List of Problems

Problem	Time limit	Memory limit	Name	Files
1	2 seconds	256 MiB	Герои 2 (только Division 1!)	input.txt or stdin, output.txt or stdout
2	2 seconds	256 MiB	Ганди и Данди	input.txt or stdin, output.txt or stdout
3	2 seconds	256 MiB	Живоглоты против бармалеев	input.txt or stdin, output.txt or stdout
4	2 seconds	256 MiB	Ночь в библиотеке	input.txt or stdin, output.txt or stdout
5	2 seconds	256 MiB	Конструктор (только Division 1!)	input.txt or stdin, output.txt or stdout
6	2 seconds	256 MiB	Смешивание жидкостей	input.txt or stdin, output.txt or stdout
7	2 seconds	256 MiB	Сборка гирлянды	input.txt or stdin, output.txt or stdout
8	2 seconds	256 MiB	Поиск шаблона	input.txt or stdin, output.txt or stdout
9	2 seconds	256 MiB	Дерево (только Division 1!)	input.txt or stdin, output.txt or stdout
10	2 seconds	266 MiB	Автопробег	input.txt or stdin, output.txt or stdout
11	2 seconds	256 MiB	Bug Preprocessor (только Division 2!)	input.txt or stdin, output.txt or stdout
12	2 seconds	256 MiB	Таблетки (только Division 2!)	input.txt or stdin, output.txt or stdout
13	2 seconds	256 MiB	Летающий слоупок (только Division 2!)	input.txt or stdin, output.txt or stdout

Unless explicitly stated in the problem statements, all input elements may be separated by an arbitrary number of whitespace characters. All input data are correct and satisfy the specification given in the problem statement.

The output of your program must exactly satisfy the output specification in the problem statement.

Each line of input is terminated by end-of-line marker (character with code 0A hex).

Задача 1. Герои 2 (Только для первого дивизиона)

Недавно вышла новая игра «Герои клавиатуры и мыши 2». 4Д-экшон суть такова. В четырёхмерном пространстве расположены некоторые города; каждый город расположен в некоторой точке. Можно строить новые города. Можно грабить корованы.

В любой момент все города окружены единой связной стеной конечного размера. Эта стена делит игровое пространство на две части: внутри неё и снаружи. Стена строится таким образом, что:

1. Все города находятся внутри стены.
2. Между любыми двумя точками внутри стены можно пройти по прямой линии, не пересекая при этом стену.
3. Область внутри стены минимальна при соблюдении условий 1 и 2.

Когда строится новый город, стена должна быть перестроена, если новый город расположен снаружи неё. Ваша задача заключается в том, чтобы определять для каждого вновь построенного города, требуется ли перестройка стены.

Гарантируется, что новый город всегда строится либо строго внутри, либо строго снаружи стены. Более того, расстояние от нового города до стены всегда больше 10^{-3}. Никакие два города не находятся в одном месте.

Входные данные

Игра начинается с пяти городов с координатами (x_1, y_1, z_1, w_1), (x_2, y_2, z_2, w_2), …, (x_5, y_5, z_5, w_5), которые заданы в первых пяти строках входного файла. Изначальный четырёхмерный объём внутри стены строго положителен.

Вторая строка содержит целое число N — количество достраиваемых городов ($1 \le N \le 800$). Каждая из следующих N строк содержит по четыре целых числа — координаты точки, в которой строится новый город. Все координаты не превосходят 5000 по абсолютной величине.

Выходные данные

Выходной файл должен содержать N строк. В K-ой строке должно быть записано слово **`Rebuild`**, если после достраивания K-ого города требуется перестроение стены, и **`Ignore`** в противном случае ($1 \le K \le N$).

Пример

input.txt	output.txt
0 0 0 0 8 0 0 0 0 8 0 0 0 0 8 0 0 0 0 8 5 1 2 2 2 2 2 3 2 8 8 8 8 3 5 3 4 -1 3 7 2	Ignore Rebuild Rebuild Ignore Rebuild

Комментарий

Первоначально пять городов расположены в вершинах координатного симплекса с длиной сторон 8 вдоль осей. Стена совпадает с границей этого симплекса. Уравнение большой гиперграни этого симплекса имеет вид $x + y + z + w = 8$.

Из этого уравнения легко видеть, что первый достраиваемый город лежит внутри симплекса, и перестроение стены не требуется, а второй город лежит снаружи симплекса, что

приводит к перестроению стены. После перестроения стены внутренняя область состоит из двух смежных симплексов.

После добавления третьего города стена перестраивается, после чего область внутри стены также состоит из двух симплексов. Четвёртый город находится внутри этой области, а пятый — очевидно снаружи.

Задача 2. Ганди и Данди

Ганди поехал в Данди. А Ганди, как известно, голова, и палец ему в рот не клади — все, до чего он может дотянуться, он поедает на своем пути.

А дотянуться он может до всех точек, расстояние от которых до некоторой точки на его пути не превосходит его радиуса поедания. При этом он едет из пункта А в Данди по кратчайшему пути, который, как известно, на земном шаре представляет дугу большого круга, проходящую через начальную и конечную точки его маршрута. При этом дуга от начальной до конечной точки не более половины длины большого круга.

Расстояние до точек, которые он пытается съесть, измеряются по поверхности Земли. Нам же необходимо найти площадь той территории, на которой нельзя находиться из-за риска быть съеденным Ганди.

Входные данные

В первой строке входного файла заданы шесть вещественных чисел x_0, y_0, x_1, y_1, R, eps — широта и долгота начала отрезка (в градусах) и, соответственно, конца отрезка, радиус земного шара, радиус поедания Ганди ($-90 \le x_0, x_1 \le 90$, $-180 \le y_0, y_1 \le 180$, $1 \le R \le 10000$, $0 \le eps \le R$).

Выходные данные

В выходной файл необходимо вывести одно вещественное число — площадь опасной территории с абсолютной или относительной погрешностью, не превосходящей 10^{-8}.

Пример

input.txt	output.txt
`-90 5 90 -100 10 5`	`378.1490948518`

Задача 3. Живоглоты против бармалеев

Арена представляет собой квадратное поле размером N×N клеток. На арене расположены живоглоты и бармалеи. Каждый из живоглотов и бармалеев имеет "силу", выраженную неотрицательным целым числом, у живоглотов сила задается нечётными числами, а у бармалеев — чётными. Живоглоты всё время движутся слева направо, а бармалеи — справа налево. При столкновении живоглота и бармалея побеждает тот, чья сила больше, слабый погибает (исчезает с арены), а сила победителя остаётся неизменной.

Живоглот не может перепрыгнуть или наступить на живоглота, а бармалей — на бармалея. Определите, сколько останется живоглотов и бармалеев на арене, в тот момент, когда никакое дальнейшее движение невозможно.

Входные данные

В первой строке входного файла записано одно целое число N ($1 \le N \le 100$).

Следующие N строк содержат по N чисел, записанных через пробел. В `i`-й строке `j`-й символ обозначает силу живоглота или бармалея, находящегося в клетке с координатами `(i, j)`, либо равен **`-1`**, если клетка свободна. Значения чисел, выражающих силу, не превосходят 100.

Выходные данные

В выходной файл необходимо вывести два числа, записанных через пробел — количество оставшихся бармалеев и живоглотов, соответственно.

Пример

input.txt	output.txt
4 5 -1 4 -1 3 -1 1 -1 -1 2 -1 4 -1 -1 -1 -1	2 3

Задача 4. Ночь в библиотеке

Однажды летом Вовочка проходил в школьной библиотеке «летнюю практику»: чинил учебники, помогал библиотекарям сортировать книги по отделам и тому подобное. У Вовочки очень длинный нос, который он вечно сует куда не следует, и библиотечные книги не стали исключением.

В одной из книг он увидел следующее:

> … обозначаемый через D(M) — кососимметричная полилинейная нормированная функция столбцов квадратной матрицы, т.е.:
> 1. Если поменять местами два столбца матрицы, его значение меняет знак:
> $$D([C_1, C_2, \dots, C_i, \dots, C_j, \dots, C_n]) = -D([C_1, C_2, \dots, C_j, \dots, C_i, \dots, C_n])$$
> 2. Если один из столбцов матрицы является линейной комбинацией двух векторов, то линейную комбинацию можно вынести:
> $$D([C_1, C_2, \dots, C_{i-1}, \alpha A + \beta B, C_{i+1}, \dots, C_n]) = \alpha\, D([C_1, C_2, \dots, C_{i-1}, A, C_{i+1}, \dots, C_n]) + \beta\, D([C_1, C_2, \dots, C_{i-1}, B, C_{i+1}, \dots, C_n])$$
> 3. Для единичной матрицы он равен единице:
> $$D(E) = D([e_1, e_2, \dots, e_n]) = 1$$
>
> Докажем, что такая функция единственна:
>
> … *(Вовочка читает книги обрывками)* …
>
> … например, методом исключения Гаусса.
>
> … *(иногда совсем маленькими)* …
>
> … матрица X является произведением столбца u на строку v, т.е.:
> $$X_{ij} = u_i v_j$$

Вовочку очень заинтересовало то, что он прочитал. Поскольку Вовочка увлекается олимпиадным программированием (т.е. зависает на сайте «Кодефорсес»), то матрицы и векторы ему не страшны. Однако с математикой у него туго.

Он хочет посчитать значение описанной кососимметричной полилинейной нормированной функции D для матрицы X, определённой в последнем отрывке. Для простоты все элементы векторов u и v являются целыми числами. Что-то подсказывает Вовочке, что ответ также будет целым. Ну а поскольку есть подозрения, что ответ может быть очень большим, то Вовочка хочет научиться находить его по модулю простого числа P (просто так принято).

Помогите Вовочке решить задачу. Иначе спам на вышеупомянутом сайте обеспечен.

Входные данные

Первая строка содержит два целых числа N и P ($1 \le N \le 100$, $2 \le P \le 2 \cdot 10^9$). Гарантируется, что P — простое число. Вторая строка содержит N целых чисел $u_1, u_2, \dots, u_n$ — элементы вектора-столбца u, а третья строка содержит N целых чисел $v_1, v_2, \dots, v_n$ — элементы вектора-строки v ($0 \le u_i, v_j < P$).

Выходные данные

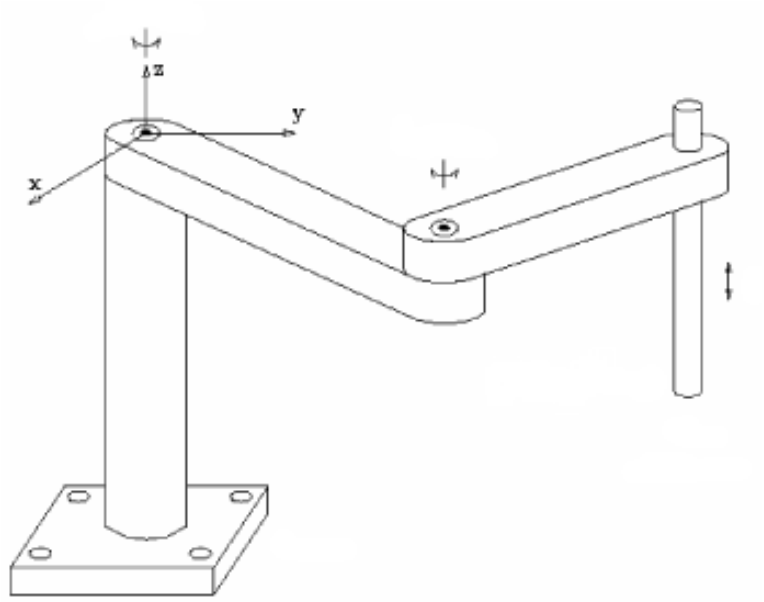
Требуется вывести одно целое число: значение D(X) по модулю P. Число должно быть неотрицательным и строго меньше, чем P.

Пример

input.txt	output.txt
3 11 0 7 8 10 5 3	0

Задача 5. Конструктор (Только для первого дивизиона)

Мальчику Пете недавно подарили конструктор для сборки роботов Mindrack 2.0. Почитав документацию, Петя решил собрать простой плоттер для рисования фигур. Плоттер представляет собой руку, состоящую из двух плеч, соединенных вращающимся шарниром, и имеющую крепление для карандаша на конце. Рука закреплена на подставке также при помощи шарнира. Дополнительно рука имеет сервоприводы, которые могут поворачивать ее относительно основания и один сегмент относительно другого. Рука устанавливается на подставке с бумагой, на которой можно рисовать различные фигуры. Схематично рука изображена на картинке ниже.



Для рисования фигур робо-рукой Петя написал программу, которая позволяет рисовать различные замкнутые ломаные по заданному набору координат вершин за счет согласованного управления сервомоторами. При этом рука рисует ломаную, не отрывая карандаш от бумаги. Единственная проблема, с которой столкнулся Петя, заключается в том, что шарниры руки не могут неограниченно проворачиваться. Шарнир в основании может проворачиваться не более чем на φ градусов влево и вправо относительно своего среднего положения. Шарнир, соединяющий плечи, может проворачиваться не более чем на θ градусов в обе стороны относительно положения, когда одно плечо сонаправлено с другим. Поэтому при рисовании ломаных могут возникать ситуации, когда рука не может продолжить рисование без отрыва карандаша от бумаги. В этом случае Пете приходится самостоятельно отсоединять карандаш от руки, переводить ее в новое положение, а затем присоединять карандаш. При этом рука продолжает рисовать ломаную с той же точки, на которой она закончила рисование перед «переключением». Естественно, что Петя хочет минимизировать количество таких ручных «переключений».

Предположим, что рука находится на координатной плоскости Oxy, причем она прикреплена к основанию в начале координат. Считается, что в среднем положении рука направлена в направлении оси Oy. Будем считать, что рука состоит из двух отрезков — длины L и l, соответствующих первому плечу, прикрепляющемуся к основанию, и второму плечу, на конце которого закреплен карандаш. Вам требуется по заданным параметрам руки — числам L, l, φ и θ, и координатам вершин замкнутой ломаной на плоскости определить минимально необходимое количество ручных переключений руки, которое требуется сделать Пете, чтобы нарисовать данную ломаную, либо определить, что это невозможно. Рука может начинать рисование с любой точки ломаной. Однако после того, как выбрана стартовая точка и направление обхода ломаной при рисовании, менять это направление уже нельзя. Также после

переключения рука должна продолжить рисование с той же точки, на которой произошло переключение режима, и в том же направлении. Руке не разрешается проводить карандашом по уже нарисованной части ломаной. В случае, когда ломаная имеет самопересечения либо самокасания, при рисовании не разрешается «перепрыгивать» в таких точках на другую часть ломаной.

Входные данные

В первой строке входного файла заданы четыре целых неотрицательных числа — L, l, φ и θ ($1 \le L + l \le 10^4$, $\varphi + \theta \le 180$). В следующей строке записано целое число N — количество вершин ломаной ($3 \le N \le 100000$). Далее идут N строк с парами целых чисел x_i и y_i – декартовыми координатами вершин ломаной. Координаты по модулю не превосходят 10^4. Никакие две вершины не совпадают. Считается, что ломаная состоит из отрезков с вершинами (x_1, y_1)–(x_2, y_2), (x_2, y_2)–(x_3, y_3),…, (x_{N-1}, y_{N-1})–(x_N, y_N) и (x_N, y_N)–(x_1, y_1).

Выходные данные

В единственную строку выходного файла необходимо вывести минимально необходимое количество ручных переключений руки для рисования заданной ломаной, либо число **–1**, если нарисовать ломаную указанным способом невозможно.

Пример

input.txt	output.txt
4 4 90 90 6 -6 2 -3 6 3 6 6 2 3 7 -3 7	1
1 1 45 45 3 1 1 5 1 1 5	-1

Задача 6. Смешивание жидкостей

На заседании общества "Меча и орала" чисто в конспиративных целях гоняют чаи и еще много чего. В горячий чай добавляют холодное молоко. При этом, естественно, каждый хочет, чтобы его смесь достигла необходимой температуры как можно быстрее. Изменение температуры смеси происходит по экспоненциальному закону:

$$T(t) = T_0 + (T_1 - T_0)e^{-kt},$$

где T(t) — температура в момент времени t,

T_1 — температура смеси в начальный момент времени,

T_0 — температура окружающей среды,

k — некоторый постоянный коэффициент.

При смешении двух жидкостей с массами m_1 и m_2 и температурами T_1 и T_2 получается смесь с температурой

$$T_1' = \frac{T_1 \cdot m_1 + T_2 \cdot m_2}{m_1 + m_2}$$

Считаем, что удельные теплоемкости всех жидкостей на этом чаепитии одинаковые. Температура первой жидкости сразу в начальный момент времени начинает изменяться по приведенному выше экспоненциальному закону, в любой момент времени мы можем ее мгновенно смешать со второй жидкостью — ее температура все это время постоянна и равна T_2, после чего температура полученной смеси изменяется по тому же закону с тем же коэффициентом k и новой температурой T_1'.

Нам же необходимо посчитать минимальное время, за которое приготавливаемая смесь может достичь необходимой температуры.

Входные данные

В первой строке входного файла записано одно целое число N — количество тестов. Далее следуют N строк, в каждой из которых по семь целых чисел T_0, T_1, T_2, m_1, m_2, T_{opt}, k. Соответственно, T_0, T_1, T_2, T_{opt} — температуры окружающей среды, первой и второй жидкостей и температура, которую нам необходимо получить ($-273 < T_0, T_1, T_2, T_{opt} \le 1000$), m_1 и m_2 — массы первой и второй жидкостей ($0 \le m_1, m_2 \le 1000$, $m_1+m_2 > 0$), k — коэффициент, отвечающий за скорость изменения температуры ($1 \le k \le 1000$).

Выходные данные

В каждую из N строк в соответствии с последовательностью данных во входном файле вывести по одному вещественному числу — минимальное время, за которое мы можем получить смесь требуемой температуры, посчитанное с относительной или абсолютной погрешностью не более 10^{-8} или сообщение **`Impossible`**, если достичь требуемой температуры никогда не удастся.

Пример

input.txt	output.txt
2 0 1 1 10 10 1 1 0 1 1 10 10 0 1	0 Impossible

Задача 7. Сборка гирлянды

Вася на Новый год купил ёлочную гирлянду «Сделай сам».

В ее комплект входят:

- 2·N разноцветных лампочек, среди них нет лампочек одинакового цвета;
- 2·N патронов, в которые эти лампочки нужно вкрутить;
- достаточное количество проводов, чтобы соединить патроны с вкрученными в них лампочками в сеть.

Каждый патрон имеет некоторое количество контактов для подсоединения проводов. В комплекте представлено N типов патронов. Патрон первого типа имеет один контакт для подсоединения провода, у патрона второго типа таких контактов два, у патрона третьего типа — три контакта для проводов, и т.д. Патроны типа N снабжены N контактами для подсоединения проводов. Патронов каждого типа представлено ровно по две штуки.

Гирлянда заработает, если так распределить лампочки по патронам и соединить соответствующие патроны проводами, чтобы выполнились следующие условия:

- у всех патронов все контакты для проводов должны быть использованы;
- к одному контакту патрона должен быть подсоединен только один провод;
- у любого провода оба конца должны быть подсоединены к контактам двух разных патронов;
- нельзя два различных патрона соединить более чем одним проводом.

Для Васи эта задача показалась очень трудной. Помогите ему собрать гирлянду к Новому году.

Входные данные

Во входном файле записано целое число N — количество типов патронов ($1 \le N \le 500$).

Выходные данные

В первую строку выходного файла необходимо вывести целое число M — количество проводов, которые потребуются для сборки гирлянды. В следующие M строк нужно вывести через пробел по два целых числа — номера лампочек, вкрученных в патроны, соединенные одним проводом. Лампочки нумеруются числами от 1 до 2·N. Если решений несколько, то выведите любое.

Пример

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

Задача 8. Поиск шаблона

Строка состоит из символов двух непересекающихся алфавитов A_1 и A_2. Две строки будут равны, если существует взаимно-однозначное отображение символов из алфавита A2 в символы алфавита A2, после применения которого к одной из строк, строки будут совпадать.

При таком определении равенства строк будем говорить, что для заданного шаблона существует его вхождение в строку, если в строке найдется подстрока, равная этому шаблону.

Для заданных строки и шаблона необходимо посчитать количество вхождений этого шаблона в строку.

Входные данные

В первой строке входного файла записано два целых числа N и M — количество символов в первом и во втором алфавитах соответственно ($1 \le N, M < 52$).

В следующих двух строках записаны без пробелов символы из алфавитов A_1 и A_2. Символами из алфавитов могут быть строчные и прописные латинские буквы.

Четвертая строка содержит строку, а пятая — шаблон. Длины строки и шаблона не превышают 10^5.

Выходные данные

В выходной файл необходимо вывести одно целое число — количество вхождений шаблона в строку.

Примеры

input.txt	output.txt
8 4 aIbctlyd eouh Itellyeh you	1
2 6 cq xyztab qxycxycztc abc	3

Задача 9. Дерево (Только для первого дивизиона)

В упорядоченном корневом дереве у каждой нелистовой вершины определён порядок сыновей. Деревья считаются изоморфными, если при некотором взаимно-однозначном отображении вершин сохраняются:

- отношение «вершина s является сыном вершины f»,
- порядок сыновей у каждой вершины.

Дано упорядоченное корневое дерево T и список его модификаций. Модификации имеют вид: отрезать вершину с заданным номером вместе с поддеревом от её текущего отца и добавить в качестве самого правого сына к некоторой вершине.

Требуется определить, после какой модификации дерево впервые становится изоморфным какому-то из предыдущих деревьев.

Входные данные

В первой строке входного файла записано два целых числа: **N** — количество вершин в дереве и M — количество модификаций ($2 \le N \le 100000$, $1 \le M \le 100000$). Вершины дерева пронумерованы натуральными числами от 1 до N. Корень дерева имеет номер 1.

В следующих N строках записаны упорядоченные списки сыновей для вершин исходного дерева. Каждый i-ый список описывается количеством K_i сыновей i-ой вершины и последовательностью номеров её вершин-сыновей ($0 \le K_i \le N - 1$).

В оставшихся M строках описаны модификации дерева. Каждая модификация определяется двумя целыми числами S_j и F_j – номером вершины, которую вместе с поддеревом отрезают от отца, и номером вершины, к которой её добавляют в качестве самого правого сына ($2 \le S_j \le N$, $1 \le F_j \le N$). Гарантируется, что вершина F_j не является потомком вершины S_j и не совпадает с ней.

Выходные данные

Если все M + 1 получающихся деревьев различны, в выходной файл необходимо вывести число **–1**.

В противном случае нужно вывести через пробел два целых числа A и B — номера двух изоморфных деревьев ($0 \le A < B \le M$). Если пар изоморфных деревьев несколько, требуется вывести пару с минимальным номером B.

Деревья нумеруются следующим образом: исходное дерево имеет номер 0. После j-ой модификации получается j-ое дерево ($1 \le j \le M$).

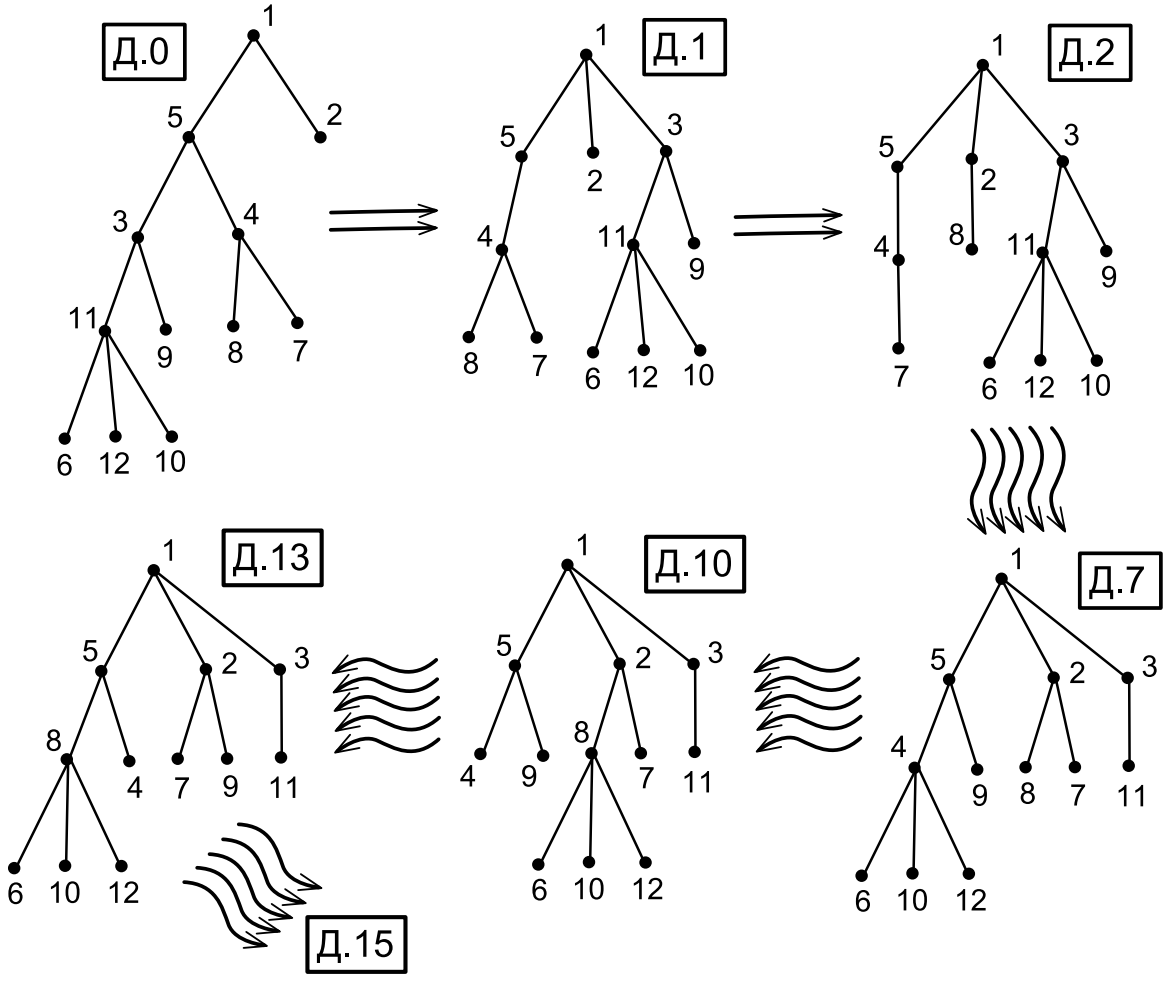
Примеры

input.txt	output.txt
12 15 2 5 2 0 2 11 9 2 8 7 2 3 4 0 0 0 0 0 3 6 12 10 0 3 1 8 2 7 2 6 4	7 13

10 4 12 4 9 5 6 8 10 8 12 8 8 5 9 2 4 5 6 9 6 8	
4 3 3 2 3 4 0 0 0 2 3 3 1 4 2	-1

Комментарий

Ниже приведены некоторые деревья для первого теста из условия.



Задача 10. Автопробег

Маршрут Антилопы-Гну пролегает по гостеприимным и хлебосольным местам. При этом существует некоторая вероятность, что в любой город на их маршруте поступит одна очень важная телеграмма, или вдруг Паниковский опять возьмется за старое, и тогда в этом городе придется подзадержаться на 24 часа. Естественно, время прохождения одного и того же маршрута — величина непостоянная и зависит от везения. В славный же город Черноморск дети лейтенанта Шмидта должны приехать как можно быстрее, ну не со стопроцентной вероятностью, а с вероятностью, не менее заданной. При этом вероятности задержаться в любом городе их маршрута одинаковые и сами эти задержки никак не зависят одна от другой. Назовем длительностью маршрута такое число T, что вероятность прохождения маршрута за время, не превышающее T, будет не менее заданной вероятности P. Во время прохождения маршрута включается возможная задержка в первом и последнем городе.

Помогите им найти маршрут минимальной длительности.

Входные данные

В первой строке два целых числа N и M — количество городов и количество дорог между ними и два вещественных числа P — заданная вероятность, с которой ищется время прохождения маршрута и P_1 — вероятность задержки на 24 часа в каждом городе ($2 \le N \le 1000$, $1 \le M \le 10000$, $0 \le P, P_1 \le 1$). P и P_1 даны с пятью знаками после десятичной точки.

Далее следуют M строк — описания дорог, в каждой три целых числа A_i, B_i, L_i — номера городов, соединенных данной дорогой и время ее прохождения в часах ($1 \le L_i \le 1000$). Города пронумерованы числами от 1 до N. Маршрут прокладывается из города с номером 1 в город с номером N. Каждая пара городов соединена не более чем одной дорогой. Ни одна дорога не соединяет город сам с собой. По дорогам можно ездить в обе стороны. Гарантируется, что между любыми двумя городами существует путь, и что при изменении P на не более чем на 10^{-9} в любую сторону, длительность любого маршрута не изменится.

Выходные данные

В первую строку выходного файла необходимо вывести одно целое число — количество городов, через которые пролегает оптимальный маршрут. Во второй строке должны быть перечислены через пробел номера этих городов в порядке следования.

Пример

input.txt	output.txt
4 4 0.99000 0.50000 1 2 1 2 3 1 3 4 1 1 4 4	2 1 4

Задача 11. Bug Preprocessor (только для Division 2!)

Группа байтландских программистов объявила о старте нового OpenSource проекта — Bug Preprocessor. Этот препроцессор находит все баги в поданном на вход коде программы и заменяет их специальной строкой-маркером.

Ваша задача — написать программу, которая удаляет маркеры (с учётом регистра!) из обработанного препроцессором кода программы.

Формат входного файла

Входной файл состоит из не более, чем 10 тестовых примеров. Каждый тестовый пример начинается со строки, содержащий целое число T ($0 \leqslant T \leqslant 1000$), а также строку-маркер B, состоящую из не более, чем 10 заглавных латинских букв (от '`A`' до '`Z`')

В последующих T строках задан исходный код программы после обработки препроцессором. Все найденные баги заменены строкой B. Каждая из T строк имеет длину не менее 0 и не более 200 символов.

Формат выходного файла

Выведите исходный текст программы, из которого были бы удалены все строки-маркеры B. Строка-маркер должна удаляться целиком, при этом ничего, что бы не являлось строкой-маркером, удалено быть не должно (в том числе и пробелы).

Пример

input.txt	output.txt

7 BUG
print "No bugs here..."
 	
void hello() {
BUGBUG
printfBUG("Hello, world!
n);
}
 	
1 ERR
wriEERRRRtelERRn("Hello E-R-R");
	print "No bugs here..."
	
	void hello() {
	
	printf("Hello, world!
	n);
	}
	
	writeln("Hello E-R-R");

Задача 12. Таблетки (только для Division 2!)

Врач выписал тёте Полли лекарство; принимать следовало по половине таблетки ежедневно. Первоначально в склянке было N таблеток.

В первый день тётя Полли взяла случайную таблетку из склянки, разрезала таблетку пополам, половину таблетки приняла, а другую бросила назад в склянку.

Каждый последующий день она вытряхивала из склянки одну таблетку (или половинку таблетки); если вытряхнулась половина таблетки, то эта половина и использовалась, если же выпадала целая таблетка, тётя разрезала её пополам, одну половинку принимала, а вторую отправляла назад в склянку.

Любопытный Том Сойер решил записывать последовательность приёма таблеток с самого первого дня следующим образом: если тётя вытаскивала целую таблетку, он записывал в соответствующей позиции букву 'W', если половину — 'H'. Получилась строка длины $2N$. Теперь Том задумался над вопросом — сколько существует различных строк такой длины, описывающих корректные сценарии приёма таблеток.

Сценарий является возможным, если ни на каком шаге не оказывается, что целых таблеток (в случае 'W') или половинок (в случае 'H') в склянке нет.

Формат входного файла

Входной файл содержит несколько (не более 1000) тестовых примеров. Каждый тестовый пример состоит из одной строки, содержащей целое число — изначальное количество N таблеток в склянке ($1 \leqslant N \leqslant 30$). Входной файл завершается примером с $N = 0$, обрабатывать который не следует.

Формат выходного файла

Для каждого тестового примера в отдельной строке выведите целое число — количество различных строк длины $2N$, описывающих корректные сценарии приёма таблеток.

Пример

input.txt	output.txt
6 1 4 2 3 30 0	132 1 14 2 5 3814986502092304

Задача 13. Летающий слоупок (только для Division 2!)

Байтландские учёные вывели новый вид животных — летающего слоупока. Он не только летает медленно, но и медленно разворачивается (и то только перед вылетом).

По заданной скорости движения слоупока по прямой и скорости поворота найдите минимальное время, за которое слоупок долетит из точки отправления в точку назначения.

Изначально слоупок смотрит на север (вдоль оси Y); в момент начала движения он разворачивается в направлении точки назначения и затем летит до неё строго по прямой.

Формат входного файла

Входной файл состоит из нескольких (не более 150) тестовых примеров. Первая строка каждого тестового примера содержит два целых числа S и T ($1 \leqslant S, T \leqslant 1000$), где S — скорость слоупока в метрах в секунду, T — скорость его разворота в градусах в секунду.

Следующая строка содержит четыре целых числа $0 \leqslant X_f, Y_f, X_t, Y_t \leqslant 10^4$ — соответственно координаты точки отправления и точки назначения, заданные в метрах.

Формат выходного файла

Для каждого тестового примера в отдельной строке выведите целое число R — минимальное время, которое слоупок затратит на полёт.

Ответ выводить с абсолютной погрешностью не более 10^{-3}.

Пример

input.txt	output.txt
3 10 0 0 0 10 3 10 0 0 10 10	3.3333 9.2140