Задача A. Бестин

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

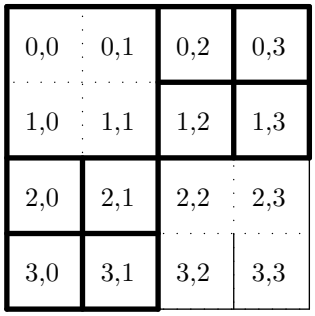
Бестин — столица планеты Татуин. Он знаменит огромным количеством гангстеров и контрабандистов, а также своей непростой географией.

Город представляет собой клетчатый прямоугольник $n \times m$ километров, разделенный на $n \cdot m$ кварталов. Каждый квартал — клетка этого прямоугольника. Во времена своего основания город был полностью отделен стеной от внешнего мира. Кроме того, все пары соседних кварталов также были отделены друг от друга стенами.

Каждый квартал однозначно характеризуется своими координатами. Первая координата квартала равна количеству кварталов между ним и северной стеной города, вторая координата квартала — количеству кварталов между ним и западной стеной города. В процессе развития некоторые стены, разделяющие соседние кварталы, были снесены.

Площадью в Бестине называют некоторый подпрямоугольник, состоящий из $x \cdot y$ кварталов. На площадь накладываются следующие ограничения:

- площадь по всему периметру отделена стенами от остального города
- никакие два соседних квартала, находящиеся внутри площади, не разделены стеной



На рисунке отмечены координаты кварталов. Пунктирные линии обозначают снесенные стены, сплошные линии обозначают оставшиеся стены.
В городе, изображенном на рисунке, ровно 9 площадей.

Вы исследуете историю развития Бестина. Вам необходимо определить количество площадей, которые присутствовали в городе после каждой снесенной стены.

Формат входного файла

Первая строка входного файла содержит два целых числа n и m ($1 \le n, m \le 100\,000$) — размеры города. Вторая строка содержит одно число k ($1 \le k \le 100\,000$) — количество стен, снесенных за время существования города. В следующих k строках содержатся описания снесенных стен. Каждая стена описывается двумя парами координат тех кварталов, между которыми находилась снесенная стена. Координаты кварталов — целые неотрицательные числа, первое из которых меньше n, а второе — меньше m.

Формат выходного файла

В выходной файл выведите k чисел, разделенных переводами строк. i-е число должно быть равно количеству площадей в городе после того, как были снесены i стен.

Примеры

bestin.in	bestin.out
4 4 7 0 0 0 1 0 0 1 0 1 0 1 1 1 1 0 1 3 3 2 3 3 2 2 2 2 2 2 3	15 13 12 13 12 11 9

Задача B. Склад Оби-Вана Кеноби

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

Оби-Ван Кеноби — один из последних представителей ордена рыцарей джедаев. Оби-Ван Кеноби родом с планеты Стьюджон, где абсолютно все жители чтут порядок, и Оби-Ван Кеноби — не исключение.

Во всех эпизодах «Звездных войн» Оби-Ван Кеноби орудует синим световым мечом, но мало кто знает, что на самом деле у него их много, причем все мечи Кеноби пронумерованы. Оби-Ван Кеноби хранит все свои мечи на очень длинном столе. Когда он хочет вооружиться, то он берет самый правый меч со стола, и идет по своим делам. Оби-Ван Кеноби орудует взятым мечом пока не потеряет или не сломает его.

Иногда кто-нибудь дарит Оби-Вану Кеноби новый меч, и тогда он просто кладет его на стол справа к тем, что уже лежат там. Но самое страшное случается тогда, когда к столу подходит мама Оби-Вана Кеноби. Мама тоже житель планеты Стьюджон, и поэтому она всегда возмущается, что мечи лежат неупорядоченно. Чтобы исправить это, она забирает всю левую половину мечей (если мечей на столе было нечетное количество, то мама забирает наибольшее целое число мечей, меньшее половины их общего количества) и начинает их подкладывать справа к оставшимся. Причем сначала она кладет меч, который изначально был самым левым, потом правее от него кладет второй меч и так далее.

Определите, в каком порядке лежат мечи у Оби-Вану Кеноби на данный момент. У каждого меча есть свой личный номер, который известен только Оби-Вану Кеноби. Ну и, конечно, вам.

Формат входного файла

В первой строке входного файла дано натуральное числа n — количество изменений, произошедших на столе ($1 \le n \le 10^6$). В следующих n строках заданы описания изменений в следующем формате:

- если строка начинается со слова `add`, то в той же строке через пробел находится число x ($1 \le x \le n$) — личный номер меча, подаренного Оби-Вану Кеноби. Гарантируется, что мечей с таким номером Оби-Вану Кеноби раньше не дарили и больше не подарят.
- если строка содержит единственное слово `take`, то это обозначает, что Оби-Ван Кеноби забрал самый правый меч со стола с собой. Если на столе не было мечей, то ничего не произошло.
- если же строка содержит единственное слово `mum!`, то это обозначает, что к столу подошла мама и навела порядок. Если на столе было меньше двух мечей, то ничего не произошло.

Изначально стол был пуст.

Формат выходного файла

В первой строке выходного файла выведите число k — количество мечей, лежащих на данный момент на столе у Оби-Вана Кеноби. В следующей строке выведите k чисел через пробел — личные номера мечей, лежащих на столе, в том порядке, в котором они на нем лежат (слева направо).

Примеры

kenobi.in	kenobi.out
8 add 1 add 2 add 4 add 3 add 5 add 8 take mum!	5 4 3 5 1 2

Пояснение

После выполнения операций `add` и `take` на столе будет лежать пять мечей в следующем порядке: 1 2 4 3 5. После чего придет мама Оби-Вана Кеноби, возьмет мечи с номерами 1 и 2, и положит их права от оставшихся трех мечей на столе.

Задача C. Покраска забора

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

Юному Энакину Скайуокеру магистр Йода поручил, на первый взгляд, простое задание. Ему необходимо было покрасить забор специальной краской. Задание состояло из n подзадач. Каждая из подзадач заключалась в покраске некоторого участка забора краской заданного цвета. Каждый цвет обозначался целым числом. Краска, которая была дана Энакину, как и любая другая, после того, как высыхала, меняла свой цвет. Но это происходило немного необычным образом.

Будем говорить, что два участка забора пересекаются, если они имеют не нулевую площадь пересечения. Разобьем все участки забора, на которые была нанесена краска, не некоторые множества. Вначале каждый отрезок отнесем к отдельному множеству. Если существуют пересекающиеся участки, которые относятся к различным множествам A и B, обьединим A и B. Будем делать так до тех пор, пока такие участки не перестанут существовать.

Теперь для каждого множества можно определить его цвет как среднее арифметическое всех цветов входящих в него участков. Зная эту величину для каждого множества, мы легко можем определить, какого цвета будет некоторая точка забора после высыхания краски. Будем говорить, что точка принадлежит множеству, если есть хотя бы один участок в множестве, который содержит в себе эту точку. Если точка забора не принадлежит ни одному множеству, она будет иметь цвет 0, иначе она будет иметь цвет соответствующего множества. Заметим, что точка не может принадлежать сразу двум множествам.

Пока Энакин занимался покраской забора, ему стало интересно, что же случится, если он закончит покраску после нанесения первых i участков. В частности, его заинтересовал такой вопрос «Точку какого наибольшего цвета можно будет найти на заборе после того, как краска высохнет?».

Формат входного файла

Первая строка входного файла содержит одно целое число n ($1 \le n \le 10^5$) — количество участков забора, которые необходимо покрасить. В следующих n строках находится по три целых числа l, r, c ($0 \le l < r \le 10^9, 1 \le c \le 10^9$) — левая и правая границы i-го участка (левая граница включается, правая нет), а также цвет, в который необходмо покрасить соответствующий участок.

Формат выходного файла

В n строках необходимо вывести по одному вещественному числу с не менее чем тремя знаками после запятой. Число в i-ой строке должно обозначать максимальный цвет точки на стене, который можно было бы найти, если бы Энакин закончил покраску забора после выполнения первых i подзадач.

Примеры

painting.in	painting.out
6 0 1 1 2 5 2 3 7 3 6 8 4 9 11 5 7 10 6	1.00000 2.00000 2.50000 3.00000 5.00000 4.00000

Задача D. Пари

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

Как известно, чтобы освободить Энакина Скайуокера и приобрести гипердрайв, Квай-Гон Джинн заключил пари с Уотто о том, кто выиграет гонку Бунта-Ив. Но далеко не всем известно, что Уотто поступил так, как часто поступают на Татуине: схитрил при заключении пари.

Чтобы официально зарегистрировать спор на Татуине, его участники должны обратиться к соответствующему чиновнику, который выдаст им две одинаковые расписки. На расписках будет записана одна и та же строка из символов, которые через много-много лет в некоторой далекой-далекой галактике назовут латинскими буквами.

Хитрость Уотто заключалась в следующем: его знакомый чиновник составил нестандартные расписки. Чтобы не нарушать закон, он составил две корректные расписки, а затем стал их менять. Он произвольным образом выбирал подстроку, которая является корректной записью натурального числа с помощью того, что в упомянутой далекой галактике назовут римскими цифрами. После этого он заменял эту подстроку записью того же числа символами, которые во все той же далекой галактике назовут арабскими цифрами, причем так, что в записи числа не было лидирующих нулей. Чиновник повторял это действие до тех пор, пока подходящих подстрок не осталось.

Как известно, обмануть Квай-Гона у Уотто не получилось, так что об этой хитрости никто никогда не упоминал. А сможете ли вы, житель далекой-далекой галактики, проверить, были ли надписи на двух нестандартных расписках получены из одной строки?

Формат входного файла

В первой и второй строке входного файла записаны две строки, состоящие их маленьких латинских букв и цифр. Длина каждой строки не превышает 10^5. Гарантируется, что никакая подстрока данных строк не является корректной записью числа римскими цифрами. Гарантируется, что все подстроки, являющиеся натуральными числами, не превышают 200 по значению. Гарантируется, что никакая подстрока, состоящая из цифр, не начинается с нуля.

Формат выходного файла

В выходной файлы выведите `YES`, если данные строки могли быть получены из одной, и `NO` — в противном случае.

Примеры

bet.in	bet.out
skywa50kera21 skywa50kera12	YES
qu1gonj20 qu1gonj200	NO

Пояснение

В римской системе счисления `I` соответствует единице, `V` — пяти, `X` — десяти, `L` — пятидесяти, `C` — ста, `D` — пятистам, `M` — тысяче. Для правильной записи чисел римскими цифрами необходимо сначала записать число тысяч, затем сотен, затем десятков и, наконец, единиц. Цифры могут повторяться, но не более трех раз. Меньшая цифра может быть записана слева от большей, тогда её следует вычесть из большей. В этом случае повторения меньшей цифры не допускаются. Существует шесть вариантов использования правила вычитания: `IV` — 4, `IX` — 9, `XL` — 40, `XC` — 90, `CD` — 400, `CM` — 900. Другие способы вычитания не допустимы.

Задача E. Генерал Гривус

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

Генерал Гривус — мастер владения световым мечом. Его уровень мастерства настолько высок, что он может одновременно использовать n световых мечей. Однако, несмотря на это, в битве с Энакином и Оби-Ваном он почувствовал их превосходство и решил ретироваться с поля боя. Для этого он решил ошеломить их с помощью припасённой на похожий случай гранаты. Он знает, что граната взрывается ровно через t секунд после её активации. Но использовать в бою часы для того, чтобы отмерить заданный промежуток времени — расточительность, и он хочет использовать для этого что-нибудь более подходящее.

После установки на световые мечи нового программного обеспечения, они стали быстрее разряжаться. Каждый из n имеющихся у Гривуса мечей разряжается за m секунд непрерывной работы одного луча. Соответственно, если он включен с двух сторон, то он разряжается за $\frac{m}{2}$ секунд. В начальный момент Гривус может включить некоторые мечи с некоторых сторон и вести бой ими. Затем, как только какой-либо из мечей разрядился, он может в тот же момент поменять состояние некоторых мечей до следующей разрядки.

Гривус хочет провести битву так, чтобы суметь отмерить промежуток времени, составляющий ровно время, необходимое для взрыва гранаты. Проверьте, сможет ли он это сделать. Гривус может включить гранату вместе с любым из мечей, а взорваться она должна в момент разрядки какого-нибудь меча.

Пусть у него есть два меча, каждый из которых разряжается за четыре секунды, а время до взрыва гранаты — одна секунда. Тогда пусть он ведёт бой следующим образом. Сначала он включает первый меч с двух сторон, а второй с одной. Когда разряжается первый меч, он активирует гранату и включает второй меч со второй стороны. Тогда в момент активации гранаты, второму мечу осталось работать две минуты. После включения его со второй стороны, это время сокращается до одной минуты, сколько и необходимо получить.

Формат входного файла

В единственной строке заданы целые числа n ($1 \le n \le 5$), t ($1 \le t \le 10^5$) и m ($1 \le m \le 10^5$) — количество доступных Гривусу мечей, время от активации гранаты до её взрыва и время работы каждого меча.

Гарантируется, что все отрезки времени, которые сможет отмерить Гривус, целые.

Формат выходного файла

Выведите `Yes`, если он сможет отмерить необходимый отрезок времени, и `No` иначе.

Примеры

grievous.in	grievous.out
2 1 4	Yes
5 640 128	Yes
2 1 8	No

Задача F. Парад победы

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

В честь уничтожения Звезды Смерти и победы над Империей, повстанцы устроили парад. В параде участвуют n человек. У каждого из них есть какое-то количество наград. Оказалось, что у каждых двух человек количество наград различно.

Организаторы парада ответственно отнеслись к его проведению. Участники парада по очереди выходят на площать и строятся в линию. При этом, очередной выходящий проходит мимо всех уже стоящих на площади. Проходя мимо обладающего бóльшим числом наград человека, проходящий обязан остановиться и отдать ему честь. На это уходит некоторое одинаковое, прописанное в уставе, время.

Во время подготовки все строго обговорили порядок, в котором будут стоять на площади. Но им стало известно, что на самом параде будет выставлена техника, которая загородит один из входов на площадь. К сожалению, ещё не известно будет ли загорожен левый или правый вход. Но, так как вся программа парада выверена до секунды, то, с какой стороны участники будут выходить на площадь, не должно сказаться на том, сколько времени им на это потребуется. Время, которое потребуется на то, чтобы выйти всем участникам, зависит только от количества раз, которое будет отдана честь.

Если будет закрыт правый выход, то люди будут выходить следующим образом: сначала тот, кто должен стоять левее всех, затем тот, кто должен стоять вторым, пройдя мимо уже стоящего первого, и так далее. Если же закрыт левый выход, то сначала выходит тот, кто будет стоять самым правым, затем тот, кто будет стоять вторым справа, проходя мимо уже стоя́щего правого, и так далее.

Интересными являются только те порядки построения, при которых время парада при выходе участников спарва и слева будет одинаковым. Вам требуется определить их количество.

Формат входного файла

В единственной строке входного файла записано единственное целое число n ($1 \le n \le 500$).

Формат выходного файла

Выведите единственное число — количество способов, по модулю $10^9 + 7$.

Примеры

victory.in	victory.out
3	0
4	6

Пояснение

Если количества наград у участников парада во втором примере равны числам 1, 2, 3 и 4, то нас интересуют следующие порядки построения:

- 1, 4, 3, 2
- 2, 3, 4, 1
- 2, 4, 1, 3
- 3, 1, 4, 2
- 3, 2, 1, 4

- 4, 1, 2, 3

Задача G. Скользкий путь

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

Люк Скайуокер ступил на скользкий путь! К счастью, его не влечет Темная сторона. Он всего лишь оказался на планете Хот, целиком покрытой снегом и льдом, потому передвигаться по местности необходимо крайне осторожно. Люку необходимо добраться из точки A в точку B и доставить ценную и хрупкую посылку.

Для простоты будем считать, что местность разбита на квадраты и представляет из себя прямоугольник размером n на m. Каждая клетка может быть одного из трех типов: здание, сугроб, либо лед. Перемещаться Люк может только между соседними клетками. Соседними считаются клетки, имеющие общую сторону. Перемещение между любыми двумя клетками занимает ровно одну единицу времени. Каждая клетка имеет свою высоту — целое число.

Между соседними клетками зданий можно перемещаться без каких-либо ограничений. Также из здания можно переместиться в соседний сугроб. Из сугроба можно попасть в соседнее здание. Из сугроба можно перейти либо в соседний сугроб, либо на соседнюю клетку со льдом, если высота новой клетки не больше изначальной. Из клетки со льдом можно переходить в соседнюю клетку со льдом или сугробом, если высота новой клетки не больше изначальной. Если же Люк переходит из клетки со льдом в клетку со льдом, высота которой строго меньше, то он начинает скользить в том же направлении. Люк останавливает скольжение, если следующей клеткой на его пути встречается сугроб, клетка со льдом, высота которой больше высоты той клетки, в которой он находится, либо край карты. В этом случае Люк снова может идти в любом направлении из той клетки, в которой он остановился. Если же Люк встречает на своем пути стену здания, то он оказывается не в силах остановить движение и разбивает посылку.

Люку необходимо как можно скорее доставить это посылку из точки A в точку B. Помогите ему найти кратчайший путь, при котором его груз уцелеет.

Формат входного файла

В первой строчке входного файла заданы два числа n и m ($1 \le n, m \le 500$) — размеры карты. Во второй строке находятся два числа a_r, a_c ($1 \le a_r \le n, 1 \le a_c \le m$) — номер строки и номер столбца, в которых находится точка A. В третьей строке находятся два числа b_r, b_c — описание точки B в том же формате. В каждой из следующих n строчек находится m символов. Если соответствующий символ равен B, то в этой клетке находится здание, S — сугроб и I — лед. В следующих n строчках находится описание рельефа местности. В каждой из этих строчек — по n целых чисел — высота соответствующей клетки на карте. Высота каждой точки — целое число h_{ij} ($0 \le h_{ij} \le 10^9$).

Гарантируется, что точки A и B находятся в зданиях.

Формат выходного файла

В выходной файл выведите длину кратчайшего пути из A в B, либо «`Impossible`», если пути не существует.

Примеры

icy.in	icy.out
3 5 1 1 3 5 BISSS SIIIS SSSIB 10 10 5 5 5 10 10 5 3 1 10 10 5 5 1	6

Задача H. Война клонов

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

Армия клонов, созданная на планете Камино, помогала джедаям сражаться против дроидов. Все изменилось, когда Палпатин отдал «Приказ 66»: без предупреждения клоны напали на джедаев. Но не все еще потеряно для Светлой стороны. Некоторые клоны не услышали приказ и продолжили сражаться на стороне джедаев. Поэтому война не закончилась, когда все джедаи были убиты. По приказу Палпатина все клоны выстроились в одну линию для финальной битвы.

Магистр Йода понял, что он должен научиться отличать имперских клонов и клонов Республики, которые остались верны джедаям. Оглядев поле, Йода присвоил каждому клону координату, соответствующую месту, где стоит клон. С помощью Силы магистр изменил расстановку клонов на поле так, что расстояние между любыми двумя клонами одной стороны стало четно, а между любыми двумя клонами вражеских сторон — нечетно. При этом никакие два клона не стоят на одной позиции. И началась война клонов...

Чтобы оценить шансы на победу, магистр Йода хочет знать количество пар клонов, которые могут сразиться между собой, то есть количество пар таких, что клоны в паре служат разным сторонам.

Формат входного файла

В первой строке входного файла задано число n ($1 \le n \le 100000$) — число клонов, участвующих в финальной битве. В следующей строке заданы n чисел a_i ($1 \le a_i \le 10^9$) — координаты клонов, все a_i различны.

Формат выходного файла

В выходной файл выведите единственное числов — количество пар клонов, которые могут сразиться между собой.

Примеры

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

Задача I. Спичрайтер Йоды

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

Всем известно, что у каждого важного общественного деятеля есть собственный спичрайтер — существо, помогающее подготовить публичную речь, сделать ее более выразительной и интересной. Глава Ордена джедаев магистр Йода не исключение.

На первый взгляд может показаться, что спичрайтеру Йоды приходится тяжелее других: все-таки речь магистра своеобразна и ее изучение требует серьезных усилий. На самом деле все несколько проще. Спичрайтеру Йоды достаточно сначала придумать речь для обычного человека, после чего поменять порядок слов в каждом предложении на обратный. В силу того, что алгоритм преобразования обычной речи в речь магистра Йоды достаточно однообразен, спичрайтер решил автоматизировать этот процесс и попросил вас написать программу, которая будет преобразовывать речь, составленную им для обычного человека в речь для Йоды.

Формат входного файла

В единственной строке входного файла задана речь, составленная спичрайтером. Речь состоит из предложений, отделенных друг от друга точками (точка ставится сразу после последнего слова в предложении).

Каждое предложение состоит из слов. Предложение содержит по крайней мере одно слово. Соседние слова разделены ровно одним пробелом. Слово — непустая последовательность строчных латинских букв. Строка не содержит лишних пробелов. Гарантируется, что строка не пуста и ее длина не превосходит 20000 символов.

Формат выходного файла

В одной строке выведите преобразованную для Йоды речь в соответствии с форматом входных данных. Следует строго соблюдать формат вывода речи, описанный во входных данных.

Примеры

yoda.in	yoda.out
you should solve this problem. its easy.	problem this solve should you. easy its.

Задача J. Печеньки

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

Юный Люк — очень одаренный джедай. Кроме недюжинного таланта влезать в неприятности он умеет перемещать предметы, не касаясь их. К большому сожалению юноши, уроки телекинеза начинаются только с пятого класса школы джедаев, зато он нашел в библиотеке отца старинную книгу «Телекинез для самых маленьких».

Первым упражнением в книге было перемещение печенья в пространстве силой мысли. У Люка как раз оказалась под рукой целая пачка четырехугольных печенюшек, поэтому он решил попробовать свои способности. Чтобы переместить куда-нибудь четырехугольную печенюшку силой мысли, как сказано в книге, нужно сначала разломить ее на ровно три трапециевидных кусочка, после чего сосредоточиться и использовать силу.

Помогите Люку разломить печенье таким образом, чтобы образовалось ровно три кусочка, каждый из которых имеет форму трапеции (выпуклого четырехугольника, две стороны которого параллельны).

Формат входного файла

Во входном файле задан четырехугольник — печенье. В четырех строках заданы координаты вершин четырехугольника в порядке обхода против часовой стрелки. В i-й строке два числа x_i и y_i — координаты i-й вершины четырехугольника.

Никакие три точки не лежат на одной прямой.

Все координаты по модулю не превышают 10 000.

Формат выходного файла

В выходной файл выведите три четверки точек. Каждая четверка точек задает трапецию. Каждая трапеция описывается четырьмя строками, в каждой из которых по два вещественных числа: x_i и y_i — координаты i-й вершины трапеции в порядке обхода против часовой стрелки. Трапеция должна иметь ненулевую площадь. После каждого описания трапеции требуется вывести пустую строку. Все точки трапеции должны находится внутри или на границе исходного четырехугольника. Трапеции не должны иметь пересечений ненулевой площади, и каждая точка исходного четырехугольника должна быть покрыта хотя бы одной из трех трапеций.

Если существует несколько решений, выведите любое. При проверке ответа на корректность площади будут равны если относительная либо абсолютная погрешность не больше 10^{-5}, точка лежит на прямой, если расстояние от точки до прямой не больше 10^{-8}. Прямые параллельны, если синус угла между ними не больше 10^{-7}.

Примеры

cookies.in	cookies.out
0 0 2 0 2 2 0 2	0 0 1 0 1 1 0 1 1 0 2 0 2 1 1 1 0 1 2 1 2 2 0 2
0 0 20 20 0 10 -20 20	0 0 11 11 3 7 -2 2 3 7 11 11 20 20 0 10 -2 2 3 7 0 10 -20 20

Пояснение

Обращаем ваше внимание на то, что у получившихся у вас трапеций обе пары противоположных сторон **могут** быть параллельны друг другу. Такой случай рассматривается в первом примере.

Второй пример изображен на следующем рисунке: