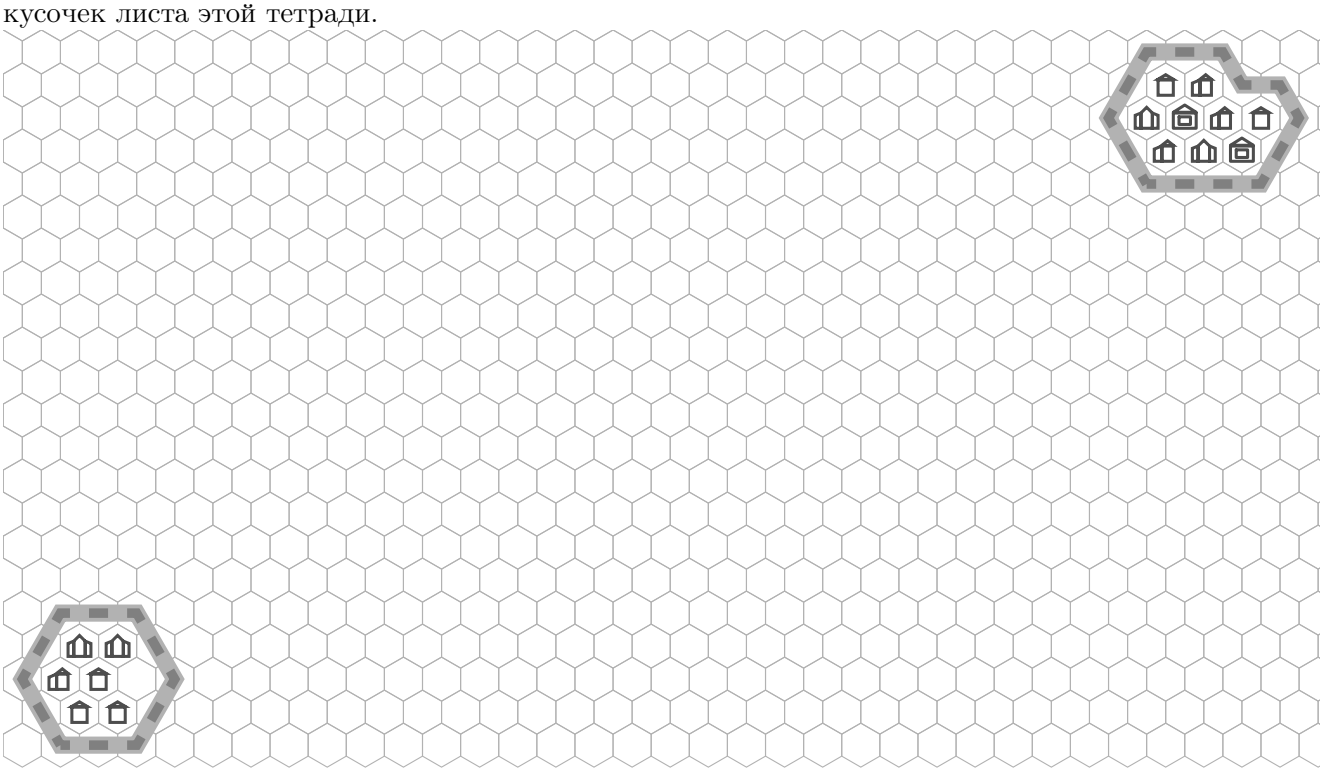
Задача A. Городская стена

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

У Пети есть замечательная тетрадь. На каждом листе этой тетради напечатаны не клетки и не линейки, а гексовая решётка — совсем как в его любимых компьютерных играх. Ниже показан кусочек листа этой тетради.



Петя хочет нарисовать в тетради город. В городе будет n домов, каждый дом будет занимать один гекс решётки, разные дома будут в разных гексах. Поскольку n в задаче может быть большим, далее будем считать лист тетради бесконечным во все стороны.

Чтобы город не захватили варвары, нужно построить вокруг него толстую стену. Формально стена будет занимать несколько гексов решётки. Длина стены d — это количество гексов в ней. Кроме того, если часовой будет обходить стену дозором, начав с какого-то гекса стены и двигаясь по ней вокруг города, каждый раз переходя на соседний по стороне гекс, то ровно через d переходов он должен посетить все гексы стены по одному разу и вернуться в исходный. Конечно, все дома города должны оказаться в той части листа, которая огорожена стеной.

Петя понимает, что постройка стены требует времени и ресурсов. Поэтому он хочет выбрать гексы с домами и со стеной так, чтобы длина стены d была как можно меньше. Чему равно минимально возможное значение d?

Формат входных данных

В первой строке записано целое число n — количество домов в городе ($1 \leqslant n \leqslant 10^9$).

Формат выходных данных

В первой строке выведите целое число d — минимально возможную длину стены.

Примеры

стандартный ввод	стандартный вывод
6	12
9	14

Задача B. Покраски домино

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мебибайт

Алиса хочет стать художницей. Для этого нужно много практики. Но ей нечего рисовать! Она попросила Боба помочь ей с этим, и он кое-что придумал.

Бобу нравится домино. Его доминошки покрашены: одна половина белая, другая чёрная. Он решил замостить прямоугольную доску $n \times m$ доминошками и дать её Алисе, чтобы она её нарисовала.

Рисунок будет выглядеть как прямоугольная решётка $n \times m$, в которой каждая клетка покрашена в чёрный или белый цвет, а как именно лежали доминошки — не будет нарисовано. Боб может замостить доску многими способами, так что Алиса сможет сделать много рисунков. Но иногда рисунки разных замощений выглядят одинаково.

Боб хочет, чтобы Алиса сделала как можно больше рисунков, но он не хочет, чтобы Алиса тратила своё время впустую, рисуя одно и то же. Помогите ему посчитать количество различных рисунков, которые Алиса может нарисовать. Так как ответ может быть очень большим, выведите его по модулю $10^9 + 7$.

Формат входных данных

Первая строка содержит целые числа n и m: размеры доски ($1 \leqslant n \leqslant 6$, $1 \leqslant m \leqslant 300$).

Формат выходных данных

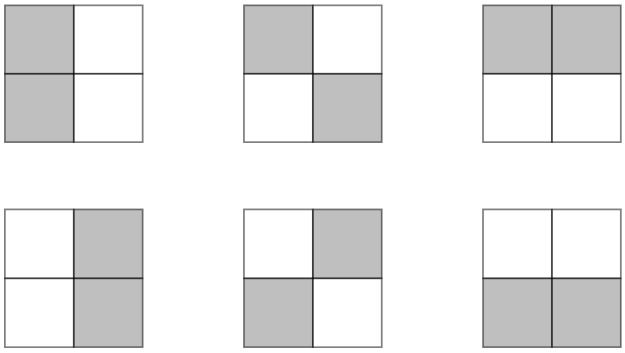
Выведите одно число: количество различных рисунков по модулю $10^9 + 7$.

Примеры

стандартный ввод	стандартный вывод
2 2	6
2 3	16
3 3	0

Пояснения к примерам

В первом примере шесть рисунков:



Заметьте, что второй рисунок может быть получен из двух замощений (границы доминошек выделены жирным):

Задача C. Counterquestion

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

В 1988 году в IX номере журнала «Квант» была опубликована следующая задача:

Определите числовое значение слова ТРАНСПОРТИРОВКА*, если одинаковые буквы заменить соответственно одинаковыми цифрами, разные — разными, причём так, чтобы были выполнены указанные на рисунке неравенства.*

(На рисунке к задаче изображена цепочка неравенств `Т>Р>А>Н<С<П<О<Р<Т>И>Р>О<В<К<А`.)

Эта задача была впоследствии многократно использована на различных турнирах и олимпиадах для младших школьников, а последние несколько лет она встречается на каком-нибудь из таких мероприятий ежегодно.

Сегодня вы работаете в жюри одной из аналогичных олимпиад. Вы тоже хотите предложить участникам эту задачу, но поскольку некоторые из них могли уже выучить ответ наизусть, жюри предстоит внести небольшие изменения в условие. А именно, необходимо заменить слово `ТРАНСПОРТИРОВКА` на какое-то другое слово, и расставить в новом слове знаки сравнения (неравенства или равенства, если рядом стоят одинаковые буквы). При этом слово выбирает другой член жюри, и вам известно, что она может выбрать из определённого словаря любое слово, в котором ровно десять различных букв. Более подробно о её словаре написано в формате ввода. Ваше же участие заключается в том, что вы расставляете в данном слове знаки, или сообщаете, что невозможно их расставить так, чтобы у задачи был единственный ответ.

Формат входных данных

В первой строке записано слово из словаря. Слово состоит из маленьких английских букв, гарантируется, что различных букв в нём ровно десять. Словарь, который используется в этой задаче — это свободно распространяемый словарь ENABLE для игр со словами на английском языке. Используемую версию словаря можно скачать здесь: `http://acm.math.spbu.ru/171217/words.unix.txt` с переводами строк для Unix или `http://acm.math.spbu.ru/171217/words.windows.txt` с переводами строк для Windows.

Формат выходных данных

В первой строке выведите цепочку сравнений, для которой существует единственный ответ (числовое значение данного слова). Цепочка должна состоять из букв данного слова в исходном порядке, разделённых знаками сравнения («>», «<», «=»). Все буквы и знаки должны быть разделены пробелами. Если существует несколько подходящих цепочек, выведите любую из них. Если для данного слова не существует ни одной такой цепочки, выведите «`Impossible`».

Примеры

стандартный ввод	стандартный вывод
`counterquestion`	`c > o > u < n > t < e < r < q < u > e > s > t > i < o > n`
`nostalgically`	`n > o > s > t > a > l < g < i < c < a > l = l > y`
`hyperparasitism`	`Impossible`

Пояснения к примерам

Рассмотрим приведённую в ответе на первый пример цепочку неравенств и решение исходной задачи для этой цепочки:

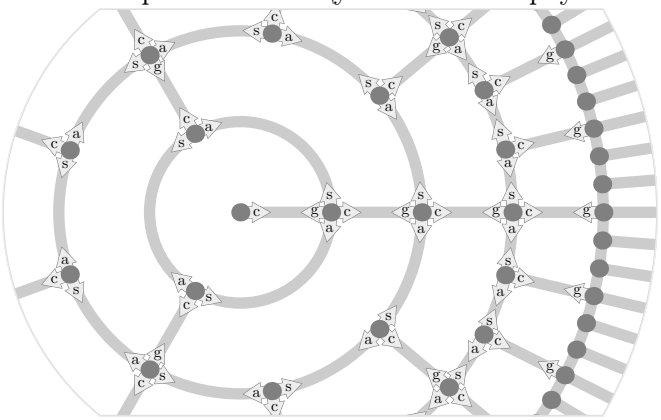
```
c > o > u < n > t < e < r < q < u > e > s > t > i < o > n
9 > 8 > 6 < 7 > 1 < 3 < 4 < 5 < 6 > 3 > 2 > 1 > 0 < 8 > 7
```

Знаки неравенств в решении оставлены для наглядности. Для слова «`counterquestion`» существуют и другие цепочки неравенств с единственным решением.

Задача D. Галактический Центр

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Эрнест и Селестина прибыли в Галактический Центр, чтобы зарегистрировать свой домик на краю Галактики. Центр — это огромная система залов и коридоров. Эрнест и Селестина очутились в центральном зале и посмотрели на висящую на стене карту ближайших окрестностей.



Залы Центра делятся на уровни: на уровне 0 лишь один зал — центральный, на уровне 1 три зала, на уровне 2 — девять залов, на третьем уровне целых 27 залов, и так далее. Галактический центр поистине огромен, так что количество уровней можно считать бесконечным. Залы каждого уровня, кроме нулевого, соединены коридорами в кольцо. Кроме того, каждый зал соединён коридором с залом следующего уровня. Соответственно, каждый третий зал соединяется с залом предыдущего уровня. Коридоры образуют регулярную структуру, как показано на карте выше. По всем коридорам можно перемещаться в обе стороны. Удивительно, но перемещение по любому коридору в любую сторону занимает одно и то же фиксированное время.

Маршруты перемещений в Центре записываются как строки из маленьких английских букв. Каждая буква означает движение по коридору до соседнего зала. Буквы соответствуют следующим направлениям на карте: «s» (спин) — против часовой стрелки, «a» (антиспин) — по часовой стрелке, «c» (центробежное) — от центра, «g» (гравитационное) — к центру. Адресом зала считается любой маршрут, начинающийся в центальном зале и ведущий в этот зал.

Эрнесту и Селестине нужно сначала поставить подпись в зале по адресу s, а затем — печать в зале по адресу t. Они уже справились с первой половиной этой задачи, и теперь находятся в зале по адресу s. Помогите им найти кратчайший маршрут в зал по адресу t. Длиной маршрута считается количество букв в нём.

Формат входных данных

В первой строке задан адрес s, а во второй — адрес t. Каждый из них содержит от 1 до 35 букв. Гарантируется, что оба адреса являются корректными маршрутами из центрального зала, а также что холлы по этим двум адресам различны.

Формат выходных данных

Выведите кратчайший маршрут из зала по адресу s в зал по адресу t. Если возможных ответов несколько, выведите любой из них.

Примеры

стандартный ввод	стандартный вывод
ccc csss	gg
cccsca cccacs	aaaa

Задача E. IBM 1403

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

IBM 1403 — принтер, выпущенный корпорацией International Business Machines в 1959-м году. Он устроен следующим образом: на вход подаётся бесконечная лента бумаги ровно в N символов шириной (к сожалению, принтер поддерживает только моноширинную печать). Собственно для печати используется циклическая цепь с L литерами. Изначально цепь расположена так, что литеры с номерами от 1 до N расположены как раз над первой строкой бумажной ленты.

Цепь равномерно вращается со скоростью 1000 литер в секунду, так что через одну миллисекунду над первой строкой будут литеры с номерами от 2 до $N + 1$, а через L миллисекунд цепь снова вернётся в исходное положение.

Для печати принтер использует два действия. Первое — поставить оттиски. В любой момент можно послать сигнал любому подмножеству литер, которые в этот момент над бумагой, и там отпечатаются соответствующие символы. Конечно же, это действие выполняется только на границе миллисекунд, никто не хочет получить символ в кривом месте. Второе действие — промотать бумагу на следующую строку. Второе действие принтер выполняет, когда работа с текущей строкой закончена (в обратную сторону бумагу прокрутить невозможно), и оно занимает одну миллисекунду. Помните, что во время перемотки цепь с литерами продолжает вращаться.

Конечно же, принтер выполняет эти действия в такой последовательности, чтобы напечатать требуемый текст за минимальное время. При этом в каждой позиции может быть отпечатан только один символ: например, нельзя напечатать «F», потом на том же месте подчёркивание и сделать вид, что получилась буква «E». Зато какие-то позиции можно оставить пустыми: пробел потому и называют *непечатаемым* символом, что печатать его не требуется.

Вася, младший лаборант КДПВ (Кафедры Длинных Программных Выводов), отправил на печать очередной отладочный лог своей программы для вычисления времени работы программ. Лог довольно большой и печататься будет долго, так что Вася решил пока что пойти поиграть в снежки с друзьями. Но ему интересно, когда можно будет возвращаться за результатом: помогите Васе определить, когда принтер напечатает весь текст лога. Гарантируется, что на цепи есть все литеры, необходимые для Васиного текста.

Напишите, пожалуйста, программу, которая по заданной конфигурации цепи и тексту, отправленному на печать, определит время, которое потратит принтер на печать этого текста (то есть до момента, когда принтер перемотает последнюю строку).

Формат входных данных

Первой строка содержит L символов и представляет из себя описание цепи принтера: символы на цепи в том порядке, в котором они на ней расположены, начиная с того, который расположен над первым символом строки в начале работы принтера. Следующие строки содержат ровно по N символов, в них записан текст, который требуется напечатать.

Ограничения:

- $1 \leqslant N \leqslant L \leqslant 10^5$,
- на цепи могут быть символы с кодами от 33 («!») до 126 («~»),
- в строках текста могут быть пробелы (символы с кодом 32) и символы, которые есть на цепи,
- размер текста не превосходит двух мебибайт (2 097 152 байт, включая переводы строк, идущие после каждой строки), при этом перевод строки для совместимости считается за два байта,
- в тексте могут встречаться несколько пробелов подряд, а также пробелы в началах и/или концах строк.

Формат выходных данных

Выведите одно число: время в миллисекундах, которое принтер потратит на печать данного текста.

Примеры

стандартный ввод	стандартный вывод
qwertyuiopasdfghjklzxcvbnm1234567890-! sankt-peterburg ␣-␣samyj␣tramva jizirovannyj␣go rod␣v␣mire!␣!␣!	148
abc acc cab cab	6
+*+*+*+*+*+* ␣␣␣␣␣␣␣␣␣␣␣ ␣␣*␣␣␣␣␣*␣␣ ␣␣␣*␣␣␣*␣␣␣ ␣␣*******␣␣ ␣**␣***␣**␣ ********** *␣*******␣* *␣*␣␣␣␣␣*␣* ␣␣␣**␣**␣␣␣	14

Замечание

Для наглядности пробел в примерах отмечен символом «␣».

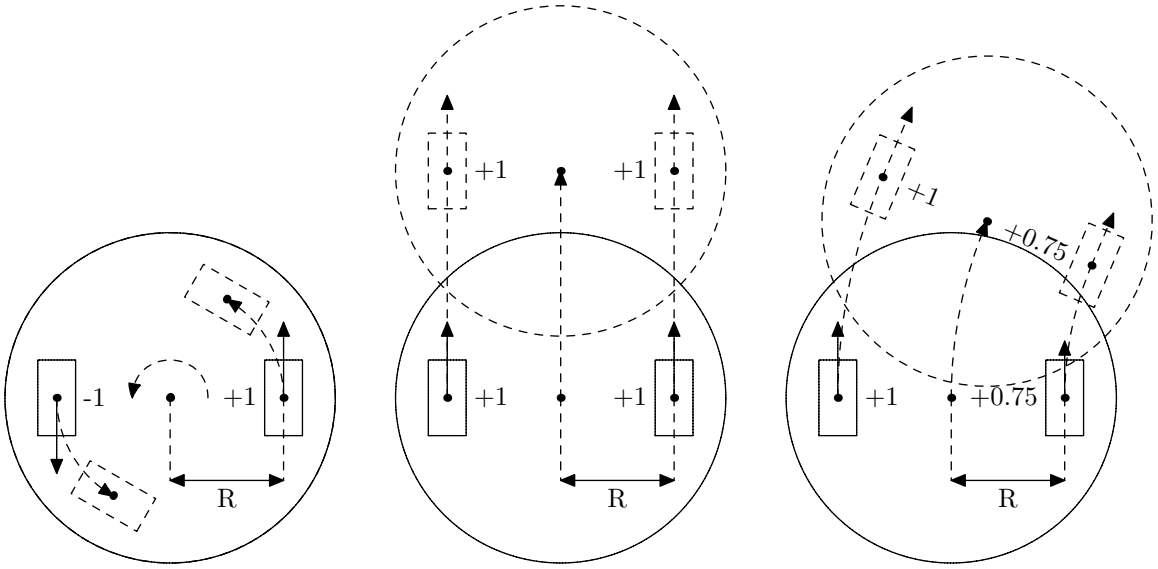
Задача F. Следование по линии

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Это интерактивная задача.

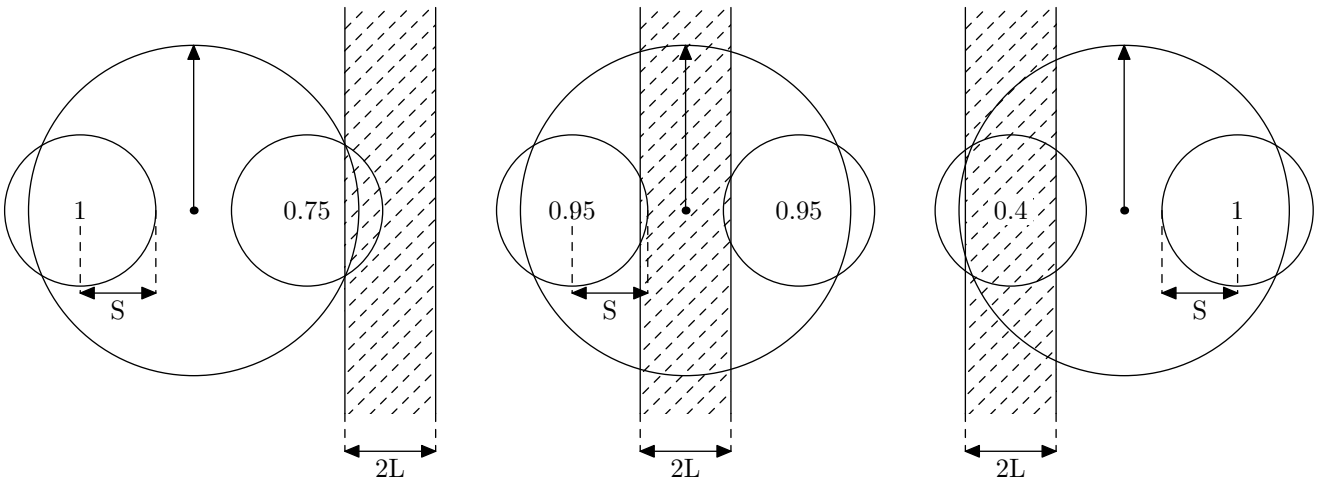
Типичный робот-пылесос может использовать *дифференциальный привод* для передвижения по плоскости. В этой схеме у робота имеется два колеса одинакового радиуса, расположенных фиксированном расстоянии $2R$ друг от друга. Каждое колесо может вращаться независимо от другого.

Мы считаем, что мотор может крутить каждое колесо с произвольной вещественной скоростью от -1 от $+1$, включительно. В зависимости от разности скоростей колёс и направления вращения робот будет либо ехать прямо, либо поворачиваться по дуге или на месте:



Сейчас робот находится на бесконечной плоскости белого цвета, на которой чёрной краской нарисована некоторая ломаная толщиной $2L$. Робот стоит в начале этой ломаной и должен доехать по ней до конечной точки.

У робота есть два датчика, каждый из которых закреплён в точности под колесом и позволяет примерно оценить долю чёрного в круге радиуса S вокруг датчика. Каждый датчик выдаёт одно вещественное число от 0 (весь круг чёрный) до 1 (весь круг белый). Например, 0.5 обозначает «примерно половина круга закрашена чёрным»:



Вам требуется написать программу для робота, которая позволит ему проследовать вдоль некоторой ломаной на плоскости.

Исходно центр робота находится над началом ломаной (её первой вершиной) под углом $\alpha + \beta$ (в радианах), где α — направление на вторую вершину ломаной, а β — равномерно распределённое случайное число от -0.2 до 0.2.

Дальнейшая эмуляция производится пошагово. На каждом шаге ваша программа должна считать показания датчиков робота и вывести скорости вращения колёс для перехода на следующий шаг. Показания датчика, находящегося на расстоянии d от ломаной, считаются равными

$$1 - \frac{\max(0, \min(S, d + L) - \max(-S, d - L))}{2S}.$$

После получения от вашей программы скоростей колёс c_l и c_r программа жюри:

1. Добавит к каждому значению среднее арифметическое K равномерно распределённых случайных чисел от -0.1 до 0.1.
2. c_l заменяется на $\min(\max(c_l, -1), 1)$, аналогично для c_r.
3. Робот T раз продвигается на $\frac{(c_l+c_r)\cdot V}{2T}$ единиц в направлении движения, после чего разворачивается на месте на $\frac{(c_r-c_l)\cdot V}{2RT}$ радиан.
4. Переходит к следующему шагу эмуляции.

Если в какой-то момент робот удаляется от центра ломаной дальше, чем на $R + S + L$, то ваше решение получает вердикт «неверный ответ». Если ваше решение не *прошло* ломаную за M шагов, то оно получает вердикт «неверный ответ». Ломаная из n вершин *считается пройденной*, если существуют такие номера шагов $t_1 < t_2 < \cdots < t_n$, что на шаге t_i робот был на расстоянии не более $R + S + L$ от вершины p_i ломаной.

Все тесты к этой задаче вы можете скачать здесь: `http://acm.math.spbu.ru/171217/linetracing.zip`. Каждый файл начинается с единственного числа — количество вершин ломаной, за которым следуют координаты вершин ломаной.

Ограничения

$R = 1.5$, $S = 1.0$, $L = 0.6$, $K = 10$, $V = 0.5$, $T = 100$, $M = 20\,000$, общая длина ломаной не превосходит 5000.

Протокол взаимодействия

В начале каждого шага вашей программе подаётся в отдельной строке два вещественных числа от 0 до 1 каждое — показания левого и правого датчика, соответственно. После этого вы должны вывести два вещественных числа от -1 до $+1$ каждое — с какой скоростью вращать колёса после текущего шага эмуляции.

Чтобы предотвратить буферизацию вывода, после каждой выведенной строчки следует вставить команду очистки буфера вывода: например, это может быть `fflush (stdout)` в C или C++, `System.out.flush ()` в Java, `flush (output)` в Pascal или `sys.stdout.flush ()` в Python. А ещё не забывайте выводить перевод строки в конце каждой выведенной строки.

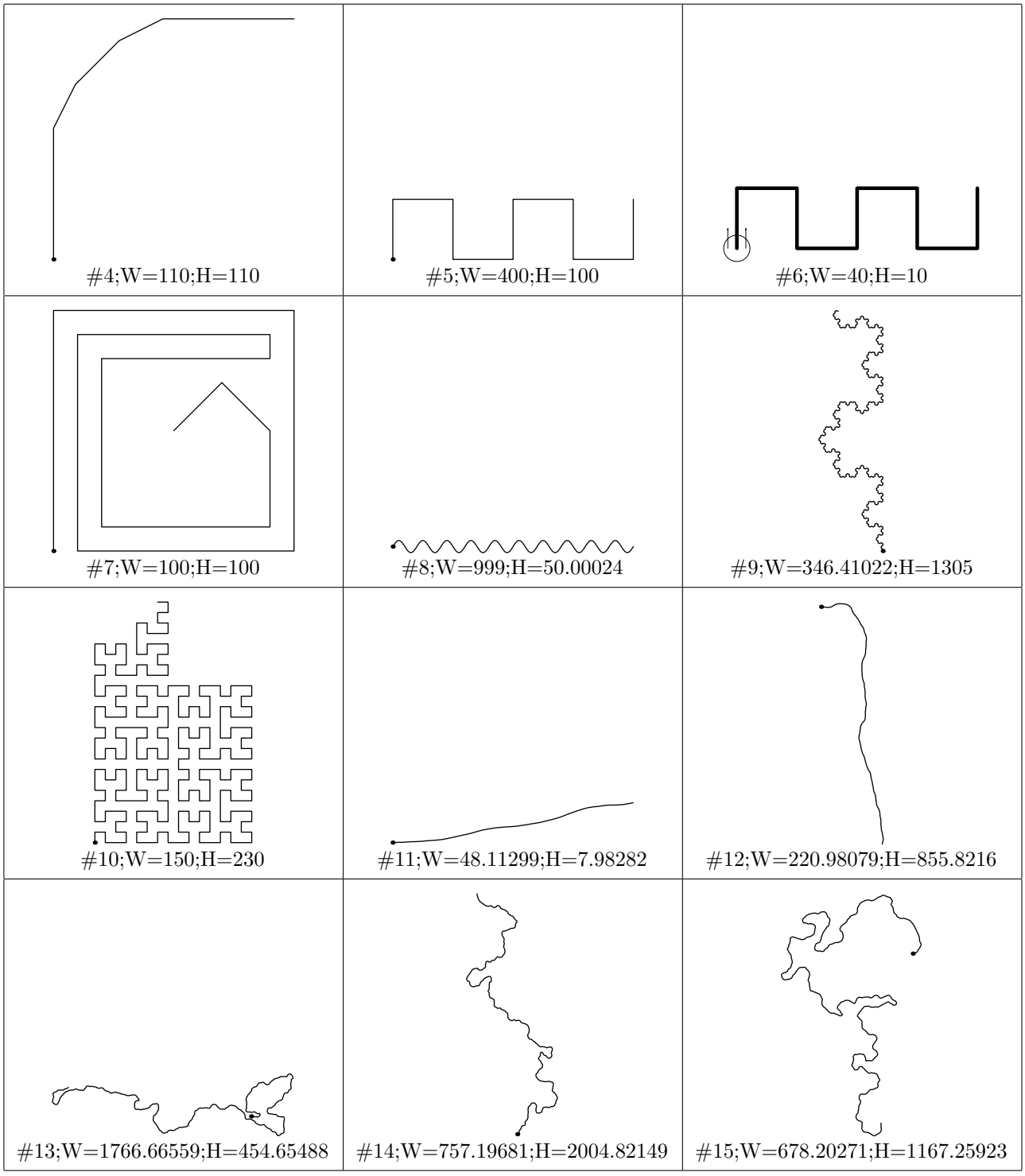
Если робот доехал до конца ломаной, ваша программа получит на вход "`-1 -1`" вместо очередных показаний датчиков, в этом случае она должна сразу корректно завершить работу.

Пример

стандартный ввод	стандартный вывод
0.96364308 0.94457910 0.99295765 0.95889118 0.99212102 0.90060966 ... -1 -1	 1 0.8 0.9 1 1 0.5 ... 1 1

Замечания

Тесты 1-3 представляют собой прямые линии длиной 10, 100 и 4000, соответственно. Остальные тесты представлены ниже.



Задача G. Ферзь и конь

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	3.5 секунд
Ограничение по памяти:	256 мебибайт

Это интерактивная задача.

Жили-были королева и рыцарь. Жили они счастливо, пока рыцарь не украл деньги из сокровищницы. Королева хочет теперь поймать Рыцаря и посадить в тюрьму.

Всё это происходит на шахматной доске $N \times N$, королева — это ферзь, а рыцарь — конь. Они ходят по стандартным шахматным правилам: ферзь на произвольное ненулевое число клеток по вертикали, горизонтали или диагонали, а конь — буквой «Г», меняя одну из своих координат на 1, а другую координату на 2.

Ваша программа будет играть за ферзя, а программа жюри — за коня.

Ваша задача — съесть коня за не более чем N ходов. Ферзь ходит первым.

Протокол взаимодействия

Чтобы пройти один тест, ваша программа должна выиграть в K сценариях ($1 \leqslant K \cdot N \leqslant 10\,000$). Все сценарии в одном тесте проходят на одной и той же доске размера $N \times N$.

Ввод начинается с одной строки, содержащей одно целое число N ($4 \leqslant N \leqslant 100$). Далее следуют сценарии.

Каждый сценарий начинается с задания начальных позиций ферзя и коня. Они даются на одной строке как четыре целых числа, разделённые пробелами: Qx, Qy, Kx и Ky, где $1 \leqslant Qx, Qy, Kx, Ky \leqslant N$. Гарантируется, что ферзь и конь начинают на различных клетках.

После этого вам необходимо делать ходы ферзём. Каждый ход делается выводом одной строки из двух целых чисел, разделённых пробелом: Qnx и Qny — новая позиция, выбранная для ферзя ($1 \leqslant Qnx, Qny \leqslant N$). Можно делать только разрешённые правилами ходы, в частности, нельзя оставаться на том же месте. Программа жюри отвечает одной строкой с двумя целыми числами: Knx и Kny — новой позицией коня ($1 \leqslant Knx, Kny \leqslant N$). Гарантируется, что конь делает только разрешённые правилами ходы.

Если одна из фигур становится на позицию другой, то вторая фигура объявляется съеденной, а оставшаяся выигрывает игру. Если вы съели коня (вы выиграли), или конь может съесть вас на своём ходу (вы проиграли), или вы уже сделали N ходов и игра не завершилась (вы проиграли), или вы сделали недопустимый ход (вы проиграли), интерактор отвечает двумя нулями вместо хода коня. В этом случае необходимо перейти к следующему сценарию.

В каждом тесте все сценарии зафиксированы заранее и не меняются по ходу соревнования. Конечно, тест считается пройденным только в том случае, если ваша программа выиграла все сценарии.

После последнего сценария в тесте, или же после первого проигранного сценария (смотря что встретится раньше), последует строка, содержащая четыре нуля вместо позиций фигур. Прочитав её, ваша программа должна сразу же корректно завершить работу.

Чтобы предотвратить буферизацию вывода, после каждого выведенного хода следует вставить команду очистки буфера вывода: например, это может быть `fflush (stdout)` в C или C++, `System.out.flush ()` в Java, `flush (output)` в Pascal или `sys.stdout.flush ()` в Python. А ещё не забывайте выводить перевод строки в конце каждой выведенной строки.

Пример

стандартный ввод	стандартный вывод
4 1 1 2 3 3 1 0 0 1 1 1 4 0 0 0 0 0 0	 2 2 3 1 3 3

Замечание

Заметьте, что в вышеприведённом примере решение проиграло во втором сценарии. Ваша программа может рассмотреть возможность взятия коня вместо того, чтобы подставлять ферзя под бой коню.

Задача H. Произведение корней

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	6 секунд
Ограничение по памяти:	256 мебибайт

Боб любит играть с многочленами. Вчера он придумал две последовательности неотрицательных целых чисел: $a_1, \ldots, a_n$ и $b_1, \ldots, b_m$, где $1 \leqslant n, m \leqslant 100\,000$.

После этого он построил многочлены

$$f(x) = \prod_{i=1}^{n}(1 + a_i x),$$

$$g(x) = \prod_{j=1}^{m}(1 + b_j x),$$

$$h(x) = \prod_{i=1}^{n}\prod_{j=1}^{m}(1 + a_i b_j x).$$

У многочлена $h(x)$ он посчитал только первые k коэффициентов ($1 \leqslant k \leqslant \min(100\,000, nm + 1)$). Поскольку Боб любит не только многочлены, но и преобразование Фурье, он посчитал коэффициенты всех трёх многочленов по модулю 998 244 353.

Однако сегодня Боб потерял и последовательности a_i и b_j, и многочлен $h(x)$. Остались только многочлены $f(x)$ и $g(x)$. Боб расстроился и теперь просит Алису помочь ему восстановить многочлен $h(x)$, зная только многочлены $f(x)$ и $g(x)$. У Алисы нет времени это делать, поэтому она попросила вас помочь ей.

Формат входных данных

В первой строке даны целые числа n, m и k ($1 \leqslant n, m, k \leqslant 100\,000$, $k \leqslant nm + 1$).

Во второй строке даны целые числа $f_0, \ldots, f_n$, где $f(x) = f_0 + f_1 x + \ldots + f_n x^n$.

В третьей строке даны целые числа $g_0, \ldots, g_m$, где $g(x) = g_0 + g_1 x + \ldots + g_m x^m$.

Коэффициенты многочленов даны по модулю 998 244 353.

Формат выходных данных

Выведите ровно k целых чисел $h_0, \ldots, h_{k-1}$, где $h(x) = h_0 + h_1 x + \ldots + h_{nm} x^{nm}$.

Выводить коэффициенты тоже нужно по модулю 998 244 353.

Пример

стандартный ввод	стандартный вывод
2 2 5 1 2 1 1 2 1	1 4 6 4 1

Задача I. Безопасное приземление

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Огромный человекоподобный боевой робот Георгий находится на плоском плато. Плато разбито на квадратные клетки, ориентированные по сторонам света: линии сетки идут с юга на север и с запада на восток. Георгий находится целиком в какой-то клетке. Будем считать плато бесконечным во все стороны.

Дипломат Марселла летает над плато на вертолёте. Она хочет приземлиться и вступить с Георгием в переговоры. Но приземление может быть небезопасным: Георгий может растоптать только что приземлившийся вертолёт до того, как переговоры вообще начнутся!

Георгий может перемещаться по плато: за одно перемещение он переходит из клетки в соседнюю по стороне. К счастью, после недавнего боя у Георгия повреждена система навигации: он может перемещаться только в три из четырёх сторон света, а в четвёртую — нет. К сожалению, неизвестно, в какую именно сторону робот не может перемещаться.

Марселла наблюдает за Георгием уже некоторое время и записывает все перемещения робота. Конечно, среди этих перемещений не могут встречаться все четыре стороны света. Сейчас Марселла находится точно над Георгием. Помогите ей выбрать для приземления безопасную клетку плато — такую клетку, в которой Георгия сейчас нет и в которой он точно не сможет оказаться — или выяснить, что наблюдения не позволяют указать такую клетку с определённостью.

Формат входных данных

В первой строке записано несколько символов без пробелов — маршрут Георгия за время наблюдения. Каждый символ — это заглавная буква английского алфавита, обозначающая сторону света, в которую переместился робот: «N» (север), «W» (запад), «S» (юг) или «E» (восток). Гарантируется, что в строке встречается не более трёх различных сторон света. Строка заканчивается переводом строки, её размер — от 1 до 100 символов.

Формат выходных данных

Если существует клетка, где Георгия сейчас нет и где он точно не сможет оказаться, выведите маршрут для вертолёта Марселлы, который начинается от клетки, где сейчас находится Георгий, и заканчивается в такой безопасной клетке. Маршрут следует выводить в том же формате, что и маршрут во входных данных, в нём должно быть от 1 до 100 символов. Если таких возможных маршрутов несколько, выведите любой из них. Если же никакую клетку плато нельзя считать безопасной с определённостью, выведите вместо маршрута одну заглавную букву «X» (икс). После ответа можно вывести перевод строки.

Примеры

стандартный ввод	стандартный вывод
NNWNEE	SES
S	X

Пояснения к примерам

В первом примере Георгий за время наблюдений перемещался на север, на запад и на восток. Значит, он точно не может перемещаться на юг. Безопасна любая клетка южнее, чем нынешнее положение Георгия. Например, можно, как показано в примере, переместиться от робота на юг, на восток и ещё раз на юг.

Во втором примере известно только, что Георгий умеет перемещаться на юг. К сожалению, остаётся три варианта того, куда он не умеет перемещаться, и из-за этого ни одна клетка не является безопасной. Например, если Марселла выберет маршрут «N», может оказаться, что Георгий не умеет перемещаться на восток, а на север — умеет. Если же выбрать маршрут «NEE», может оказаться, что Георгий умеет перемещаться и на север, и на восток, а не умеет — на запад, и так далее.

Задача J. Точный квадрат

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Целое число n называется точным квадратом, если существует целое число x, для которого верно равенство $x \cdot x = n$.

Задано целое положительное число n. Определите, является ли оно точным квадратом. Имейте в виду, что число n может быть очень большим!

Формат входных данных

В первой строке записано целое положительное число n в десятичной записи. Гарантируется, что в этой записи от 1 до 1 000 000 цифр, а первая цифра положительна.

Формат выходных данных

Выведите «`Yes`», если число n является точным квадратом, и «`No`» в противном случае. Регистр букв не имеет значения — например, можно вывести «`yES`» или «`NO`».

Примеры

стандартный ввод	стандартный вывод
25	Yes
27	No