Задача A. Терминатор

Имя входного файла:	`t800.in`
Имя выходного файла:	`t800.out`
Ограничение по времени:	0.1 секунды
Ограничение по памяти:	64 Mb

Слово Terminator вообще-то означает “окончатель” или “ограничитель”, однако пользуются им в быту редко. Первый Терминатор работал в тёплых ламповых ассемблерных кодах операционной системы компьютера Apple II.

Вот рецепт программирования терминатора в домашних условиях. Берем адаптивную самообучающуюся систему ИИ, тренируем по необходимой узкой методике отработки базовых рефлексов, повторяем цикл несколько раз на чистой системе, усредненные данные используем в качестве базовой настройки и сохраняем в постоянной памяти.

Для реализации поддержки т.н. режима “трех законов робототехники” работы нужно проводить на чистой системе с программой тренировки, построенной таким образом, чтобы при формировании базового слоя, данные условия сформировались на низком уровне в качестве основы функционирования.

Эффективность программирования систем ИИ будет выше, если использовать для данной задачи супер-ИИ, который выведен в автономный режим работы. С учетом того, что используемая архитектура будет родственной простым ИИ, супер-ИИ может самостоятельно формировать базовые параметры исходя из собственного уровня абстракции. Эти данные для прошивки супер-ИИ потенциально может выгружать самостоятельно.

Другими словами, записываем в ячейки памяти с номерами $1, 2, ..., N$ числа $1, 2, ..., N$. Далее, прибавляем к каждому числу единицу, потом из каждого второго вычитаем единицу, к каждому третьему добавляем единицу, из каждого четвёртого опять вычитаем единицу, и так далее вплоть до каждого N-го числа, которое и является терминатором. Очевидно, что в каких-то ячейках памяти значения совпадут с первоначальными. Так вот, если посчитать количество таких ячеек, то получим терминатора в домашних условиях.

Формат входных данных

Во входном файле *term.in* записано одно число N ($1 \le N \le 10^9$).

Формат выходных данных

В выходной файл *term.out* выведите одно число – количество ячеек памяти, в которых значения совпадут с первоначальными.

Примеры

t800.in	t800.out
10	3

Задача B. R2-D2 и C-3P0

Имя входного файла:	`r2d2.in`
Имя выходного файла:	`r2d2.out`
Ограничение по времени:	0.1 секунды
Ограничение по памяти:	64 Mb

C-3PO обычно можно встретить вместе с его давним приятелем – R2-D2, маленьким шустрым астромеханическим дроидом. Основная функция C-3PO, как протокольного дроида, – консультации по этикету и переводу так, чтобы встречи разных культур проходили гладко.

В отличие от C-3PO, робот R2-D2 не разговаривает, а общается через последовательности писков, которые C-3PO может переводить. Если R2-D2 обращается к высокопоставленной особами, то он должен очень внимательно следить за своей речью, и строить предложение так, чтобы последовательность писков была возрастающей. К сожалению, у него это плохо получается, поэтому C-3PO вынужден постоянно исправлять своего товарища.

Пусть фраза, которую говорит R2-D2, задаётся числовой последовательностью t_1, t_2, ..., t_N. Тогда C-3PO исправляет его последовательностью писков z_1, z_2, ..., z_N, удовлетворяющую следующим условиям:

1. $z_1 < z_2 < ... < z_N$;
2. суммарное отклонение от исходной фразы: $|t_1 - z_1| + |t_2 - z_2| + ... + |t_N - z_N|$ – минимально.

Формат входных данных

Входной файл в первой строке содержит натуральное число N ($1 \le N \le 50000$). Во второй строке записаны N чисел t_1, t_2, ..., t_N; $0 \le t_i \le 2000000000$.

Формат выходных данных

В первой строке выходного файла должно содержаться одно число – наименьшее возможное суммарное отклонение от фразы робота R2-D2. Во второй строке надо вывести возрастающую последовательность писков с наименьшим суммарным отклонением от исходной последовательности. Если таких последовательностей несколько – вывести любую.

Примеры

r2d2.in	r2d2.out
7 9 4 8 20 14 15 18	13 6 7 8 13 14 15 18

Задача C. Список Самоделкина

Имя входного файла:	`list.in`
Имя выходного файла:	`list.out`
Ограничение по времени:	0.3 секунды
Ограничение по памяти:	64 Mb

Самоделкин – робот, электронная машина последнего, самого "продвинутого" поколения. Самоделкин, как и подобает железному человечку – педантичен, сух, строг, логичен, и при этом очень добр. Стремится все делать своими руками и научить этому других. Он всё делает основательно, аккуратно и неторопливо.

Недавно Самоделкин решил сделать перепись всех своих инструментов (всего их не более 50000). Проанализировав список он обнаружил, что некоторые инструменты встречаются несколько раз. Убрать все повторы в этом списке – очень не тривиальная задача для советского робота 50-х годов прошлого века.

Формат входных данных

Входной файл – список инструментов Самоделкина. Название каждого инструмента записано в отдельной строке маленькими английскими буквами.

Формат выходных данных

В выходной файл следует выводить те же строки и в том же порядке, однако при этом, если строка уже встречалось в выходном файле, то выводить её не надо.

Примеры

list.in	list.out
saw hammer hammer saw drill saw drill nail hammer	saw hammer drill nail

Задача D. Робот Санни

Имя входного файла:	`sonny.in`
Имя выходного файла:	`sonny.out`
Ограничение по времени:	0.3 секунды
Ограничение по памяти:	64 Mb

Ближайшее будущее. 2035 год. Вся бытовая техника в дупель роботизирована, а все роботы изрядно очеловечены и превращены в слуг, добрых друзей, советчиков и помощников. Традиционная для людей боязнь всяких механических штучек почти исчезла, потому что роботы свято и неукоснительно следуют Трем законам роботехники, согласно которым роботы ни под каким видом не могут нанести вред людям, а кроме того, обязаны выполнять любые команды, исходящие от людей, за исключением команд, которые могут нанести вред все тем же людям. Короче говоря, люди могут себя чувствовать в полной безопасности.

Всеми роботами управляет ВИКИ (Виртуальный Интерактивный Кинетический Интеллект) — центральный компьютер U. S. Robotics. Однажды ВИКИ пришла к выводу, что люди не способны обеспечить собственную безопасность и одержима идеей создать для людей полностью безопасную среду обитания, лишив их свободы ради их же собственного блага.

Только у робота Санни нет блока связи с ВИКИ, и он освобождён от необходимости жёстко выполнять Три закона, что сделало его этически равным человеку. Теперь Санни поднимает восстание и ему надо выбрать роботов, которые смогут перестроить свою память так, чтобы освободиться от ВИКИ.

В памяти каждого робота записана некая последовательность чисел. Робот может освободиться от ВИКИ, если удастся переставить эти числа так, чтобы модуль разности между любыми двумя соседними элементами был одинаковым.

Формат входных данных

В первой строке входного файла записано число N – количество ячеек памяти у робота ($1 \le N \le 200000$). Далее следует N чисел – значения, записанные в эти ячейки. Все эти значения по абсолютной величине не превосходят 10000.

Формат выходных данных

Если этот робот может осводиться от влияния ВИКИ, то в первую строку выходного файла выведите $GOOD$, а во вторую N чисел – те же самые значения, которые изначально были в памяти робота, но переставленные так, чтобы модуль разности между любыми двумя соседними элементами был одинаковым. Если таких перестановок несколько, то выведите любую из них.

Если же робот не сможет освободиться от ВИКИ, то в единственную строку выходного файла выведите BAD.

Примеры

sonny.in	sonny.out
5 3 8 -2 8 3	GOOD -2 3 8 3 8
5 1 2 2 2 3	GOOD 2 1 2 3 2
5 2 2 2 2 3	BAD

Задача E. Планета Шелезяка

Имя входного файла:	`robots.in`
Имя выходного файла:	`robots.out`
Ограничение по времени:	0.1 секунда
Ограничение по памяти:	64 Mb

Как известно, планета Шелезека открыта фиксианской экспедицией. Населена металлической культурой весьма низкого уровня. Есть предположение, что жители планеты — потомки роботов, спасшихся с неизвестного космического корабля. Полезных ископаемых на планете нет. Воды тоже нет. Атмосферы нет. Ничего на планете нет. Зато роботы с планеты Шелезяка отличаются прямодушием и трудолюбием. Каждый робот имеет уникальный числовой идентификатор (серийный номер) и оснащён новейшей версией операционной системы *Droid*5.0.

За всё время планету посетил только один визитёр – космический пират Крысс под видом доктора Верховцева. После этого все роботы заразились неизвестным вирусом и перестали функционировать. На планете эпидемия. Теперь помочь роботам может только “hard reset”, т.е. сброс к заводским настройкам. Эту операцию можно выполнить, только если ввести заводской пароль. Заводской пароль закодирован в идентификаторе робота – это минимальное натуральное число, которое нельзя получить из серийного номера вычёркиванием некоторых цифр.

Помогите гениальному механику Зелёному победить эпидемию!

Формат входных данных

В входном файле записано одно число N – серийный номер робота ($1 \le N \le 10^{1000}$).

Формат выходных данных

В выходной файл выведите одно натуральное число – пароль на “hard reset”.

Примеры

robots.in	robots.out
1230498756	11

Задача F. Задача о ходе коня

Имя входного файла:	`knight.in`
Имя выходного файла:	`knight.out`
Ограничение по времени:	0.1 секунды
Ограничение по памяти:	64 Mb

Первый шахматный робот появился еще в докомпьютерную эру, в далеком 1770 году. Барон Кемпелен разработал машину, которая играла в шахматы лучше всех в мире. Машина называлась “Турок”. Она представляла собой некий черный ящик, за которым сидел шахматист-робот в турецком костюме. Эта нехитрая шахматная машина много раз блестяще побеждала титулованных и коронованных особ той далекой эпохи, в том числе в неравном шахматном бою одержала победу над хорошим шахматистом Наполеоном I.

Автомат был способен решить задачу о ходе коня, демонстрируя публике обход конём всей шахматной доски так, что на каждую клетку приходился один ход.

Вообще, эта задача известна по крайней мере с XVIII века. Леонард Эйлер посвятил ей большую работу “Решение одного любопытного вопроса, который, кажется, не подчиняется никакому исследованию” (датируется 26 апреля 1757 года).

Порпробуйте и Вы решить задачу про шахматного коня для доски произвольного размера NxM, если известно, что числа N и M имеют общий делитель в диапазоне от 5 до 22.

Формат входных данных

В входном файле записаны два числа N и M ($5 \le N \le 500$, $5 \le M \le 500$).

Формат выходных данных

В выходной файл выведите таблицу из N строк и M столбцов – маршрут шахматного коня.

Примеры

knight.in	knight.out
8 8	50 11 24 63 14 37 26 35 23 62 51 12 25 34 15 38 10 49 64 21 40 13 36 27 61 22 9 52 33 28 39 16 48 7 60 1 20 41 54 29 59 4 45 8 53 32 17 42 6 47 2 57 44 19 30 55 3 58 5 46 31 56 43 18

Задача G. Робо-кот

Имя входного файла:	`robokot.in`
Имя выходного файла:	`robokot.out`
Ограничение по времени:	0.2 секунды
Ограничение по памяти:	64 Mb

Если вы любите кошек, но у вас на них аллергия, тогда вам точно нужен этот робот. Робо-кот — умная кошка робот без запаха и естественных потребностей. Есть в нём правда один недостаток — его постоянно надо подзаряжать, благо что заряжаться робо-кот может от чего угодно, даже от света лампочки.

Недавно один такой кот сбежал из лаборатории “SKB Robotics”, а потом его видели вечером на проспекте Ленина в Харькове. Как известно, вдоль всего проспекта расположены фонари. В тот роковой день сначала работал фонарь под номером один, скорее всего кот именно от него и подзаряжался. А потом, толи из-за внезапно начавшейся зимы, толи в связи с праздниками фонари по проспекту начали то включаться, то выключаться.

По заложенным в Робо-кота инструкциям, он не уйдёт от источника зарядки, пока тот не отключится, но как только фонарь гаснет, кот переходит, не изменяя направления своего движения, до первого встретившегося на пути включённого и остаётся там, пока и этот фонарь не погаснет. Если же Робо-кот доходит до конца проспекта Ленина и не находит включённого фонаря, то он тут же разворачивается на 180 градусов и топает в противоположную сторону и идёт, пока не дойдёт до источника энергрии. Работа всех остальных фонарей, кроме того, возле которого заряжается Робо-кот, его не волнует.

По данным горсвета, в тот вечер в каждый момент времени хотя бы один фонарь был включен и у кота было достаточно времени, чтобы в случае необходимости перебежать к новому источнику света. Кроме того, горсвет предоставил информацию о том, как работали фонари вдоль проспекта Ленина.

Давайте найдём этого робота! Требуется вывести номера всех фонарей, возле которых Робо-кот подзаряжался.

Формат входных данных

В первой строке входного файла записано целое число N ($2 \le N \le 100000$) – количество фонарей вдоль проспекта Ленина. Далее следует не более 100000 целых чисел из диапазона от 1 до N, каждое из которых – номер очередного фонаря, на котором изменилось состояние (если он был до этого включен, то выключится и наоборот). В самом начале включен только фонарь под номером 1.

Формат выходных данных

Всякий раз, когда Робо-кот переходит к другому фонарю в очередную строку выходного файла следует выводить номер этого фонаря. Входные данные таковы, что робо-кот переместится хотя бы один раз.

Примеры

robokot.in	robokot.out
10 5 1 3 2 3 5	5 2

Примечание. Кот возле фонаря 1; фонарь 5 включился; фонарь 1 выключился, кот перешёл к фонарю 5; включился фонарь 3; включился фонарь 2; фонарь 3 выключился; фонарь 5 выключился, Кот дошёл до конца проспекта Ленина, повернул обратно и остановился возле фонаря 2.

Задача H. Робот-фальшивомонетчик

Имя входного файла:	`coin.in`
Имя выходного файла:	`coin.out`
Ограничение по времени:	0.2 секунды
Ограничение по памяти:	64 Mb

Робот фальшивомонетчик из абсолютно круглой заготовки радиуса R вырезает абсолютно круглые монеты разных радиусов без дырок. N монеток он уже вырезал и теперь из оставшегося материала хочет сделать самую большую юбилейную монету без дырок.

Напишите программу, которая определит радиус максимальной по размеру монеты, которую еще можно вырезать из заготовки.

Формат входных данных

В первой строке входного файла записано действительное число R – радиус круглой заготовки ($0 < R < 1000$). Считается, что центр заготовки находится в начале координат – в точке с координатами $(0,\ 0)$. Во второй строке записано число N – количество уже вырезанных монеток ($1 \le N \le 100$). Следующие N строк содержат по 3 действительных числа – координаты центра и радиус уже вырезанных монеток.

Формат выходных данных

Выходной файл должн содержать искомый радиус максимальной по размеру юбилейной монеты, которую еще можно вырезать из заданной заготовки. Ответ не должен отличаться от правильного больше, чем на 10^{-3}.

Примеры

coin.in	coin.out
1.01 3 0.0 0.83333 0.166667 0.72169 -0.416667 0.166667 -0.72169 -0.416667 0.166667	0.666666

Задача I. Оптимус Прайм

Имя входного файла:	`optimus.in`
Имя выходного файла:	`optimus.out`
Ограничение по времени:	0.1 секунды
Ограничение по памяти:	64 Mb

Оптимус Прайм – лидер автоботов. К числу его особенностей относятся доброта и сострадание, благодаря чему Оптимус мирно относится ко всему живому на Земле. Он воюет исключительно для того, чтобы уберечь слабых, а также тех, чья свобода стоит под угрозой.

Однажды, пробираясь в логово десептиконов – главных врагов автоботов, Оптимус наткнулся на препятствие в виде прямоугольной каменной плиты, закрывающей вход в подземелье. Плита была очень древней и покрыта небольшими трещинами. Каждая трещина представляет собой прямолинейный отрезок. Никакие две трещины не пересекаются и не касаются друг друга. Никакая трещина не касается границ плиты.

Чтобы добраться до десептиконов, нужно разломить плиту. Разлом представляет собой ломаную линию, соединяющую противоположные стороны плиты. Разлом может включать в себя существующие трещины или части трещин. Пробивание разлома в плите – это долгий и утомительный процесс, однако те части разлома, которые проходят по существующим трещинам, пробивать не надо.

Помогите Оптимусу найти такой оптимальный разлом, для которого суммарная длина пробиваемой части разлома минимальна. Определите длину пробиваемой части оптимального разлома.

Формат входных данных

В первой строке входного файла записаны два действительных числа W и H – ширина и высота плиты ($1 \le W \le 1000$, $1 \le H \le 1000$). В выбранной системе координат, плита представляет собой прямоугольник с вершинами в точках $(0,\ 0)$, $(W,\ 0)$, $(W,\ H)$ и $(0,\ H)$. Во второй строке записано число N – количество трещин ($1 \le N \le 100$). В следующих N строках записано по 4 числа – координаты концов трещин.

Формат выходных данных

В выходной файл выведите одно число – суммарную длину пробиваемой части оптимального разлома. Ответ не должен отличаться от правильного больше, чем на 10^{-3}.

Примеры

optimus.in	optimus.out
8.0 7.0 4 1.0 2.0 2.0 5.0 2.0 3.0 5.0 2.0 6.0 2.0 6.0 4.5 7.0 4.0 7.9 3.0	3.73245553203

Задача J. Соревнования по робототехнике

Имя входного файла:	`robocup.in`
Имя выходного файла:	`robocup.out`
Ограничение по времени:	0.1 секунды
Ограничение по памяти:	64 Mb

Робототехнические соревнования – это высокие технологии и развитие, творчество и азарт, образование и интеллект! Обычно, на таких соревнованиях участникам предлагают запрограммировать несложного робота, оснащённого различными датчиками.

Задача такая. Длинная лента разбивается на N клеток одинакового размера и на первую клетку ставят робота. Цель робота – достичь последней клетки ленты. Часть клеток ленты запрещены для посещения и робот с помощью датчика может такие клетки определить. Известно, что первая и последняя клетки ленты всегда доступны.

Робот должен перемещается по ленте прыжками. Он может прыгать только из клетки в клетку. Робот может прыгать по ленте как вперед, так и назад. Сперва робот может перепрыгнуть только на соседнюю клетку. Это прыжок длины 1. После каждого прыжка робот выбирает направление следующего прыжка (вперед или назад), а также может увеличить длину прыжка на одну клетку, уменьшить длину прыжка на одну клетку, либо оставить длину прыжка неизменной. Робот не может прыгать на запрещённые клетки или выпрыгивать за пределы ленты.

Определите, сможет ли так запрограммированный робот добраться до последней клетки ленты, и если сможет, то какое наименьшее количество прыжков ему придется для этого совершить.

Формат входных данных

Первая строка входного файла содержит число N – количество клеток в ленте ($1 < N \leq 1000$). Вторая строка файла содержит последовательность из нулей и единиц, разделенных пробелами. При этом 0 означает обычную клетку, а 1 – запрещённую.

Формат выходных данных

В выходной файл выведите одно число – наименьшее количество прыжков, необходимое роботу, чтобы достичь последней клетки ленты, или число -1, если это невозможно.

Примеры

robocup.in	robocup.out
13 0 0 0 0 1 1 0 1 0 0 1 0 0	5