждать и тогда он просто уходит из зала ожидания и исключается из очереди.

Конечно, обработка событий по видео-камерам — увлекательный процесс, но САО решили, что проще будет дать уже готовую последовательность данных. Итак, вашей системе нужно обрабатывать следующие запросы:

1. `in x y` — в зал фуршета приходит гость с номером x, который дружит с гостем номер y. Если y уже находится в очереди, то x встает за ним, иначе в конец. Стоит отметить, что гости очень восхищаются Звездой и могут приходить и не по одному разу.

2. `out x` — гостю под номером x надоело ждать и он уходит (**гарантируется, что в данный момент гость с таким номером находится в зале**).

3. `check` — Звезда готова дать очередной автограф и САО хочет узнать, кто первый в очереди (после этого счастливец получит автограф и уйдет по своим делам). Если очередь пуста, выведите -1.

Формат входных данных

В первой строке вводится единственное натуральное число q ($1 \leqslant q \leqslant 200\,000$) — количество запросов. Далее в q строках вводятся вышеописанные запросы. Все числа, содержащиеся в запросах натуральные и не превосходят 10^8.

Формат выходных данных

На каждый запрос ***check*** в отдельной строке выведете номер первого человека в очереди.

Примеры

стандартный ввод	стандартный вывод
10 in 1 1 in 2 1 in 3 1 in 4 2 check out 4 check in 5 6 in 6 5 check	1 3 2
10 check in 10 5 in 9 3 in 6 7 out 6 in 5 3 in 3 4 check in 7 2 in 2 8	-1 10