Задача A. Автогонки

Имя входного файла:	`auto.in`
Имя выходного файла:	`auto.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

В городе N в ближайшее время состоится этап чемпионата мира по автогонкам среди автомобилей класса *Формула-0*. Поскольку специальный автодром для этих соревнований организаторы построить не успели, было решено организовать трассу на улицах города.

В городе N есть n перекрестков, некоторые пары которых соединены дорогами, движение по которым возможно в обоих направлениях. При этом любые два перекрестка соединены не более чем одной дорогой, и есть возможность доехать по дорогам от любого перекрестка до любого другого.

Трасса, на которой будут проводиться соревнования, должна быть круговой (т.е. должна начинаться и заканчиваться на одном и том же перекрестке), при этом в процессе движения по ней никакой перекресток не должен встречаться более одного раза.

На предварительном этапе подготовки оргкомитетом был создан список всех дорог города. Теперь настало время его использовать. Первый вопрос, который необходимо решить, — вопрос о существовании в городе требуемой круговой трассы (разумеется, если ответ будет отрицательным, орагнизаторам придется в срочном порядке построить еще несколько дорог). Единственная проблема заключается в том, что у организаторов есть подозрение, что, поскольку список составлялся не очень внимательно, в нем некоторые дороги указаны больше одного раза.

Напишите программу, которая по заданному списку дорог города определит, возможна ли организация в городе требуемой круговой трассы.

Формат входного файла

Первая строка входного файла содержит два целых числа: n ($1 \le n \le 1000$) — количество перекрестков в городе N и m ($0 \le m \le 100000$) — количество дорог в составленном списке.

Последующие m строк описывают дороги. Каждая дорога описывается двумя числами: u и v ($1 \le u, v \le n$, $u \ne v$) — номерами перекрестков, которые она соединяет. Так как дороги двусторонние, то пара чисел (u, v) и пара чисел (v, u) описывают одну и ту же дорогу.

Формат выходного файла

В выходной файл выведите слово `YES`, если в городе возможно организовать круговую трассу для соревнований, и слово `NO` — в противном случае.

Примеры

auto.in	auto.out
3 4 1 2 2 3 3 1 3 2	YES
2 3 1 2 2 1 2 1	NO

Задача B. Цепочка слов

Имя входного файла:	`chain.in`
Имя выходного файла:	`chain.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Цепочкой слов длины n назовем последовательность слов $w_1, w_2, \ldots, w_n$ такую, что для $1 \le i < n$ слово w_i является собственным префиксом слова w_{i+1}.

Напомним, что слово u длины k называется *собственным префиксом* слова v длины l, если $l > k$ и первые k букв слова v совпадают со словом u.

Задано множество слов $S = \{s_1, s_2, \ldots, s_m\}$. Найдите максимальную длину цепочки слов, которую можно построить, используя (возможно, не все) слова этого множества.

Формат входного файла

Первая строка входного файла содержит целое число m ($1 \le m \le 255$). Каждая из последующих m строк содержит по одному слову из множества S.

Все слова не пусты, имеют длину, не превосходящую 255 символов, и состоят только из строчных букв латинского алфавита.

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

chain.in	chain.out
3 a ab abc	3
5 a ab bc bcd add	2

Задача C. Шифр «Решетка»

Имя входного файла:	`cipher.in`
Имя выходного файла:	`cipher.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Рассмотрим перестановочный шифр, называемый «Решетка» («перестановочный» означает, что символы, составляющие послание, не изменяются, но меняются местами). Суть его заключается в следующем. Выбирается четное число n, затем в квадрате $n \times n$ вырезается $n^2/4$ клеток. При этом клетки выбираются так, что если наложить решетку на квадрат $n \times n$, и затем последовательно развернуть ее на 90, 180 и 270 градусов, то каждый раз квадратики, совмещенные с вырезанными клетками, будут различны.

Такой квадрат $n \times n$ называется «правильным ключом». Ваша задача — посчитать количество «правильных ключей». Так как это число может быть очень большим, мы предлагаем Вам найти его значение по модулю m. Ключи получаемые поворотом на 90, 180 и 270 градусов считаются различными.

Формат входного файла

В первой строке записаны целые числа n ($2 \le n \le 10^6$, n — четно) и m ($2 \le m \le 10^6$).

Формат выходного файла

Выведите количество «правильных ключей» размером $n \times n$ по модулю m.

Примеры

cipher.in	cipher.out
2 100	4

Задача D. Гирлянда

Имя входного файла:	`garland.in`
Имя выходного файла:	`garland.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Приближается Новый Год, и в магазинах начинают появляться различные елочные украшения. На прилавках можно увидеть различные шарики, шишечки, звездочки, но все-таки самым красивым украшением является гирлянда из разноцветных лампочек. Одна из фирм, занимающихся изготовлением елочных украшений, решила в этом году изготавливать гирлянды на заказ.

Гирлянды, изготавливаемые этой фирмы, состоят из лампочек различных цветов, соединенных проводами. Всего в гирлянде n лампочек, каждая из которых покрашена в один из k цветов, и m проводов (каждый провод соединяет ровно две лампочки). Далее мы будем считать, что лампочки пронумерованы натуральными числами от 1 до n.

К сожалению, не каждый дизайн гирлянды соответствует эстетическим взглядам заказчиков. Во-первых, лампочки, соединенные одним проводом должны быть разного цвета, во-вторых сама конфигурация гирлянды (то есть то, какие лампочки и как соединены проводами) не может быть любой.

Один из отделов фирмы уже провел исследование и нашел наиболее «удачную» конфигурацию. Ваша же задача состоит в том, чтобы найти число способов раскрасить лампочки, чтобы получившаяся гирлянда удовлетворяла эстетическим взглядам заказчиков.

Формат входного файла

Первая строка входного файла содержит три целых числа: n, k, m ($1 \le n, k \le 8$, $0 \le m \le 10$).

Последующие m строк описывают провода. Описание каждого провода состоит из двух чисел u и v ($1 \le u, v \le n$, $u \ne v$) — номеров лампочек, соединенных этим проводом.

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

garland.in	garland.out
2 2 1 1 2	2
4 4 0	256
4 4 6 1 2 1 3 1 4 2 3 2 4 3 4	24

Задача E. Игра «Жизнь»

Имя входного файла:	`life.in`
Имя выходного файла:	`life.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Игра «Жизнь» была придумана английским математиком Джоном Конвейем в 1970 году. Впервые описание этой игры опубликовано в октябрьском выпуске (1970) журнала *Scientific American*, в рубрике «Математические игры» Мартина Гарднера.

Место действия этой игры — «вселенная» — это размеченная на клетки поверхность. Каждая клетка на этой поверхности может находиться в двух состояниях: быть живой или быть мёртвой. Клетка имеет восемь соседей. Распределение живых клеток в начале игры называется первым поколением. Каждое следующее поколение рассчитывается на основе предыдущего по таким правилам:

- пустая (мёртвая) клетка с ровно тремя живыми клетками-соседями оживает;
- если у живой клетки есть две или три живые соседки, то эта клетка продолжает жить; в противном случае (если соседок меньше двух или больше трёх) клетка умирает (от «одиночества» или от «перенаселённости»).

В этой задаче рассматривается игра «Жизнь» на торе. Представим себе прямоугольник размером n строк на m столбцов. Для того, чтобы превратить его в тор мысленно «склеим» его верхнюю сторону с нижней, а левую — с правой. Таким образом, у каждой клетки, даже если она раньше находилась на границе прямоугольника, теперь есть ровно восемь соседей.

Ваша задача состоит в том, чтобы найти конфигурацию клеток, которая будет через k поколений от заданного.

Формат входного файла

Первая строка входного файла содержит три целых числа: n, m, k ($4 \le n, m \le 100$, $0 \le k \le 100$). Последующие n строк содержит по m символов каждая и описывают начальную конфигурацию. j-ый символ i-ой строки равен «.» (точка), если соответствующая клетка мертва, и «*» — если жива.

Формат выходного файла

В выходной файл выведите конфигурацию клеток через k поколений после начального в том же формате, в каком конфигурация задается во входном файле.

Примеры

life.in	life.out
5 5 1 **... ..**. .*... ..*.. ...*.	.*.** *.*.. .*.*. ..*.. .**..
5 5 5 **... ..**. .*... ..*.. ...*.	.***. .*... .*... ..**. .....
4 7 5 .*.*.*. *.*.*.* .*.*.*. *.*.*.*	....

Задача F. Оптимизация на окружности

Имя входного файла:	`optimize.in`
Имя выходного файла:	`optimize.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Найдите максимальное значение функции $f(x, y) = A\,x + B\,y$ при условии, что $x^2 + y^2 = R^2$, и аргументы, при котором оно достигается.

Формат входного файла

Входной файл содержит три вещественных числа: A, B, R ($1 \le A, B, R \le 10$), заданные не более чем с тремя знаками после десятичной точки.

Формат выходного файла

В первой строке выведите максимальное значение, которое достигает функция. Во второй строке выведите значения аргументов x_0, y_0, при которых это значение достигается. Все числа должны быть выведены с точностью не хуже 10^{-5}.

Примеры

optimize.in	optimize.out
`1 1 1`	`1.41421356237` `0.70710678119 0.70710678119`
`2.0 1.0 1.0`	`2.2360679775` `0.8944271910 0.4472135955`

Задача G. Словарные квадраты

Имя входного файла:	`square.in`
Имя выходного файла:	`square.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Некоторые наборы из n слов длины n обладают интересным свойством — их можно расположить в клетках квадрата $n \times n$ так, что все слова набора можно прочитать как во вертикали, так и по горизонтали. Причем слово, записанное в первой строке, должно быть записано в первом столбце, записанное во второй строке — во втором столбце, ..., записанное в n-ой строке — записано в n-ом столбце.

Примером такого набора слов является { "`DATE`", "`FIND`", "`IDEA`", "`NEXT`" }. Их можно расположить так:

F	I	N	D
I	D	E	A
N	E	X	T
D	A	T	E

Заметьте, что каждое слово можно прочитать как по горизонтали, так и по вертикали. Такие квадраты называются *словарными квадратами*, наибольший известный словарный квадрат в английском языке имеет размер 10×10.

Рассмотрим еще один пример словарного квадрата:

C	R	A	B
R	A	R	E
A	R	T	S
B	E	S	T

Задан квадрат 4×4, в каждой клетке которого стоит заглавная буква латинского алфавита. Необходимо проверить, является ли он словарным квадратом.

Формат входного файла

Входной файл содержит ровно четыре строки, каждая из которых содержит ровно четыре заглавные буквы латинского алфавита, разделенные пробелами.

Формат выходного файла

В выходной файл выведите фразу `WORD SQUARE`, если во входном файле задан словарный квадрат, и фразу `SOMETHING OTHER` — иначе.

Примеры

square.in	square.out
A B C D E F G H I J K L M N O P	SOMETHING OTHER
F I N D I D E A N E X T D A T E	WORD SQUARE
C R A B R A R E A R T S B E S T	WORD SQUARE
A A B B A A B B B B A A B B A A	WORD SQUARE

Задача H. Диалоги по *UCM*

Имя входного файла:	`ucm.in`
Имя выходного файла:	`ucm.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Федя живет активной жизнью онлайн. Он легко ищет самые сложные рефераты, используя всемирно известные поисковые машины, скачивает музыку и делится ею с друзьями, он присоединился к многим популярным сообществам в *Јивом Јурнале*. И конечно же, он много общается с интернет-друзьями по *UCM (You Seek Me)* — известному чат-клиенту.

Федя учится в школе. В школе изучают русский язык. Однажды учительница задала домашнее задание — написать не менее 10–15 фраз, используя косвенную речь.

Феде лень переписывать стандартные фразы из школьных учебников, он хочет мыслить нестандартно. И вот у Феди блеснула оригинальная идея — а не взять ли несколько диалогов из его любимого *UCM* и не перевести ли их в косвенную речь?

Идея хороша, но, к сожалению, Федя не умеет программировать, так как его еще не учили этому в школе. Помогите ему!

Формат входного файла

Во входном файле находится распечатка одного из диалогов по *UCM*. В первой строке находится сообщение о том, что собеседник Феди вошел с ним в контакт. Оно выглядит так:

`ЧЧ:ММ:СС: <Имя> signed on`

где `ЧЧ:ММ:СС` —время, когда собеседник вошел в контакт, `<Имя>` — имя собеседника, записанное в транслите (латинскими буквами). Любое число в отображении времени занимает две цифры, например, 9 часов 43 минуты 5 секунд будет выглядеть как 09:43:05.

В каждой из последующих строк (кроме самой последней строки файла) находится реплика участника диалога. Если каждой реплике присвоить её порядковый номер (начиная с единицы), то нечетные реплики были произнесены Федей, а четные — его собеседником.

Реплика находится целиком на одной строке и состоит из метки времени, смысловой части и знака препинания конца предложения.

Метка времени имеет вид `ЧЧ:ММ:СС:`.

Смысловая часть отделена от метки времени пробелом и содержит текст, состояший из больших или маленьких латинских букв, цифр, пробелов, апострофов (заменяют мягкий и твердый знаки в транслите), а также запятых, точек с запятыми и круглых скобок.

Знак препинания — это восклицательный знак, вопросительный знак или точка. Он может быть опущен, в этом случае подразумевается точка.

Пример реплики:

`11:22:33: Ya skazal etu frazu.`

Последняя строка файла имеет вид

`ЧЧ:ММ:СС: <Имя> signed off`

где `<Имя>` — имя все того же собеседника.

Объем входного файла не превышает 20 килобайт. Длина каждой строки не превосходит 250 символов.

Формат выходного файла

Выведите в выходной файл для каждой реплики её же, преобразованную в косвенную речь.

Преобразование заключается в следующем. Сначала заключите реплику в двойные кавычки, при этом если она оканчивается на точку, то вместо точки поставьте запятую, иначе оставьте знак препинания без изменений, например:

`Zakanchivayu tochkoi.` → `"Zakanchivayu tochkoi,"`

`Eto voskhititel'no!` → `"Eto voskhititel'no!"`

Затем поставьте пробел, три знака «-» (тире) и снова пробел, после чего выведите слово «skazal» и имя собеседника, произнесшего эту реплику. После имени выведите точку. Федя запишется как «Fedya», имя его собеседника можно узнать из первой строки входного файла.

Примеры

ucm.in	ucm.out
08:59:59: Vasya signed on 09:00:00: Privet! 09:00:31: Privet 09:00:59: Vasya signed off	"Privet!" --- skazal Fedya. "Privet," --- skazal Vasya.