Для всех задач:

Имя входного файла: `input.txt`
Имя выходного файла: `output.txt`
Ограничение по памяти: *64 Мб*

№	Задача	Ограничения по времени на 1 тест в секундах
1	Подвешивание деревьев	6
2	Интегральные схемы	2
3	Спирали сюжета	2
4	Зависшая программа	2
5	Первый блин...	2
6	Конструктивная блочная геометрия	2
7	Кривая Коха	2
8	Сравнение картинок (только для высшей лиги)	6
9	Отец Фёдор и капуста	2
10	Бег с препятствиями	6
11	Морской бой (только для высшей лиги)	3
12	Преобразование (только для первой лиги)	2
13	Размер вознаграждения (только для первой лиги)	2

Задача 1. Подвеши вание деревьев

С утра заведующего ВЦ НИИЧАВО Александра Привалова вызвал к себе Янус Полуэктович. Речь шла об одной из разработок отдела абсолютного знания — машине времени для передвижения во временных пространствах, сконструированных искусственно, в испытаниях которой, как известно, участвовал Привалов. Напомним, что сконструированная Луи Седловым машина позволяла путешествовать по времени в созданных воображением пространствах. Литературоведы и писатели оценили перспективы, открывающиеся при использовании этой новинки, и одно из крупнейших в мире издательств сделало заказ на разработку серийного экземпляра машины. А, следовательно, ВЦ НИИЧАВО предстояло заняться моделированием её узлов.

Одним из основных узлов машины должен стать блок моделирования внутреннего времени, которое в машине представляется в виде дерева. Учитывая, что серийный экземпляр должен позволять одновременную обработку большого количества разных текстов, информацию придётся сжимать. Одна из подзадач, возникающих при этом, формулируется следующим образом. Даны два произвольных дерева. Необходимо узнать, можно ли так подвесить оба дерева и удалить некоторые вершины из первого дерева, чтобы после этого деревья стали изоморфными. Заметим, что при удалении вершины автоматически удаляется все поддерево.

В случае положительного ответа на поставленный вопрос требуется найти минимальное число удалений, которое потребуется сделать для этого, и указать вершины, за которые будут подвешены деревья.

Входные данные

В первой строке входного файла записано через пробел два числа n и m ($1 \le n, m \le 100$), где n — количество вершин первого дерева, а m — количество вершин второго дерева.

Далее описываются деревья, сначала — первое, затем — второе. Каждое дерево задается в следующем формате. Все вершины дерева пронумерованы числами от 1 до количества вершин. Каждое дерево подвешено за первую вершину. Для каждой вершины по порядку, кроме первой, задается вершина, которая является ее непосредственным предком в подвешенном дереве. Степень каждой вершины в дереве не превосходит 21.

Выходные данные

В выходной файл необходимо выдать три целых числа, записанных через пробел: количество вершин (корней поддеревьев), которые нужно удалить, номер вершины первого дерева, номер вершины второго дерева, за которые надо подвешивать, чтобы получить данный ответ. Если существует несколько вариантов, то вывести первый в лексикографическом порядке. Если решения не существует, то вывести `-1`.

Примеры

input.txt	*output.txt*
2 2 1	0 1 1

1	
3 2 1 1 1	1 1 1

Замечание

Будем считать два дерева изоморфными, если между их вершинами можно установить взаимно однозначное соответствие, сохраняющее отношение инцидентности.

Задача 2. Интегральные схемы

В качестве элементной базы для электронных узлов проектируемой машины планируется использовать интегральные схемы (ИС). Топология ИС часто собирается из библиотечных элементов путем попарной сборки. Пусть операция + соответствует размещению одного блока над другим, а операция * — расположению одного блока справа от другого.

Тогда план сборки ИС можно описать при помощи обратной польской записи. Обратная польская или постфиксная запись — это такая запись выражения, в котором знак операции располагается после операндов. Например, план сборки `B1B2+B3*` показан на рисунке. Каждый библиотечный элемент может иметь несколько вариантов реализаций, отличающихся размерами. Каждая реализация описывается парой (h, w), где h — это высота блока, а w — его ширина.

При объединении двух библиотечных элементов, они располагаются встык друг к другу, а размер нового блока определяется по правилу:

$w(B_iB_j+) = \max(w_i, w_j)$
$h(B_i,B_j+) = h_i + h_j$
$w(B_iB_j*) = w_i + w_j$
$h(B_iB_j*) = \max(h_i, h_j)$

Надо выбрать по одной реализации для каждого вхождения блока в план сборки, чтобы результирующий блок имел минимальную площадь.

Рассмотрим пример с тремя блоками. Блок B1 имеет одну реализацию (2, 2), блок B2 — две реализации (2,3) и (3, 2), а блок B3 имеет три реализации (1, 4) , (4, 1), и (2, 5). При вертикальной сборке блоков B1 и B2 с выбранной первой реализацией ширина результирующего блока стала равной 3, а высота — 4. Если взять также первую реализацию блока B3, то получится новый блок с шириной 7 и высотой 4, как показано на рисунке. Таким образом, площадь полученного блока будет равна 28. Однако, если бы мы взяли вторые реализации блоков B2 и B3, то получили бы блок с площадью 15.

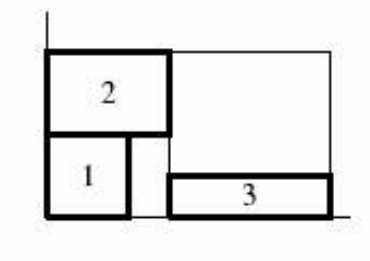


Рис. План сборки B1B2+B3* для блоков B1(2, 2), B2(2,3), B3(1,4).

Входные данные

В первой строке входного файла записано число N – количество библиотечных блоков ($1 \le N \le 100$) . Блоки пронумерованы числами от 1 до N. В следующих N строках даны реализации каждого блока по порядку номеров. В строке, описывающей реализации одного блока, сначала идет целое число — количество реализаций этого блока r_i ($1 \le r_i \le 1000$, $1 \le i \le N$), затем через пробел записываются r_i пар целых положительных чисел, не превосходящих 100000, задающих высоту и

ширину блока для каждой реализации. В последней строке дан план сборки интегральной схемы — это выражение в обратной польской записи, в котором операндами являются слова вида BK, где K — номер блока, а операциями — + и *. Выражение содержит не менее одного операнда. Количество операндов не превосходит 100.

Выходные данные

В выходной файл необходимо выдать одно число — минимальную площадь, которую может иметь собранная интегральная схема.

Пример

input.txt	*output.txt*
3 1 2 2 2 2 3 3 2 3 1 4 4 1 2 5 B1B2+B3*	15

Задача 3. Спирали сюжета

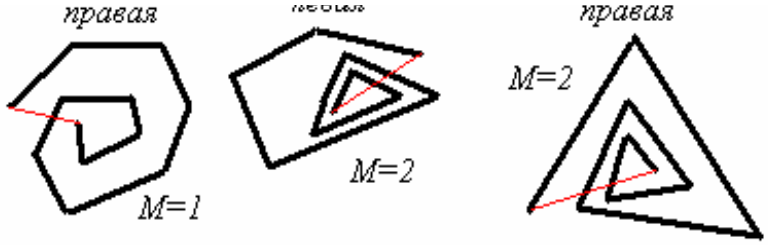
В отличие от опытного экземпляра, серийный экземпляр машины должен позволять путешествие в «продолжение» искусственно созданной реальности, то есть, по сути дела, автоматическое продолжение написанного автором текста. Эта задача была решена Кристобалем Хунтой, проверявшим известный эксперимент с обезьянами и продолжением «Войны и мира».

При генерации продолжений ряда текстов (особенно это касается так называемых остросюжетных произведений) возникает понятие «закрученность» сюжета. При анализе сюжетов система представляет их в виде спиралей, каждая из которых является закрученной ломаной. Первые эксперименты позволили выяснить, что сюжеты, которые воспринимаются читателями как интересные, представляются в виде «правой» (то есть закрученной по часовой стрелке) спирали.

Назовём спираль сюжета «красивой спиралью уровня M», если отрезок, проведённый из последней точки спирали в ее начальную точку, пересекает не менее M отрезков, образующих спираль. Число таких пересечений можно назвать «закрученностью» спирали. Считается, что отрезок, ведущий из начала в конец спирали не пересекается с первым и последним отрезками, образующими спираль.

Задача продолжения сюжета, согласно модели Хунты, сводится к следующей. На вход задаются N точек — события, которые должны произойти в продолжении. Все точки находятся в верхней полуплоскости ($0 < Y \le 1000$).

Необходимо построить спираль сюжета, начинающуюся в начале координат, закрученную вправо и являющуюся красивой спиралью заданного уровня M. После того, как спираль построена, генерируется продолжение сюжета, начиная с точки с номером S в этой спирали. Продолжение состоит из K звеньев спирали.



Красивые спирали

Входные данные

В первой строке входного файла содержатся четыре целых числа N, M, S, K, где N – количество точек, M – минимальное число закрученности для красивой спирали, S – номер начальной точки для вывода последовательности фрагмента спирали, K – количество подлежащих выводу последовательных точек спирали ($0 \le S \le N$, $0 < N \le 1000$). Последующие строки исходного файла содержат N пар (X, Y) целых чисел – координаты точек, каждая из которых должна быть точкой излома спирали ($-1000 \le X \le 1000$, $0 < Y \le 1000$).

Точка (0, 0) имеет в спирали номер, равный 0. Исходные точки нумеруются с 1 в порядке ввода. Гарантируется, что среди точек, заданных в файле, точка с координатами (0, 0) отсутствует. Все точки во входном файле различны и нет трех точек, лежащих на одной прямой, включая точку с координатами (0, 0).

Выходные данные

Если по заданными точкам можно построить красивую спираль уровня *M*, то в первую строку выходного файла необходимо вывести единственное слово `Beautiful`, **а** во вторую строку нужно вывести *K* номеров точек этой спирали, начиная с точки с номером *S*.

Если же красивую спираль построить не удается, то нужно выдать слово `Ugly`.

Примеры

input.txt	*output.txt*
4 0 4 1 6 3 5 3 5 5 3 4	Beautiful 2
4 3 2 0 6 3 5 3 5 5 3 4	Ugly

Задача 4. Зависшая программа

Для расчёта одного из узлов машины Привалов написал многопоточную программу. Но в самый ответственный момент случилось зависание. Оказалось, что корень всех зол — дедлоки (deadlock). Дедлок – это такая ситуация, когда потоки «ожидают» друг друга. Программа Привалова запустила несколько потоков, каждый из которых имеет несколько ресурсов. Каждый поток может затребовать ресурсы какого-то потока. Что и произошло. С помощью отладчика Привалов узнал о каждом из потоков, ресурсами какого потока тот пытается овладеть. Причем, оказалось, что каждый поток ожидает ресурсы не более, чем одного потока. Если отключить какой-либо поток, то он освободит все свои ресурсы.

Привалов хочет отключить минимальное количество потоков так, чтобы разрешить все циклические ожидания. Все выбранные для отключения потоки отключаются одновременно. Если существует несколько вариантов, то требуется отключить те потоки, для которых суммарное количество ожидающих их потоков минимально.

Входные данные

В первой строке входного файла записано через пробел два целых числа N и M.

N – количество потоков в программе, M – количество зависимостей между потоками ($1 \le N \le 10000$, $0 \le M \le N$).

Далее следует M строк, на каждой из них даны числа $1 \le A_i, B_i \le N$, означающие, что поток A_i не может захватить ресурсы потока B_i. Все A_i различны.

Выходные данные

В первую строку выходного файла вывести одно целое число — количество отключенных потоков, а во вторую через пробел в произвольном порядке выдать номера отключенных потоков. Если существует несколько вариантов отключения, то вывести любой.

Пример

input.txt	*output.txt*
6 5 2 5 1 3 4 1 3 1 6 6	2 3 6

Задача 5. Первый блин...

На первом стендовом испытании основных узлов машины в качестве «входного» текста была использована «Алиса в Зазеркалье». К сожалению, программное обеспечение было ещё «сырым», и испытатель-доброволец вместо пешки на обычной шахматной доске оказался шахматным слоном на полуисчезнувшей шахматной доске большего размера. Возможно, что слово «Chessboard» было ошибочно воспринято как «Cheshire board» — со всеми вытекающими отсюда последствиями.

Слону необходимо добраться до определённой точки доски, пока доска окончательно не исчезла. Шахматный слон двигается только по диагонали. Каждая остановка для смены направления движения занимает много времени, поэтому для спасения испытателя Вам требуется найти минимальное возможное количество остановок на пути слона к цели.

Входные данные

В первой строке входного файла задано одно целое число N — количество оставшихся на ней клеток, ($2 \le N \le 10^5$)

Далее следуют N строк вида A_i, B_i с целыми координатами оставшихся клеток ($1 \le A_i, B_i \le 10^9$).

Первая заданная клетка — начало движения слона, вторая заданная клетка — цель. Начальная и целевая клетки различны.

Выходные данные

В выходной файл необходимо вывести единственное целое число — минимальное количество смен направления движения. Если цель недостижима, то вывести `-1`.

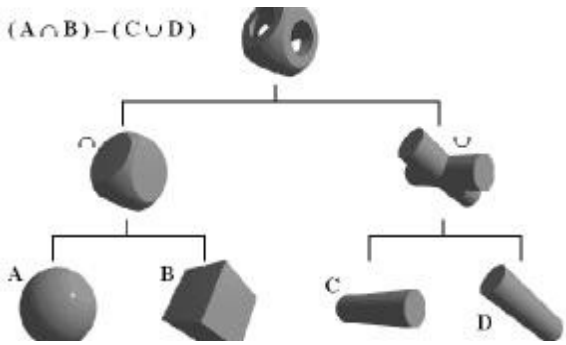
Пример

input.txt	*output.txt*
7 1 1 2 4 2 2 3 2 3 3 3 1 3 4	1

Задача 6. Конструктивная блочная геометрия

После стендового испытания было принято решение на данном этапе ограничиться компьютерным моделированием — до исправления ошибки в программном обеспечении.

Для моделирования пространственных объектов была использована конструктивная блочная геометрия (constructive solid geometry, CSG) — техника построения сложных геометрических объектов путем комбинирования примитивов при помощи логических операций. Построенные объекты называют CSG-моделями.



В качестве примитивов могут выступать шар, прямоугольный параллелепипед, цилиндр, тор и другие фигуры. Над объектами обычно определяются три операции: объединение (union), пересечение (intersection) и вычитание (difference).

Была выбрана упрощенная модель, в которой имеется только три примитива, все операции выполняются последовательно и имеют одинаковый приоритет. Скобки в описании модели отсутствуют.

Вам задан набор примитивов и последовательность операций над ними. Также заданы несколько точек в пространстве. Для проверки точности позиционирования материализованных объектов требуется определить принадлежность заданных точек полученной CSG-модели.

Входные данные

В первой строке входного файла находится единственное число N ($1 \le N \le 1000$) — количество точек. Каждая из следующих N строк описывает одну точку и содержит три числа: x y z – соответствующие координаты.

Далее указывается число K ($1 \le K \le 1000$) — количество операций над примитивами. Следующие строки содержат информацию о построении CSG-объекта виде:

```
<описание примитива>
<операция>
<описание примитива>
<операция>
<описание примитива>
…
```

Описание примитива выглядит следующим образом:

Примитив	***Описание***
Шар радиуса `r` с центром в точке (`x, y, z`).	`sphere` `x y z r`

Прямоугольный параллелепипед с рёбрами, параллельными координатным осям, и с двумя диагонально-противолежащими точками (`x1, y1, z1`) и (`x2, y2, z2`).	`cuboid`
	`x1 y1 z1`
	`x2 y2 z2`

Тор с осью вращения, параллельной оси z, с центром в точке (`x, y, z`) с внутренним и внешним радиусами `r1` и `r2` соответственно.	`torus`
	`x y z`
	`r1 r2`

Могут быть использованы следующие операции:

Операция	***Описание***
Объединение двух объектов.	`union`
Пересечение двух объектов.	`intersection`
Вычитание второго объекта из первого.	`difference`

Все координаты целочисленные и по модулю не превосходят 10^4.

Выходные данные

Для каждой из *N* точек (в порядке следования во входном файле) необходимо в выходной файл в отдельной строке вывести слово `in`, если точка принадлежит итоговому CSG-объекту, в противном случае вывести `out`.

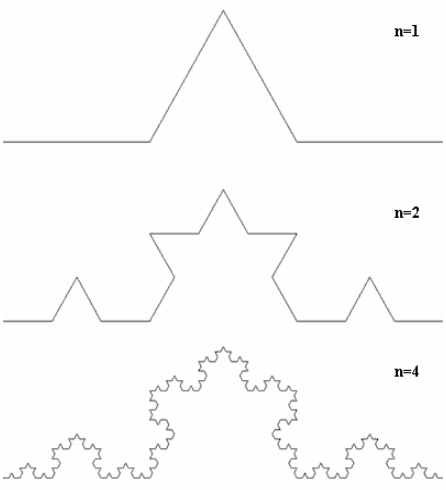
Пример

input.txt	*output.txt*
3 1 0 2 0 0 0 -4 0 0 3 sphere 0 0 0 3 intersection cuboid -2 -2 -2 2 2 2 union torus 0 0 0 3 5 difference sphere 0 0 0 1	in out in

Задача 7. Кривая Коха

Хотя проект выполнялся силами отдела абсолютного знания и ВЦ института, информация о том, что «наконец-то абсолютники сделали что-то стоящее» распространилась по всему НИИЧАВО. Так что ничего удивительного в том, что Кристобаль Хозевич Хунта однажды появился в ВЦ и предложил протестировать текущую версию программного обеспечения, не было.

Естественно, что Хунта сразу попытался дать неразрешимую, по его мнению, задачу — в качестве базового текста была задана «Сказка про белого бычка». Эксперимент оказался успешным — машина зависла.



Оказалось, что при моделировании такого рода «рекурсивных» текстов возникают фрактальные конструкции. В данном конкретном случае адекватной моделью оказалась кривая, которая строится следующим образом. Берём ориентированный единичный отрезок, разделяем на три равные части и заменяем средний интервал равносторонним треугольником без этого сегмента. Новая вершина при этом находится слева от отрезка. В результате образуется ломаная, состоящая из четырех ориентированных звеньев длины 1/3. На следующем шаге повторяем операцию для каждого из четырёх получившихся звеньев и так далее. Кривая, получаемая в пределе, называется кривой Коха.

Введем декартову систему координат так, что первоначальный отрезок имеет координаты начала и конца (–0.5, 0) и (0.5, 0) соответственно, а максимальная координата по оси ординат среди точек кривой Коха равна 1.

Для определения, какие из событий в «рекурсивном» тексте происходят одновременно, требуется определить количество точек пересечения заданной горизонтальной прямой с кривой Коха. Решить эту задачу при разработке новой версии программного обеспечения поручено Вам.

Входные данные

В первой строке входного файла даны два целых числа A и B ($-10^6 \le A \le 10^6$, $1 \le B \le 10^6$) — числитель и знаменатель дроби, задающей координату пересечения горизонтальной прямой с осью ординат.

Выходные данные

В первую строку выходного файла вывести единственное число – количество точек пересечения. Если таких точек бесконечно много, то вывести слово `INF`. Гарантируется, что конечный результат не превышает 10^9.

Примеры

input.txt	*output.txt*
1 3	4

0 1	INF

Задача 8. Сравнение картинок

Согласно основной идее, положенной в основу проекта машины, возможность объединения одновременных событий из «искусственно созданных реальностей», порождаемых различными текстами, в одной пространственной сцене зависит от того, насколько различается описание обстановки, в которой происходит действие.

Существующий уровень маготехники позволяет представить обстановку в виде чёрно-белого растрового изображения Q, состоящего из $N(Q)\times M(Q)$ пикселей. На изображение наложена система координат: ось X направлена вниз, а ось Y — вправо.

Каждый пиксель изображения Q имеет целые координаты (x,y) $(0 \le x < \mathrm{N(Q)}, 0 \le y < \mathrm{M(Q)})$. Для каждого пикселя изображения Q с координатами (i, j) задана яркость пикселя $Q[i, j]$ — целое число от 0 до 255 включительно.

Сравнение картинок происходит следующим образом. Пусть дано два изображения: A и B. Будем говорить, что прямоугольный фрагмент изображения A с началом в точке (s, t) (координаты левого верхнего угла фрагмента) отличается от изображения B на величину $D[s, t]$:

$$D[s,t] = \sum_{\substack{i=0..N(B)-1 \\ j=0..M(B)-1}} (A[s+i,t+j]-B[i,j])^2$$

При этом s и t меняются в пределах: $0 \le s \le N(A) - N(B)$, $\quad 0 \le t \le M(A) - M(B)$.

Для решения задачи нужно найти прямоугольный фрагмент A, имеющий наименьшее значение величины D[s, t] для изображения B. Если существует несколько таких фрагментов, то нужно выбрать фрагмент с минимальной координатой s. Если по-прежнему таких фрагментов несколько, нужно выбрать фрагмент с минимальной координатой t.

Входные данные

Во входном файле сначала описано изображение A, затем изображение B.

Для каждого изображения Q в первой строке через пробел записаны размеры: целые числа $N(Q)$ и $M(Q)$ $(1 \le N(A), M(A) \le 500, 1 \le N(B) \le N(A), 1 \le M(B) \le M(A))$.

В $N(Q)$ следующих строках записано по $M(Q)$ целых чисел через пробел. В i-ой строке j-ое число равно $Q[i, j]$ — яркость пикселя с координатами (i, j).

Выходные данные

В выходной файл нужно выдать координаты (`s`, `t`) начала искомого фрагмента и величину `D[s,t]` в формате `s t: D[s,t].` Формат вывода представлен в примерах.

Примеры

input.txt	*output.txt*	*Комментарии*
5 4 208 214 49 14 29 175 170 208 140 133 0 140 100 20 76 255 64 29 153 31 3 2 16 94 172 98 17 8	1 0: 16012	D[0,0] = 108396 D[0,1] = 59942 D[0,2] = 37306 D[1,0] = 16012 D[1,1] = 46815 D[1,2] = 132550 D[2,0] = 30815 D[2,1] = 67282 D[2,2] = 55262
4 4 0 1 2 0 1 0 0 1 0 1 1 1 1 1 1 1 1 2 1 1	2 1: 0	

Задача 9. Отец Фёдор и капуста

На первом испытании машины заказчики пожелали узнать дальнейшую судьбу отца Фёдора из «Двенадцати стульев» после выписки из лечебницы.

Оказывается, что отец Федор после погони за стульями вернулся к прежнему своему делу и снова занялся разведением кроликов. Однако, памятуя печальный свой опыт, теперь капусту он рубит сам. Правда делает он это несколько странно. Но что уж тут удивительного, все-таки многочисленные нервные потрясения дают о себе знать. Итак, действует он следующим образом: выберет кочан капусты правильной шарообразной формы, установит его, вспомнит первый свой стул, как он его рубил топором и давай кромсать капусту, разрезая кочан параллельными сечениями. Потом немного остынет, повернет кочан произвольным образом в пространстве, и снова режет его параллельными разрезами. Причем он наловчился так ловко разрезать кочан, что тот не распадается на куски. Количество параллельных разрезов у него за каждый подход одинаково. Количество подходов у него не меньше 1.

Похоже, машина работала. Однако неожиданно председатель комиссии и затесавшийся в неё в привычной роли научного консультанта профессор Выбегалло начали спорить, на какое максимальное количество частей может быть таким образом разрезан идеальный сферический кочан капусты в вакууме. Переключить их внимание на текущие задачи оказалось невозможным. Так что ответ надо вычислить как можно быстрее — иначе работа комиссии может затянуться.

Ваша задача — определить, на какое максимальное количество частей отец Фёдор может разрезать кочан за некоторое количество подходов.

Входные данные

Во входном файле в первой строке записано одно целое число L ($1 \le L \le 10000$) – количество кочанов, которые отец Федор порубил для кроликов.

В следующих L строках записано по два целых числа — K_i и N_i, где K_i — соответственно количество параллельных сечений для одного кочана капусты за один подход, а N_i — количество подходов к нему ($1 \le K_i \le 100$, $1 \le N_i \le 10000$).

Выходные данные

Выходной файл должен состоять из L строк. В i-ю нужно выдать одно целое число — максимальное количество частей, на которые отец Фёдор может разрубить сферический кочан капусты в i-м случае. Кочаны нумеруются в порядке их следования во входном файле.

Пример

input.txt	*output.txt*
2 2 3 3 1	27 4

Задача 10. Бег с препятствиями

Последнее из тестовых заданий при испытаниях председатель комиссии снова предложил по «12 стульям». Требовалось восстановить эпизод с погоней мадам Грицацуевой...

Мадам Грицацуева стремительным домкратом мчится по узкому редакционному коридору. Коридор в некоторых местах перекрывается дверями, наглухо закрывающими проход. Бедная вдова так встревожена за судьбу своего любимого суслика, что несется по коридору, сметая все на своем пути. Единственное, что способно остановить ее — вовремя закрытая дверь или конец коридора. В этом случае она стремительно разворачивается и бежит обратно. Ровно в двенадцать часов ночи она засыпает, утомленная всей этой суетой.

Задание было сформулировано так: определить точку, в которой заснёт вдова. Как известно, у Ильфа и Петрова никакого точного определения этой точки не было, так что задача довольно сложная. Для контроля работы машины Вам также предлагается решить подобную задачу.

Входные данные

Во входном файле в первой строке записаны через пробел три целых числа L, N и M, где L — длина коридора, N — количество дверей в нем и M — начальное местоположение мадам Грицацуевой($1 \le L \le 1000$, $0 \le N \le L-1$, $0 \le M \le L$).

Во второй строке сначала идет латинская буква L или R — если первоначальное направление ее движения влево или вправо соответственно, а затем время T ($1 \le T \le 20000$) в секундах, оставшееся до полуночи. Сам коридор, ограниченный стенами с обеих сторон, мы можем считать горизонтальным отрезком длины L с концами в точках 0 и L. Расстояния в нем отсчитываются от крайней левой точки. Соответственно двери можно считать точками на этом отрезке. Через любую дверь мы либо можем пройти, если она открыта, либо нет — в противном случае.

В последующих N строках идет описание каждой двери. Оно задается последовательностью целых чисел, записанных через пробел. Первое число в этой последовательности a_i — расстояние от крайней левой точки коридора до двери ($1 \le a_i \le L-1$). При этом ни одно из a_i не совпадает с M. Второе число — 0 (если дверь закрыта) или 1 (если дверь открыта) — указывает на состояние двери в начальный момент времени $T = 0$. Далее идет целое число p_i ($0 \le p_i \le 20$), показывающее, сколько раз эта дверь будет менять свое состояние на противоположное. Затем идут p_i строго возрастающих целых чисел t_k — моменты времени, когда дверь меняет свое положение ($1 \le t_k \le$ T). При этом в момент времени t_k дверь считается уже сменившей свое положение, а в любой строго меньший — еще нет. Все расстояния измеряются в метрах, время — в секундах. Скорость мадам Грицацуевой постоянна и равна 1 м/сек, изменение направление движения происходит моментально.

Выходные данные

В выходной файл необходимо выдать одно целое число — место, где будет находиться мадам Грицацуева в момент времени T.

Пример

input.txt	*output.txt*
20 2 9 L 10 2 0 0 6 0 1 3	5

Задача 11. Морской бой

После испытаний промышленного образца машины и написания предыдущих десяти задач возникла идея проверить машину на тексте этих задач. В качестве продолжения была сгенерирована следующая задача.

Автоматическое береговое орудие обстреливает участок поля, на котором находится корабль противника. При этом все поле представлено в памяти орудия схематично как прямоугольник размером $N \times M$, разбитый на квадраты 1×1 (клетки). Каждой клетке, согласно разведданным, задана оценка от 1 до 3. Орудие каждым выстрелом поражает какую-либо клетку поля.

Корабль состоит из секторов, каждый из которых имеет размер 1×1 и занимает полностью одну клетку поля на карте. Всего возможны три формы корабля – 1×1, 1×2, 1×3, корабль может быть расположен на поле по горизонтали или по вертикали.

Орудие спроектировано так, что оно запоминает всю информацию, полученную в ходе обстрела, и использует ее при выборе клетки для следующего выстрела.

Огонь ведется согласно следующему алгоритму:

1. Выбираются все клетки, по которым еще не производилось выстрелов, и в которых есть ненулевая вероятность нахождения противника (при этом орудие подсчитывает вероятности, используя информацию, которую оно знает к этому времени).

2. Из всех выбранных на предыдущем шаге клеток случайно выбирается одна. Выбор производится на основании следующих оценок. Пусть на предыдущем шаге были отобраны k клеток с оценками $q_1, q_2, \ldots, q_k$. Тогда вероятность выбора среди них клетки j с оценкой q_j равна $\frac{q_j}{\sum_{i=1}^{k} q_i}$.

3. Производится выстрел по выбранной клетке и орудие получает информацию — произошел промах, повреждение какого-то сектора корабля или полное его уничтожение. Корабль считается полностью уничтоженным, если произведены выстрелы по всем клеткам, где расположены его сектора.

4. Орудие повторяет эти действия до полного уничтожения корабля противника.

В начальный период времени орудие **не знает** форму и расположение корабля противника, и не имеет никакой информации о клетках поля, кроме их оценок, но **знает** размер поля, а также то, что на поле расположен ровно один корабль и имеет информацию обо всех трех возможных формах кораблей.

Вам требуется написать программу, которая по заданным размерам поля, оценкам клеток поля

и расположению корабля вычислит математическое ожидание числа выстрелов, которое потребуется орудию для уничтожения корабля.

Математическое ожидание вычисляется по формуле: $\sum_{i=1}^{N*M} i * p_i$,

где i – количество выстрелов, а p_i –вероятность того, что ровно после i-го выстрела корабль будет уничтожен.

Входные данные

В первой строке заданы два натуральных числа – N и M ($1 \le N \cdot M \le 50$).

Начиная со второй, идут N строк по M символов из множества {‘1’, ‘2’, ‘3’} в каждой. Эти символы представляют собой оценки соответствующих клеток поля.

Далее идут N строк, в каждой из которых по M символов. Символ ‘**.**’ (точка) обозначает пустую клетку, а символ ‘**x**’ (маленькая латинская буква x) означает, что в данной клетке расположен сектор корабля.

Выходные данные

В выходной файл требуется вывести одно число — искомое математическое ожидание с точностью до шести знаков после запятой.

Примеры

input.txt	*output.txt*
2 2 31 33 .x ..	3.25
2 2 33 33 .. .x	2.5
2 3 111 222 ... xxx	3.6
3 3 122 233 122 ... .xx ...	4.435714285714285
1 5	3.75

11111 .xx..	

Задача 12. Преобразование (только для первой лиги)

Исследуя библиотеку функций, используемых в микрокоде полученной машины, на предмет возможности её применения в других разработках, Привалов наткнулся на следующую функцию от неотрицательного целого числа.
Двоичная запись числа просматривается слева направо, и число разбивается на наибольшие последовательности подряд идущих нулей или единиц. Затем каждая такая последовательность заменяется на двоичное представление её размера. Получившееся число и является результатом работы функции.
Например, $f(1111000110) = 10011101$ (1111 заменяется на 100, 000 заменяется на 11, 11 на 10, 0 на 1), а $f(100000011111) = 1110101$.

Привалов заинтересовался вопросом, будет ли последовательность $x, f(x), f(f(x))$... периодической, и если будет, то какова длина периода этой последовательности.

Входные данные

Во входном файле задано число x, не превосходящее 10^{100}.

Выходные данные

Первая строка выходного файла содержит слово `YES`, если последовательность становится периодической, и `NO` в другом случае.

В случае, если ответ `YES`, вторая строка содержит длину периода, третья — первое повторяющееся число.

Примеры

input.txt	*output.txt*
1	YES 1 1
2	YES 3 11

Задача 13. Размер вознаграждения (только для первой лиги)

Переговорами с представителем заказчика об окончательной сумме выплат за реализацию проекта занялся Янус-А.

В результате переговоров с представителем заказчика об окончательной сумме выплат за реализацию проекта было принято следующее решение. К изначальной сумме (целое положительное число) для получения окончательной суммы требовалось дописать цифры из некоторой заданной последовательности цифр. Дописывать цифры разрешается между двумя любыми цифрами существующего числа, а также в начало и в конец числа. Каждая цифра из последовательности должна быть использована ровно один раз (например, для последовательности 1,1,1 должны быть добавлены на какие-то места три единицы).

Требуется дописать все цифры последовательности так, чтобы полученное в результате дописывания число было максимальным.

Входные данные

Первая строка ввода содержит изначальное число n, состоящее из не более, чем 10^4 знаков.
Вторая строка содержит не более 10^4 цифр, которые надо дописать.

Выходные данные

Выведите требуемое максимальное значение.

Пример

input.txt	*output.txt*
27 17	7271