Задача A. Арифметическая прогрессия

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

Однажды Петя узнал очень важную последовательность из n чисел. Тщательно проанализировав ее, он обнаружил, что она является арифметической прогрессией. Чтобы не забыть он записал ее элементы на n карточках.

Но затем случилась неприятность. Не зная всю важность этой последовательности, его брат Вовочка взял еще n карточек и написал на них произвольные числа, а потом перемешал все $2n$ карточек.

Теперь Петя хочет восстановить исходную последовательность по этим карточкам. К сожалению возможно, что это можно сделать несколькими способами, но Петю устроят любые n чисел, образующие арифметическую прогрессию.

Петя не может сделать это вручную, поэтому обратился к вам за помощью.

Напомним что последовательность $a_1, a_2, \ldots, a_n$ называется арифметической прогрессией, если $a_i = a_{i-1} + d$ для всех i от 2 до n и некоторого d. Число d называется *разностью* арифметической прогрессии.

Формат входного файла

В первой строке входного файла находится целое число n ($1 \leqslant n \leqslant 100\,000$). В следующей строке находится $2n$ целых чисел по модулю не превосходящих 10^9 — числа, написанные на карточках, перечисленные в произвольном порядке. Гарантируется, что можно выбрать n из них так, чтобы они образовывали арифметическую прогрессию.

Формат выходного файла

В первой строке выходного файла выведите a_1 и d — первый элемент и разность найденной арифметической прогрессии. Если $d = 0$, число a_1 должно встречаться среди заданных чисел n раз.

Если существует несколько решений, выведите любое.

Примеры

arithm.in	arithm.out
3 8 7 1 5 4 3	1 3

Задача B. Бендер

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

Робот Бендер решил открыть аттракцион «Кручу-Верчу» с целью своего обогащения. Аттракцион состоит в следующем: Бендер прячет шарик под одним из k одинаковых стаканчиков, расположенных на позициях от 1 до k, затем n раз быстро меняет местами какие-то пары стаканчиков, после чего предлагает отгадать под каким из стаканчиков сейчас шарик.

Бендер — робот, поэтому действует он по определенной программе. Бендер строит последовательность целых чисел x_i, при этом $x_1 = c$, а $x_i = a \cdot x_{i-1} + b$ для $i > 1$.

На i-ом шаге Бендер меняет местами стаканчики на позициях с номерами $(x_i \bmod k) + 1$ и $((x_i + 1) \bmod k) + 1$.

В начале робот прячет шарик под стаканчик на позиции с номером r. Бендер хочет, чтобы после n обменов шарик оказался под стаканчиком на позиции с номером l.

Найдите такие a, b и c, чтобы стаканчик с шариком переместился с r-й позиции на l-ю.

Формат входного файла

В единственной строке входного файла четыре целых числа n, k, r и l ($1 \leqslant n \leqslant 10^5$; $2 \leqslant k \leqslant 10$; $1 \leqslant r, l \leqslant k$).

Формат выходного файла

Если таких чисел не существует, выведите в выходной файл единственное слово «`Impossible`». Иначе выведите три целых неотрицательных числа a, b и c. Числа не должны превосходить 1000.

Примеры

bender.in	bender.out
2 3 1 2	0 0 1
3 4 2 4	2 0 1
10 2 1 2	Impossible

Задача C. Различные числа

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

На днях Алиса делала уборку в своей комнате и нашла дневник, который вела в начальной школе. Там она с удивлением обнаружила запись о том насколько ее поразило то, что $2 + 2 = 2 \cdot 2$. Невероятно, умножение и сложение дают один и тот же результат!

Эта запись натолкнула Алису на следующую задачу: пусть целые заданы числа a и b. Сколько различных значений в наборе чисел

$$\begin{array}{llllll} a+b, & a-b, & a\cdot b, & a/b, & a^b, \\ b+a, & b-a, & b\cdot a, & b/a, & b^a. \end{array}$$

Деление происходит без округления, результат деления может не быть целым числом. Если какое-либо выражение из этого набора некорректно, то Алиса его не рассматривает. Некорректными считаются деление на ноль и возведение нуля в неположительную степень.

Формат входного файла

Первая строка входного файла содержит целые числа a и b, разделенные пробелом ($|a|, |b| \leqslant 10^9$).

Формат выходного файла

Выведите в выходной файл количество различных чисел в приведенном наборе.

Примеры

differ.in	differ.out
2 5	8
0 1	3

Задача D. Игра

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

На планете Шелезяка катастрофа — запасы смазки подходят в концу. В связи с этим правительство решило организовать всепланетное соревнование, главный приз в котором — вагон смазочных материалов.

Соревнование проводится в несколько этапов, каждый из которых поделен на множество раундов. В каждом раунде принимают участие два игрока. Жюри предоставляет им большое целое число n. Затем игроки делают ходы по очереди. Первый ход первого игрока заключается в том, что игрок выписывает на специальном поле цифру, причем первым ходом запрещается выписывать ноль. В дальнейшем ход игрока заключается в том, что он приписывает справа к уже написанному числу произвольную цифру. Выигрывает тот, после чьего хода выписанное число больше или равно n.

Знаменитый ученый-робот «ЩК-33» считает, что результат игры легко предсказуем. Для доказательства он решил предоставить программу, которая определяет, кто выигрывает, если оба игрока действуют по оптимальной стратегии. К сожалению, из-за недостатка смазки, его манипуляторы вышли из строя, поэтому он просит вас о помощи.

Формат входного файла

Первая строка входного файла содержит целое число n ($1 \leqslant n \leqslant 10^{100\,000}$). Это число не содержит ведущих нулей.

Формат выходного файла

Выведите «`First`», если при оптимальной игре выигрывает первый игрок, и «`Second`» в противном случае.

Примеры

game.in	game.out
22	First

Задача E. Огромная парковка

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

Джон работает на огромной парковке. Парковка представляется собой прямоугольное поле $n \times m$, разбитое на $n \times m$ квадратных позиций размера 1×1. Одну из угловых позиций занимает выезд с парковки.

Машин на парковке много и вывести машину не так уж просто. Единственное, что Джон может сделать — это переместить один из автомобилей на соседнюю позицию, если она свободна. Соседними считаются позиции, имеющие общую сторону. Однако задача усложняется наличием на парковке столбов. На позиции, где стоят столбы, нельзя поставить машину. Парковка вся занята машинами и столбами и единственное свободное место — выезд с парковки. Задача Джона — вывести с парковки один из автомобилей. Помогите ему узнать, какое минимальное число действий ему придется совершить.

Формат входного файла

В первой строке входного файла два целых числа n и m ($1 \leqslant n, m \leqslant 50$) — размеры парковки. Далее следуют n строк по m символов в каждой. Символ «`.`» означает пустую позицию, единственная пустая позиция — выезд с парковки. Символ «`#`» означает столб. Столбы нельзя перемещать и на место столба нельзя ставить автомобили. Символ «`c`» означает автомобиль. Символ «`X`» — автомобиль, который необходимо вывести с парковки. Автомобиль считается выведенным, как только он достигает выезда с парковки. Гарантируется, что хотя бы одно из чисел n, m больше единицы и каждый из символов «`.`» и «`X`» встречается во входном файле ровно один раз. Символ «`.`» всегда располагается в верхнем левом углу парковки.

Формат выходного файла

Если машину вывести невозможно, выведите в выходной файл единственное слово «`Impossible`». Иначе в единственной строке выведите единственное число — минимальное количество действий для вывода автомобиля.

Примеры

parking.in	parking.out
3 3 .#X ccc c#c	Impossible
2 3 .cX ccc	7

Задача F. День рождения

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

У Коли сегодня день рождения! По этому случаю он решил после олимпиады сходить с друзьями в парк аттракционов. И какая удача — можно купить групповой билет сразу на всех, всего за S рублей!

Конечно, скидываться придется всем поровну. То есть, если Коля позовет k своих друзей, то каждому придется заплатить $S/(k+1)$ рублей (да, сам Коля тоже должен внести свою долю). При этом S не обязательно должно делиться на $k+1$: главное — купить билет, а между собой друзья уж как-нибудь договорятся.

Всего у Коли n друзей, при этом i-й из них готов пойти с Колей в парк, если доля, которую ему придется заплатить не больше b_i (больше денег у него просто с собой нет) и не меньше a_i (иначе он решит, что Колин день рождения — это скучно, и пойдет играть в волейбол с Сережей).

Так что может так получиться, что всех позвать не удастся. Ну и ладно. Для каждого своего друга Коля знает число f_i — количество веселья, который тот произведет, если его позвать.

Помогите Коле выбрать подмножество друзей, которых Коля должен позвать с собой, чтобы максимизировать суммарное веселье.

Формат входного файла

В первой строке входного файлы содержится два целых числа: n и S ($1 \leqslant n \leqslant 100\,000$, $0 \leqslant S \leqslant 10^9$) — количество друзей Коли и стоимость билета. В следующих n строках содержится по три целых числа: в i-й из этих строк находятся числа a_i, b_i и f_i ($0 \leqslant a_i \leqslant b_i \leqslant S$, $0 \leqslant f_i \leqslant 10^9$). Они означают, что i-го друга можно позвать на вечеринку, если доля, которую ему придется заплатить, лежит между a_i и b_i, и он произведет f_i веселья.

Формат выходного файла

В первой строке выходного файла выведите два числа: k (количество приглашенных на вечеринку друзей) и F (максимальное суммарное веселье, которое можно получить). Во второй строке выведите k чисел — номера друзей, которых нужно пригласить.

Пример

party.in	party.out
4 10 4 5 40 2 4 30 2 6 10 3 5 20	2 50 2 4

Задача G. Гонка со временем

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

Директор школы «Лицей Программистов» города Линейска сумел привить ученикам этой школы хорошую дисциплину, но, несмотря на это, многие ученики продолжают опаздывать на уроки.

В Линейске есть всего одна главная улица, на которой расположен и сам лицей, и живут все его ученики. Дисциплинированные школьники выходят из своих домов в одно и то же время. Но, к сожалению, все они живут на разном расстоянии от школы и добираются с разной, но постоянной скоростью (среди учеников есть как весьма неторопливые, так и будущая чемпионка мира по бегу на 100 метров Маша Гайка).

Для улучшения ситуации с опозданиями школа купила один веломобиль, который взялся водить сторож школы. Веломобиль способен помимо водителя перевозить одного школьника. Веломобиль перемещается с постоянной скоростью v.

Ночь веломобиль проводит в школьном гараже, а утром, ровно в тот момент, когда все ученики выходят из своих домов, отправляется им навстречу, чтобы подвезти какого-нибудь школьника. Конечно же, ему приходится подвозить школьника прямо до ворот школы, потому как никто не хочет быть высаженным посреди пути. После того, как веломобиль помог одному ученику быстрее добраться в учебное заведение, он снова может ехать навстречу следующему опаздывающему человеку.

Директор хочет начинать занятия как можно раньше, но для этого нужно, чтобы все ученики добрались до лицея. Он поручил вам помочь составить план, по которому должен действовать водитель веломобиля, чтобы время прибытия последнего школьника в школу было минимальным.

Формат входного файла

Первая строка входного файла содержит два целых числа n, v ($1 \leqslant n \leqslant 10^5$, $1 \leqslant v \leqslant 1000$) — соответственно количество людей и скорость веломобиля. Следующие n строк содержат по два целых числа x_i, v_i ($1 \leqslant x_i, v_i \leqslant 1000$) — расстояния от школы до дома i-ого ученика и его скорость.

Формат выходного файла

В первую строку выходного файла выведите вещественное число t — минимальное время, за которое все школьники доберутся до лицея. Во второй строке выходного файла выведите единственное число k — количество учеников, которых нужно подвезти. В следующих k строках выведите по два числа — номер школьника, которого нужно подвезти, и расстояние от школы, на котором этот школьник должен сесть в веломобиль.

Школьников нужно выводить в том же порядке, в котором они будут ехать на веломобиле.

Выводите все вещественные числа как можно точнее. При проверке вашего решения при сравнении вещественных чисел будет допускаться абсолютная или относительная погрешность 10^{-6}.

Примеры

race.in	race.out
5 4 1 1 4 2 3 1 7 5 5 1	2.4 2 5 4 3 0.8

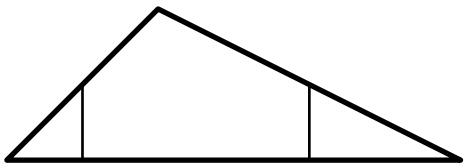
Задача H. Скоростной диаметр для кольцевой дороги

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

Новый мэр крупного города Флатбурга Иван Котянин начал свою работу с решения проблем с пробками в городе. Как и в любом крупном городе, во Флатбурге есть кольцевая автодорога. Во Флатбурге она представляет собой монотонный многоугольник. Монотонным называется многоугольник, с которым каждая прямая, проходящая строго с севера на юг, имеет не более двух общих точек.

После совещания с правительством города было принято решение построить новую магистраль — скоростной диаметр, который вел бы строго с севера на юг и соединял две точки кольцевой автодороги.

Помимо борьбы с пробками решено было обновить рекорд по протяженности самой длинной магистрали, проложенной с севера на юг. Для того чтобы обновить рекорд необходимо построить магистраль длиной хотя бы d километров, а поскольку лишних денег в бюджете Флатбурга не так уж и много, решено было построить дорогу длиной ровно d.



Министр транспорта Флатбурга решил предоставить мэру все варианты строительства новой дороги. Для начала он решил подсчитать, сколько существует способов построить магистраль. Помогите министру.

Формат входного файла

Первая строка входного файла содержит два целых числа n — количество вершин у многоугольника, задающего кольцевую автодорогу ($3 \leqslant n \leqslant 100000$), и d — длину новой магистрали ($1 \leqslant d \leqslant 10^8$).

Далее следует описание расположения вершин — каждая из следующих n строк содержит координаты x и y ($-10^8 \leqslant x, y \leqslant 10^8$) соответствующей вершины. Вершины заданы в порядке их обхода против часовой стрелки.

Направлению с севера на юг соответствуют прямые, задаваемые уравнением $x = c$ для некоторого c. Заданный многоугольник является монотонным.

Формат выходного файла

В выходной файл выведите количество способов построить дополнительную магистраль длиной ровно d. Если способов постройки бесконечно много, выведите в выходной файл «`Infinity`».

Примеры

ring.in	ring.out
3 1 0 0 6 0 2 2	2
4 4 0 0 3 0 4 4 1 4	Infinity

Задача I. Откат

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

Сергей работает системным администратором в очень крупной компании. Естественно, в круг его обязанностей входит резервное копирование информации, хранящейся на различных серверах и «откат» к предыдущей версии в случае возникновения проблем.

В данный момент Сергей борется с проблемой недостатка места для хранения информации для восстановления. Он решил перенести часть информации на новые сервера. К сожалению, если что-то случится во время переноса, он не сможет произвести откат, поэтому процедура переноса должна быть тщательно спланирована.

На данный момент у Сергея хранятся n точек восстановления различных серверов, пронумерованных от 1 до n. Точка восстановления с номером i позволяет произвести откат для сервера a_i. Сергей решил разбить перенос на этапы, при этом на каждом этапе в случае возникновения проблем будут доступны точки восстановления с номерами $l, l+1, \ldots, r$ для некоторых l и r.

Для того, чтобы спланировать перенос данных оптимальным образом, Сергею необходимо научиться отвечать на запросы: для заданного l, при каком минимальном r в процессе переноса будут доступны точки восстановления не менее чем k различных серверов.

Помогите Сергею.

Формат входного файла

Первая строка входного файла содержит два целых числа n и m, разделенные пробелами — количество точек восстановления и количество серверов ($1 \leqslant n, m \leqslant 100\,000$). Вторая строка содержит n целых чисел $a_1, a_2, \ldots, a_n$ — номера серверов, которым соответствуют точки восстановления ($1 \leqslant a_i \leqslant m$).

Третья строка входного файла содержит q — количество запросов, которые необходимо обработать ($1 \leqslant q \leqslant 100\,000$). В процессе обработки запросов необходимо поддерживать число p, исходно оно равно 0. Каждый запрос задается парой чисел x_i и y_i, используйте их для получения данных запроса следующим образом: $l_i = ((x_i + p) \bmod n)+1$, $k_i = ((y_i + p) \bmod m)+1$ ($1 \leqslant l_i, x_i \leqslant n$, $1 \leqslant k_i, y_i \leqslant m$). Пусть ответ на i-й запрос равен r. После выполнения этого запроса, следует присвоить p значение r.

Формат выходного файла

На каждый запрос выведите одно число — искомое минимальное r, либо 0, если такого r не существует.

Примеры

rollback.in	rollback.out
7 3 1 2 1 3 1 2 1 4 7 3 7 1 7 1 2 2	1 4 0 6

Задача J. Рыцарский щит

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

Погостив пару недель у Темного Властелина и прослушав истории о всех его похождениях за последние годы, сэр Петрейн понял, что он уже давно не совершал никаких подвигов. Посидев за чашкой чая и тщательно обсудив будущий подвиг, они решили, что Петрейну нужно победить *ужасного дракона*, который уже давно терроризирует западные окраины Личного королевства. И вот он отправился готовиться к великому походу.

Но какой рыцарь идет на дракона без рыцарского обмундирования? Поэтому Петрейну нужны доспехи, щит и меч. Всем известно, что чем щит больше, тем эффективней будет он в бою. Сейчас у Петрейна есть два треугольных щита, но он считает их недостаточно надежными и хочет сделать из них один.

Королевский оружейник, взявшийся за изготовление щита, предложил следующий способ: два имеющихся щита кладутся рядом так, чтобы они соприкасались сторонами и фиксируются в таком положении. Сэр Петрейн заметил, что как бы оружейник не старался, у полученного в результате щита всегда будет одинаковая площадь, а значит его эффективность в бою с драконом будет зависеть только от того, какие щиты дал Петрейн оружейнику, но не от того, как они скреплены.

Но ему нужен не просто кусок металла, а щит с символикой его рода: золотым обрамлением по периметру. Однако золото сейчас дорого, поэтому Петрейну хочется, чтобы периметр полученного щита был как можно меньше. Помогите ему выяснить, какой минимальный периметр может иметь щит.

Формат входного файла

В первой строке заданы три числа a_1, b_1 и c_1 — длины сторон первого щита. Во второй строке заданы три числа a_2, b_2 и c_2 — длины сторон второго щита. Обе строки задают корректные невырожденные треугольники. Все числа во входном файле не превосходят 100 000.

Формат выходного файла

Выведите единственное число — минимальный периметр щита, который можно изготовить из них указанным способом.

Примеры

shield.in	shield.out
1 1 1 1 1 1	4
3 4 5 8 7 6	23

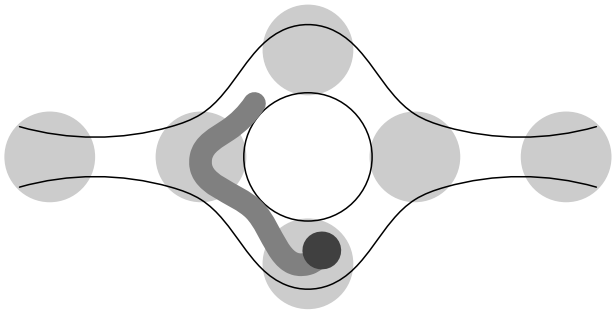
Задача K. Змея в метро

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

2390 год. В заброшенном метрополитене города N-ска завелась громадная змея-мутант. Она ползает вдоль перегонов между станциями, повергая в ужас случайно забредающих под землю потомков людей. Размеры змеи настолько велики, что иногда голова появляется на той станции, вдоль которой еще ползет какая-то другая часть тела, и змея повергает в ужас сама себя. Чтобы избавиться от этой проблемы, змея поймала вас и потребовала написать для нее программу, которая может ей прокладывать кратчайший маршрут для головы от одной станции до другой, не проползая при этом по станциям, где находятся участки ее тела.

Будем называть маршрутом последовательность станций, каждые две последовательные из которых соединены перегоном.

Все перегоны в метрополитене имеют одинаковую длину, а змея имеет длину в $(l-0.5)$ перегонов. Змея может ползти вдоль перегонов, переползая с одного на другой на станциях. Змея может ползти вдоль перегона только в один слой, а ее голова не может появляться на станции, если в этот момент по станции проползает другая часть ее тела. Змея умеет ползать только головой вперед.



С точки зрения теории графов метрополитен города N-ска является вершинным кактусом. Это означает, что ни одна станция не лежит на двух различных циклических маршрутах и никакие два перегона не соединяют одну и ту же пару станций, никакой перегон не соединяет станцию саму с собой, от каждой станции до любой другой до появления змеи можно было добраться по перегонам.

По заданным карте метрополитена, начальному положению змеи и станции, на которую змея хочет поместить свою голову, выясните, какое минимальное количество перегонов придется проползти змее.

Формат входного файла

В первой строке ввода записано два числа n и k — количество станций и количество перегонов в метрополитене ($1 \leqslant n, k \leqslant 100\,000$). В следующих k строках записано по два различных целых числа a и b — номера станций, соединенных соответствующим перегоном.

В следующей строке записано единственное число l, характеризующее длину змеи. В следующей строке записано $l + 1$ число: номера станций, на которых лежат последовательные части змеи, начиная с головы, а также номер станции, в перегоне к которой лежит хвост змеи длиной в 0.5 перегона. Исходно змея расположена таким образом, что ни в каком перегоне не находится одновременно две различных части змеи и змея не пересекает себя ни на какой станции.

В последней строке записано единственное целое число — станция, на которую змея хочет поместить свою голову.

Формат выходного файла

Если змея сможет выполнить свою задачу, выведите длину пути — количество перегонов, через которые необходимо проследовать голове змеи.

Если задача невыполнима, выведите единственное число -1.

Примеры

snake.in	snake.out
6 6 1 2 2 3 3 4 4 5 5 2 4 6 2 5 2 3 1	4