Задача A. Выбор заявок

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

Вы прекрасно знаете, что в ЛКШ.Зима 2019 лекции читают лучшие преподаватели мира. К сожалению, лекционных аудиторий у нас не так уж и много, поэтому каждый преподаватель составил список лекций, которые он хочет прочитать ЛКШатам. Чтобы ЛКШата, утром идя на завтрак, увидели расписание лекций, необходимо его составить прямо сейчас. И без вас нам здесь не справиться.

У нас есть список заявок от преподавателей на лекции для одной из аудиторий. Каждая заявка представлена в виде временного интервала $[s_i, f_i)$ — время начала и конца лекции. Лекция считается открытым интервалом, то есть какая-то лекция может начаться в момент окончания другой, без перерыва. Необходимо выбрать из этих заявок такое подмножество, чтобы суммарно выполнить максимальное количество заявок. Учтите, что одновременно в лекционной аудитории, конечно же, может читаться лишь одна лекция.

Формат входных данных

В первой строке вводится натуральное число N, не более 1000 — общее количество заявок на лекции. Затем вводится N строк с описаниями заявок — по два числа в каждом s_i и f_i. Гарантируется, что $s_i < f_i$. Время начала и окончания лекции — натуральные числа, не превышают 1440 (в минутах с начала суток).

Формат выходных данных

Выведите одно число — максимальное количество заявок, которые можно выполнить.

Примеры

request.in	request.out
1 5 10	1
3 1 5 2 3 3 4	2

Задача B. Планирование заданий

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

Имеется некоторое множество заданий и один исполнитель. На выполнение одного задания уходит единица времени. Задания можно выполнять начиная с момента времени 0. У каждого задания есть две характеристики: d_i и w_i. Если задание не было выполнено к моменту времени d_i, взимается штраф в размере w_i. Требуется минимизировать суммарный штраф.

Формат входных данных

Первая строка входного файла содержит натуральное число n — количество заданий ($1 \leqslant n \leqslant 1000$). Следующие n строк содержат по два натуральных числа, разделенных пробелом — d_i и w_i ($0 \leqslant d_i, w_i \leqslant 1000$).

Формат выходных данных

Выведите одно число — минимальный суммарный штраф.

Примеры

schedule.in	schedule.out
2 1 1 1 2	1
1 0 5	5

Задача C. Последовательность

Имя входного файла:	`sequence.in`
Имя выходного файла:	`sequence.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Дана последовательность натуральных чисел $a_1, a_2, \ldots, a_n$, и известно, что $a_i \leqslant i$ для любого $1 \leqslant i \leqslant n$. Требуется определить, можно ли разбить элементы последовательности на две части таким образом, что сумма элементов в каждой из частей будет равна половине суммы всех элементов последовательности.

Формат входных данных

В первой строке входного файла находится одно целое число n ($1 \leqslant n \leqslant 40\,000$). Во второй строке находится n целых чисел $a_1, a_2, \ldots, a_n$ ($1 \leqslant a_i \leqslant i$).

Формат выходных данных

В первую строку выходного файла выведите количество элементов посделовательности в любой из получившихся двух частей, а во вторую строку через пробел номера этих элементов. Если построить такое разбиение невозможно, выведите -1.

Примеры

sequence.in	sequence.out
3 1 2 3	1 3

Задача D. Банкомат

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

— Купюры в нашем банкомате самые что ни на есть настоящие! И их хватит на всех! — орал Фагот, приглашая желающих испробовать чудесную машину. Присутствующие щелкали клавишами, автомат гудел и отсчитывал новенькие банкноты прямо в руки. *Уже через полчаса им было суждено бесследно растаять, но откуда могли об этом знать зрители?*

— Но позвольте! Пусть в банкомате есть c_1 купюр по пятьдесят рублей, c_2 купюр по сотне, c_3 полутысячных, c_4 по тысяче и аж целых c_5 пятитысячных, — начал рассуждать конферансье, — если первый зритель возьмет a_1 рублей, второй — a_2, и так далее, то найдется такой x, что x-му зрителю банкнот точно не хватит!

— Вот вы и назовете x. Причем мгновенно! — Кот зловеще сверкнул глазами, — А если не назовете, то и голова вам не пригодится!..

Покрываясь холодным потом, конферансье судорожно думал о том, что банкомат магическим образом заранее знает о том, какие суммы будут запрашивать зрители, и, кроме того, после выдачи очередной суммы банкноты в машине не восстанавливаются.

Формат входных данных

В первой строке записаны пять целых чисел c_1, c_2, c_3, c_4, c_5 — количества банкнот по 50, 100, 500, 1000, 5000 рублей, соответственно.

Во второй строке записано целое число n — количество зрителей. В следующих n строках записано по одному целому числу a_i — необходимое i-му зрителю количество рублей.

$1 \leqslant n \leqslant 100\,000$, $0 \leqslant a_i \leqslant 10^9$, $0 \leqslant c_j \leqslant 10^9$.

Формат выходных данных

Выведите единственное число x — номер первого человека, которому ни при каком способе выдачи денег не получится выдать точную сумму, которую он запросил. Если всех зрителей удастся удовлетворить, выведите число $n + 1$.

Примеры

cash-machine.in	cash-machine.out
1 2 3 4 5 5 25000 4000 1500 200 50	6
3 4 2 0 4 3 100 1000 500	3

Задача E. Такси

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

После затянувшегося совещания директор фирмы решил заказать такси, чтобы развезти сотрудников по домам. Он заказал N машин — ровно столько, сколько у него сотрудников. Но когда они подъехали, оказалось, что у каждого водителя такси свой тариф за 1 километр.

Каждый сотрудник сказал директору, сколько километров ему нужно проехать до дома. Разные сотрудники должны сесть в разные такси. Теперь директор хочет определить, какой из сотрудников на каком такси должен поехать домой, чтобы суммарные затраты на такси (а их несет фирма) были минимальны.

Формат входных данных

Сначала во входном файле записано натуральное число N ($1 \leqslant N \leqslant 1000$) — количество сотрудников компании (совпадающее с количеством вызванных машин такси). Далее записано N чисел, задающих расстояния в километрах от работы до домов сотрудников компании (первое число — для первого сотрудника, второе — для второго и т. д.). Все расстояния — положительные целые числа, не превышающие 1000. Далее записано еще N чисел — тарифы за проезд одного километра в такси (первое число — в первой машине такси, второе — во второй и т.д.). Тарифы выражаются положительными целыми числами, не превышающими 10000.

Формат выходных данных

В выходной файл выведите N чисел — оптимальное распределение сотрудников по такси. Первым выведите номер такси, в которое должен сесть первый сотрудник, вторым — номер такси, в которое должен сесть второй и т.д. Если есть несколько вариантов рассадки сотрудников, при которых затраты минимальны, выведите любой из них.

Примеры

taxi.in	taxi.out
3 10 20 30 50 20 30	1 3 2
5 10 20 1 30 30 3 3 3 2 3	1 2 3 5 4

Задача F. Минимальное покрытие

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

На прямой задано несколько отрезков с целочисленными координатами концов $[L_i, R_i]$. Выберите такое подмножество этих отрезков, что оно целиком покрывает отрезок $[0, M]$ и при этом состоит из минимально возможного количества элементов.

Формат входных данных

В первой строке находится целое число M ($1 \leqslant M \leqslant 5000$).

В следующих строках записана информация об отрезках — по одному на строку. Отрезок задаётся парой целых чисел L_i и R_i ($|L_i|, |R_i| \leqslant 50\,000$). Список завершается парой нулей. Количество отрезков не превышает 100 000.

Формат выходных данных

В первой строке выходного файла выведите минимальное количество отрезков, необходимое для покрытия отрезка $[0, M]$. Далее выведите координаты отрезков из покрывающего подмножества, упорядоченный по возрастанию координат левых концов отрезков.

Если покрыть отрезок $[0, M]$ нельзя, выведите единственную фразу «`No solution`».

Примеры

covering.in	covering.out
1 -1 0 -5 -3 2 5 0 0	No solution
1 -1 0 0 1 0 0	1 0 1

Задача G. Число

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
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

stdin	stdout
2 20 004 66	66220004
3	3

Задача H. Максимальная сумма «И»

Имя входного файла:	`andsum.in`
Имя выходного файла:	`andsum.out`
Ограничение по времени:	4 секунды
Ограничение по памяти:	256 мегабайт

Омар Вольдемар обожает шестнадцатеричные числа, а также очень любит операцию побитового «И», поэтому он наверняка обрадовался бы этой задаче.

Даны n целых неотрицательных чисел, заданных четырьмя цифрами в шестнадцатеричной системе счисления. Требуется разбить эти числа на два **непустых** множества так, чтобы сумма результата операции побитового «И» всех чисел первого множества и результата операции побитового «И» всех чисел второго множества была максимальна.

Формат входных данных

В первой строке входных данных записано число n ($2 \leqslant n \leqslant 1000$) — количество чисел, которые требуется разбить на два непустых множества. В последующих n строках представлены числа a_i в виде четырёх цифр в шестнадцатеричной системе счисления (возможно, с ведущими нулями).

Формат выходных данных

Выведите максимальную сумму результата операции побитового «И» для всех чисел первого множества и результата операции побитового «И» для всех чисел второго множества, которую можно получить при разбиении исходного множества на два. Ответ требуется вывести в шестнадцатеричной системе счисления без ведущих нулей.

Примеры

andsum.in	andsum.out
3 0023 031A 0121	33B
2 8000 8000	10000
3 0000 0000 0000	0

Замечание

В первом примере даны числа, которые в десятичной системе счисления записываются как 35, 794 и 289. Если отнести 794 к одной группе, а 35 и 289 к другой группе, то получатся числа 794 и 33, сумма которых 827, а если перевести в шестнадцатеричную систему — $33B$.