Задача A. Нечётный букет

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Максим хочет подарить своей подруге Ире на день рождения букет цветов.

Около его дома есть магазин, в котором продаются цветы n видов. Максим выяснил, что в магазине есть a_i цветов i-го вида. Он знает, что Ира очень любит нечётные числа. Поэтому Максим решил, что цветов каждого вида в букете должно быть нечётное количество, а также общее число цветов в букете также должно быть нечётно.

Помогите Максиму определить, из какого наибольшего количества цветов он может собрать букет?

Формат входных данных

В первой строке находится целое число n — количество видов цветов, которые продаются в магазине ($1 \leqslant n \leqslant 100\,000$).

Во второй строке находятся n целых чисел $a_1, a_2, \ldots, a_n$ — для каждого вида цветов указано, сколько цветов этого вида есть в магазине ($1 \leqslant a_i \leqslant 1000$).

Формат выходных данных

Выведите единственное число — максимальное количество цветов, из которого может состоять букет.

Примеры

стандартный ввод	стандартный вывод
3 3 5 8	15
3 1 1 1	3

Задача B. Перекрытие отрезков

Ограничение по времени: 2 секунды
Ограничение по памяти: 512 мегабайт

Антон разрабатывает графический движок для 2D игр, который должен покорить рынок. Для того, чтобы объекты отображались корректно, ему нужно уметь выяснять, перекрывают ли объекты друг друга, если игрок смотрит в определённом направлении.

В предварительной версии движка объекты — это непересекающиеся отрезки на плоскости. Отрезок a перекрывает отрезок b по направлению вектора $\vec{v}$, если найдутся такие точки A на отрезке a и B на отрезке b, что вектор $\overrightarrow{AB}$ сонаправлен с вектором $\vec{v}$. Другими словами, какая-либо точка отрезка a, двигаясь по направлению вектора $\vec{v}$, окажется на отрезке b.

Антон сейчас слишком занят поиском инвесторов, поэтому попросил вас реализовать часть движка, которая проверяет перекрытие отрезков.

Формат входных данных

Первая строка содержит целое число n — количество проверок на перекрытие отрезков, которые необходимо выполнить ($1 \leqslant n \leqslant 50\,000$).

Каждая из следующих n строк содержит 10 целых чисел: ax_1, ay_1, ax_2, ay_2, bx_1, by_1, bx_2, by_2, vx, vy — координаты концов первого отрезка, координаты концов второго отрезка и координаты вектора, задающего направление, соответственно. Все координаты не превосходят 10^6 по абсолютной величине. Гарантируется, что отрезки невырожденные и не содержат общих точек, а также что вектор $\vec{v}$ ненулевой.

Формат выходных данных

Для каждой проверки выведите в отдельной строке «`Yes`», если первый отрезок перекрывает второй отрезок по заданному направлению, иначе выведите «`No`».

Пример

стандартный ввод	стандартный вывод
2 0 2 1 1 2 2 3 1 1 1 0 2 1 1 2 2 3 1 -1 -1	Yes No

Задача C. Великая теорема Ферма

Ограничение по времени: 4 секунды
Ограничение по памяти: 512 мегабайт

Как вам, вероятно, известно, для всех натуральных чисел a, b, c и n при $n \geqslant 3$ выполнено неравенство $a^n + b^n \neq c^n$. Однако все известные доказательства этого факта сложно проверить, поэтому группа программистов решила написать своё доказательство, проверить которое, по их мнению, будет намного легче.

Эта группа написала программу, которая перебирает все четвёрки натуральных чисел (a, b, c, n), таких что $n \geqslant 3$, в порядке увеличения максимума из этих чисел, а при равенстве максимумов — в лексикографическом порядке.

Таким образом, сначала будет перебрана четвёрка $(1, 1, 1, 3)$, затем четвёрка $(1, 1, 2, 3)$ и так далее. А, например, за четвёркой $(3, 3, 3, 3)$ будет следовать четвёрка $(1, 1, 1, 4)$.

Для каждой четвёрки программа сравнивает числа $a^n + b^n$ и c^n и выводит соответствующее неравенство: $a^n + b^n > c^n$ или $a^n + b^n < c^n$.

Теперь программисты хотят проверить своё доказательство. Поэтому они просят вас воспроизвести свою работу и вывести выписанные их программой неравенства с l-го по r-е, включительно.

Формат входных данных

В первой строке входных данных находятся два целых числа l и r ($1 \leqslant l \leqslant r \leqslant 10^{12}$; $r - l \leqslant 10^4$).

Формат выходных данных

Выведите часть доказательства, начиная с l-го неравенства и заканчивая r-м, каждое на отдельной строке. Для обозначения возведения в степень используйте символ карет («^», символ таблицы ASCII с кодом 94). Не выводите пробелы.

Пример

стандартный ввод	стандартный вывод
1 4	1^3+1^3>1^3 1^3+1^3<2^3 1^3+1^3<3^3 1^3+2^3>1^3

Задача D. Угадай путь

Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мегабайт

Это интерактивная задача.

Вы готовитесь принять участие в соревнованиях по робототехнике. С помощью запусков робота на специальном поле с установленными на нём датчиками вам предстоит определить один из кратчайших путей по этому полю, который загадало жюри.

Поле, на котором предстоит запускать робота, имеет вид таблицы с m строками и n столбцами. Будем обозначать как (i, j) клетку, лежащую в i-й строке и j-м столбце ($1 \leqslant i \leqslant m, 1 \leqslant j \leqslant n$). Любой путь, по которому можно запустить робота, должен начинаться в левой верхней клетке, имеющей координаты $(1, 1)$, проходить по некоторым клеткам поля и заканчиваться в правой нижней клетке, имеющей координаты (m, n).

За один шаг робот может переместиться на одну клетку вниз или на одну клетку вправо. Таким образом, если робот находится в клетке с координатами (i, j), то он может переместиться в клетку с координатами $(i + 1, j)$ или в клетку с координатами $(i, j + 1)$. Робот не может выходить за границы поля. Путь робота состоит из $m + n - 2$ шагов, при выполнении которых робот перемещается из клетки с координатами $(1, 1)$ в клетку с координатами (m, n), посещая при этом некоторые клетки поля.

Жюри соревнований загадало некоторый путь, по которому должен проехать робот. Этот путь неизвестен участникам. В каждой клетке, по которой проходит этот путь, был установлен специальный датчик. Когда робот оказывается в клетке, где установлен датчик, датчик фиксирует появление там робота.

Участник может несколько раз запустить своего робота вдоль любого корректного пути из клетки $(1, 1)$ в клетку (m, n). После каждого запуска жюри сообщает в каких клетках датчики зафиксировали появление робота. Задачей участника является, запустив робота не более 10 раз, определить путь, который был загадан жюри.

Протокол взаимодействия

Сначала на вход вашей программе подается два целых числа m и n — размеры поля, на котором запускается робот ($1 \leqslant m, n \leqslant 1000$, $m + n > 2$).

После этого программа может делать запросы, каждый запрос соответствует запуску робота.

Для выполнения запроса необходимо вывести отдельной строке «`?` s», где s — строка, задающая путь, по которому должен проехать робот. Эта строка должна иметь длину $n + m - 2$ и состоять из символов «`D`» и «`R`». Если i-й символ строки равен «`D`», то на i-м шаге робот перемещается вниз, а если i-й символ строки равен «`R`», то вправо. Робот начинает путь в клетке с координатами $(1, 1)$, после исполнения всех шагов он должен оказаться в клетке с координатами (m, n).

В ответ на запрос программа жюри выводит целое число t — количество клеток, в которых датчики зафиксировали прохождение робота ($2 \leqslant t \leqslant m + n - 1$). В следующих t строках она выводит по два целых числа r_i, c_i — координаты клеток (r_i, c_i), в которых датчики зафиксировали прохождение робота (для всех $1 \leqslant i \leqslant t$ выполнено $1 \leqslant r_i \leqslant m$, $1 \leqslant c_i \leqslant n$). Гарантируется, что все t клеток различны и выведены по возрастанию r_i, а при равенстве r_i по возрастанию c_i.

Определив корректный путь, ваша программа должна вывести строку «`!` s», где s — строка, задающая в описанном выше формате загаданный жюри путь.

Если ваша программа сделает более 10 запросов запуска робота, задаст некорректный запрос или неверно угадает путь, она получит вердикт «`Wrong Answer`».

Пример

стандартный ввод	стандартный вывод
3 4 3 1 1 3 3 3 4 4 1 1 2 2 2 3 3 4	 ? DDRRR ? DRRRD ! RDRDR

Замечание

После каждого действия вашей программы выводите перевод строки. После каждого действия вашей программы делайте сброс потока вывода.

Если вы используете «`writeln`» в Паскале, «`cout << ... << endl`» в C++, «`System.out.println`» в Java, «`print`» в Python, «`Console.WriteLine`» в C#, то сброс потока вывода у вас происходит автоматически, дополнительно ничего делать не требуется. Если вы используете другой способ вывода, рекомендуется делать сброс потока вывода. Обратите внимание, что перевод строки надо выводить в любом случае.

Пояснение к примеру

На рисунке 1 серым цветом закрашены клетки пути, загаданного жюри соревнований в первом тесте. Строка, задающая этот путь выглядит как «`RDRDR`».

(1,1)	(1,2)	(1,3)	(1,4)
(2,1)	(2,2)	(2,3)	(2,4)
(3,1)	(3,2)	(3,3)	(3,4)

Рис. 1. Загаданный путь «`RDRDR`».

На рисунке 2 жирной рамкой показаны клетки пути «`DDRRR`». При этом серым цветом закрашены клетки, в которых срабатывает датчик — клетки, лежащие на загаданном жюри пути, не закрашены клетки, в которых датчика нет.

На рисунке 3 жирной рамкой показаны клетки пути «`DRRRD`». При этом серым цветом закрашены клетки, в которых срабатывает датчик — клетки, лежащие на загаданном жюри пути, не закрашены клетки, в которых датчика нет.

(1,1)	(1,2)	(1,3)	(1,4)
(2,1)	(2,2)	(2,3)	(2,4)
(3,1)	(3,2)	(3,3)	(3,4)

Рис. 2. Запрос «`DDRRR`».

(1,1)	(1,2)	(1,3)	(1,4)
(2,1)	(2,2)	(2,3)	(2,4)
(3,1)	(3,2)	(3,3)	(3,4)

Рис. 3. Запрос «`DRRRD`».

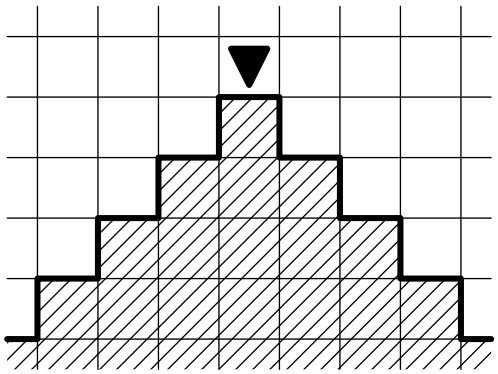
Задача E. Робо-прятки

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Миша занимается конструированием роботов для игры в робо-прятки. Игра в робо-прятки происходит на клетчатом поле $m \times n$, в некоторых клетках которого расположены роботы.

Каждый робот изначально смотрит в одном из четырех направлений: вверх, вниз, влево или вправо. У каждого робота есть его поле зрения. Рассмотрим робота, смотрящего вниз. В его поле зрения находится одна клетка, расположенная в следующей строке, три клетки, расположенные в строке через одну, пять — в строке через две, и так далее. В k-й из следующих строк в поле зрения робота находятся $2k - 1$ клеток, образующих отрезок, центром которого является клетка, расположенная в том же столбце, что и робот.

Области зрения роботов, смотрящих вправо, вверх и влево определяются аналогично.



Робот A видит робота B, если клетка, в которой расположен робот B, находится в поле зрения робота A. Для того, чтобы начать игру в робо-прятки, необходимо сделать так, чтобы на поле не существовало пары роботов, которые видят друг друга. Иначе говоря, для любой пары роботов A, B должно выполняться хотя бы одно из двух условий: A не видит B или B не видит A.

Миша может за одну операцию повернуть любого робота на 90 градусов по или против часовой стрелки, поменяв соответствующим образом направление, в котором смотрит робот. Он хочет начать игру как можно скорее, поэтому хочет сделать минимальное количество операций, чтобы получить конфигурацию, при которой можно начать игру.

Помогите ему найти такую конфигурацию. Можно доказать, что для любого начального расположения роботов искомая конфигурация существует.

Формат входных данных

В первой строке ввода находятся два целых числа m и n — количество строк и столбцов на поле ($1 \leqslant n, m \leqslant 2\,000$).

Следующие m строк состоят каждая из n символов «`U`», «`D`», «`L`», «`R`» или «`.`». Символ «`U`» означает, что в клетке находится робот, смотрящий вверх, символ «`D`» — вниз, «`L`» — влево, «`R`» — вправо, а «`.`» — что в клетке нет робота.

Формат выходных данных

Выведите m строк по n символов в каждой — конфигурацию, в которой можно начать игру. Конфигурация должна получаться из заданной во входных данных за минимальное число операций поворота робота на 90 градусов. Если оптимальных ответов несколько, вы можете вывести любой.

Обратите внимание, что Миша может только поворачивать роботов, но не может перемещать, удалять или добавлять их. Следовательно в выведенном ответе роботы должны находится в тех и только тех клетках, в которых они находились во входных данных.

Примеры

стандартный ввод	стандартный вывод
2 3 RDL .U.	UDL .R.
2 2	.. ..

Задача F. Равнобедренные треугольники

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Дан правильный многоугольник с n вершинами.

Требуется найти количество равнобедренных треугольников, вершины которых являются вершинами многоугольника.

Формат входных данных

На вход подаётся одно целое число n — количество вершин правильного многоугольника ($3 \leqslant n \leqslant 10^9$).

Формат выходных данных

Выведите одно целое число — количество равнобедренных треугольников, вершины которых являются вершинами многоугольника.

Примеры

стандартный ввод	стандартный вывод
3	1
5	10

Замечание

Треугольник называется равнобедренным, если у него есть хотя бы две равные стороны.

Задача G. Слишком много минусов

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Различные статьи, научные тексты, и даже условия к задачам олимпиад чаще всего создаются с использованием систем вёрстки TEX и LATEX, исходно разработанных Дональдом Кнутом и Лэсли Лампортом.

В этих системах используются специальные макросы, которые позволяют вводить специальные символы, которых нет на клавиатуре. Например, если написать несколько минусов подряд, то они преобразуются системой в тире разной длины. Из-за этого, когда необходимо получить в документе строку из нескольких минусов подряд, приходится идти на специальные ухищрения.

Для того, чтобы группа подряд идущих минусов не воспринималась как макрос, используются фигурные скобки. По аналогии с многими языками программирования, например, C++, фигурные скобки используются в TEX для группировки блоков символов. Последовательность фигурных скобок должна образовывать правильную скобочную последовательность. А именно, если удалить из строки все символы кроме фигурных скобок, а затем заменить открывающую фигурную скобку «`{`» на $+1$, а закрывающую «`}`» на -1, то сумма всех элементов получившейся последовательности должна быть равна 0, а сумма любого префикса получившейся последовательности должна быть неотрицательна.

Чтобы последовательность минусов не заменялась на тире, между любыми двумя подряд идущими минусами должна располагаться хотя бы одна фигурная скобка.

Отметим, что группа подряд идущих плюсов, в отличие от минусов, не соответствует никакому макросу и никаких условий на два подряд идущих плюса не накладывается.

Рассмотрим строку s, состоящую из плюсов и минусов. Необходимо добавить в неё минимальное количество фигурных скобок с соблюдением описанных правил, чтобы никакие два минуса не шли подряд. Будем называть полученные строки *защищёнными* строками для s.

Например, для строки «`++--`» оптимальными являются пять защищённых строк: «`++-{-}`», «`++-{}-`», «`++{-}-`», «`+{+-}-`», «`{++-}-`».

Строки выше перечислены в *лексикографическом порядке*: они упорядочены по первому символу, при равном первом символе — по второму, и так далее, с учётом порядка «`+`» < «`-`» < «`{`» < «`}`».

Рассмотрим все минимальные по длине строки, которые могут быть получены из s добавлением фигурных скобок, с соблюдением описанных правил. Требуется найти и вывести k-ю в лексикографическом порядке из этих строк, или сказать, что k слишком велико.

Формат входных данных

В первой строке ввода задана строка s, состоящая из плюсов и минусов, s непуста и имеет длину не более 60.

Во второй строке задано целое число k ($1 \leqslant k \leqslant 10^{18}$).

Формат выходных данных

Требуется вывести k-ю в лексикографическом порядке минимальную по длине защищённую строку для заданной во входном файле строки s.

Если k больше количества минимальных защищенных строк, следует вывести «`Overflow`».

Примеры

стандартный ввод	стандартный вывод
`++--` `2`	`++-{}-`
`-+-+-` `2`	`Overflow`

Задача H. Девятая планета

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Несколько лет назад была выдвинута гипотеза о существовании девятой планеты Солнечной системы, которая объясняет некоторые нестыковки в орбитах удалённых астероидов и малых небесных тел. Для того, чтобы проверить эту гипотезу, ученые сконструировали специальный прибор, наблюдающий за небосводом, и направили его в точку, которую предположительно должна пересекать траектория планеты.

У учёных не слишком большой бюджет, поэтому прибор имеет только один целочисленный регистр. Каждый раз, когда указанную точку пересекает какое-либо небесное тело, значение регистра увеличивается на девять. Из-за особенностей конструкции прибора первая цифра десятичной записи числа в регистре может исчезнуть, если она равна единице. При этом если в регистре находится число 1, то единственная единица в его десятичной записи также может исчезнуть, после чего в регистре оказывается число 0.

Назовём прибавление одной или нескольких девяток событием первого типа, а исчезновение одной или нескольких единиц — событием второго типа.

Когда в регистре прибора находилось число a, учёные ушли на обед, а, вернувшись, увидели в регистре число b. Помогите им восстановить процесс, происходящий во время их отсутствия. Обед длился недолго, так что с прибором не могло произойти суммарно более 1000 событий.

Если есть несколько возможных последовательностей событий, которые могли привести к тому, что число a в приборе преобразовалось в число b, то учёных устроит любая их них.

Формат входных данных

В единственной строке ввода заданы два целых неотрицательных числа a и b ($0 \leqslant a, b \leqslant 10^9$).

Формат выходных данных

Если прибор неисправен, и из a невозможно получить b, требуется вывести «`Broken`» в единственной строке вывода.

Иначе первой строкой вывода должно быть слово «`Stable`». В следующих строках необходимо вывести описание способа получения из числа a из числа b.

Во второй строке вывода должно находиться единственное целое число n — количество событий, которые произошли с прибором ($0 \leqslant n \leqslant 1000$). Обратите внимание, что вам не требуется минимизировать число n.

Каждая из следующих n строк должна описывать одно событие.

- Строка «`+` x», где $x > 0$ означает, что прибор зафиксировал x небесных тел, а к текущему числу прибавилось $9x$.
- Строка «`-` y», где $y > 0$ означает, что на приборе исчезли первые y цифр в десятичной записи текущего числа; все эти цифры должны быть равны 1.

После последовательного применения описанных n операций к числу a должно получиться число b. Промежуточные числа, которые получаются после применения каждой из операций, не должны превосходить 10^{18}.

Примеры

стандартный ввод	стандартный вывод
0 0	Stable 0
1 9	Stable 2 + 2 - 1

Задача I. Даты

Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

Одной из проблем обработки дат в тексте является то, что есть несколько общепринятых форматов записи дат.

Петя на работе получил задание написать программу для автоматической обработки большого массива текстов. Часть текстов использует принятый в России и Европе формат записи дат: «день.месяц.год». Другая использует американский формат «месяц/день/год».

Здесь год представляет собой число от 1 до 9999, возможно, дополненный ведущими нулями до 2, 3 или 4 цифр, месяц — число от 1 до 12, возможно, дополненное ведущим нулём до 2 цифр, день — число от 1 до 31, возможно, дополненное ведущим нулём до 2 цифр. При этом дата не обязательно корректна, ведь в тексте может быть, например, подобное объяснение: «так как 2001 год не является високосным, 29.02.2001 — некорректная дата».

Петя выделил в тексте даты и для удобства дальнейшей обработки хочет привести их к единому формату. Поскольку он не знает, в каком формате будет удобнее передать даты следующему обработчику, он хочет вывести каждую дату в обоих форматах, для единообразия, при необходимости, дополнив день и месяц до 2 цифр, а год — до 4 цифр ведущими нулями.

Задан массив полученных Петей строк, задающих даты. Для каждой даты выведите её сначала в формате «`DD.MM.YYYY`», а затем в формате «`MM/DD/YYYY`».

Формат входных данных

В первой строке дано одно число n ($1 \leqslant n \leqslant 20\,000$).

В следующих строках даны даты. Каждая строка содержит дату либо в формате «день.месяц.год», либо в формате «месяц/день/год».

Формат выходных данных

Выведите n строк. В каждой строке выведите два представления очередной даты: сначала в формате «`DD.MM.YYYY`», а затем через пробел в формате «`MM/DD/YYYY`».

Примеры

стандартный ввод	стандартный вывод
2 11.12.2000 1.2.1	11.12.2000 12/11/2000 01.02.0001 02/01/0001
2 20.10.2100 1/29/3000	20.10.2100 10/20/2100 29.01.3000 01/29/3000

Задача J. На заводе

Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Есть подозрение, что военный завод на окраине города причастен к серии недавних преступлений. Бюро общественной безопасности направило инспектора Акане выяснить, так ли это.

У Акане есть прямоугольная карта завода, являющаяся таблицей $m \times n$. Каждая ячейка карты или является пустой, или содержит один из цехов завода. Разумеется, все цеха завода клеточно связаны, иначе говоря, от любого цеха до любого другого можно добраться через промежуточные цеха, переходя между цехами только по общей стороне. Также гарантируется, что завод не огораживает территорию, не являющуюся заводом, то есть от любой ячейки, не являющейся территорией завода, можно добраться до границы карты, переходя между ячейками, не являющимися цехами завода, по общей стороне.

Акане предполагает, что если на заводе и есть какие-то улики, то искать их в центре какого-либо цеха бесполезно. Гораздо вероятнее, что они находятся в углу цеха. Для упомянутой карты рассмотрим все узлы сетки, являющиеся углом хотя бы одного цеха. Акане называет такие узлы важными. Количество важных узлов, очевидно, не превосходит $(m + 1) \cdot (n + 1)$. Акане выяснила, что для каждой клетки с цехом, по её краям проложены коридоры, которые соединяют соседние узлы сетки, расположенные в углах этой клетки.

Акане хочет обойти все важные узлы, причём сделать это как можно более эффективно. А именно, Акане хочет составить маршрут по узлам сетки, такой что:

- маршрут начинается и заканчивается в одном и том же узле;
- между каждыми двумя узлами Акане переходит по коридорам;
- каждый важный узел посещается хотя бы один раз;
- маршрут проходит только по важным узлам;
- по каждому коридору Акане проходит не более одного раза.

Помогите Акане найти любой подходящий маршрут, или определите, что таковых не существует. Обратите внимание, что минимизировать длину маршрута не требуется.

Формат входных данных

Первая строка содержит числа m и n ($1 \leqslant m, n \leqslant 20$) — размеры карты с заводом.

Следующие m строк содержат по n символов и задают карту завода. Символ «`*`» обозначает цех, а «`.`» — пустое место на карте.

Формат выходных данных

Если обойти завод нельзя, выведите «`No`».

Иначе выведите «`Yes`», затем выведите количество коридоров, по которым пройдёт Акане в найденном маршруте.

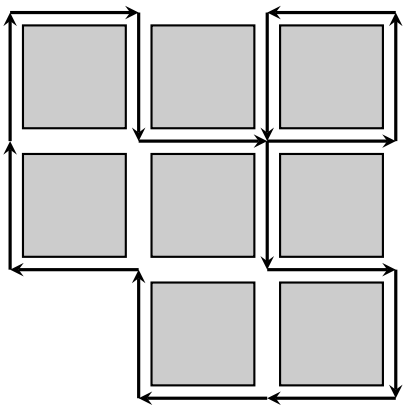
Затем выведите сам маршрут, по одному узлу в строке. Пронумеруем линии сетки, разделяющие ячейки карты, горизонтальные от 0 до m сверху вниз, вертикальные от 0 до n слева направо. Каждый узел описывается двумя числами: r_i и c_i ($0 \leqslant r_i \leqslant m$, $0 \leqslant c_i \leqslant n$) — номерами линий сетки, на пересечении которых он расположен.

Примеры

стандартный ввод	стандартный вывод
3 3 *** *** .**	Yes 16 0 0 0 1 1 1 1 2 1 3 0 3 0 2 1 2 2 2 2 3 3 3 3 2 3 1 2 1 2 0 1 0
1 4 ****	Yes 10 0 0 0 1 0 2 0 3 0 4 1 4 1 3 1 2 1 1 1 0
2 2 ** **	No

Замечание

Маршрут Акане в первом примере показан на рисунке ниже.



Задача K. Почти сортировка вращением

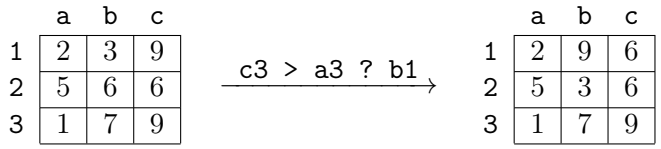
Ограничение по времени: 1 секунда
Ограничение по памяти: 512 мегабайт

В квадрате $n \times n$, заполненном числами, робот исполняет последовательность команд. Единственный тип команд, который умеет исполнять робот, таков:

```
[клетка 1] > [клетка 2] ? [клетка 3]
```

Такая команда исполняет следующее действие: если значение в клетке 1 больше чем значение в клетке 2, то поверни квадрат 2×2 с левым верхним углом в клетке 3 на 90 градусов против часовой стрелки.

Каждая клетка задаётся буквой, обозначающей столбец, и цифрой, обозначающей строку, записанными без пробела. Столбцы пронумерованы слева направо, начиная с a, строки пронумерованы сверху вниз, начиная с 1.



Пример применения одной команды.
Условие выполнено ($9 > 1$), поэтому поворот выполняется.

Если в одной из команд одна из клеток не входит в квадрат $n \times n$, а также если клетка 3 находится в нижней строке или в правом столбце квадрата $n \times n$, то робот ломается.

После окончания работы программы нижние $n - 2$ строки в порядке сверху вниз выписываются друг за другом (3-я строка, 4-я строка, ..., n-я строка). Полученный массив из $n \cdot (n - 2)$ чисел должен быть отсортированным по неубыванию.

Вам дано число n. Выведите программу только из инструкций описанного типа, исполнив которую на любом начальном наборе значений, робот, во-первых, не ломается, а во-вторых, добивается описанного состояния квадрата с числами.

Формат входных данных

В первой строке содержится целое число n ($2 \leqslant n \leqslant 9$) — размер квадратного поля.

Формат выходных данных

Выведите корректную программу. Она должна состоять из не более чем 100 000 команд. Следуйте формату, показанному в примере.

Пример

стандартный ввод	стандартный вывод
2	a2 > b2 ? a1 a2 > b2 ? a1 a2 > b2 ? a1

Замечание

Чтобы не подсказывать участникам, пример решения для $n \geqslant 3$ не приводится.

Для $n = 2$ достаточно добиться 0 отсортированных строк, поэтому любая синтаксически корректная программа является правильным решением.

Однако приведённая программа добивается не нуля, а целой одной отсортированной строки в нижней части квадрата: она поворачивает единственный квадрат 2×2 против часовой стрелки до тех пор, пока его нижняя строка не станет отсортированной по неубыванию. Несложно доказать, что за 0, 1, 2 или 3 поворота такая ситуация точно наступит.

Задача L. Путешествия во времени

Ограничение по времени: 2 секунды
Ограничение по памяти: 512 мегабайт

В будущем, после изобретения машины времени, изучать историю стало намного проще, ведь можно просто переместиться в нужное прошлое и посмотреть, как все было на самом деле! Этим изобретением активно пользуется Василий Андреевич, профессор университета, изучающий устройство дорог Берляндии на протяжении Великой Дорожной Реформы.

В течение k последовательных лет из-за Великой Дорожной Реформы, проходившей в Берляндии, её система дорог регулярно менялась. В этой стране n городов, пронумерованных для удобства целыми числами от 1 до n. В течение каждого года эти города были соединены $n-1$ двусторонними дорогами, причем между любой парой городов существовал единственный путь по дорогам, не проходящий через один и тот же город дважды.

Каждый рабочий день Василий Андреевич выбирал пару городов s, f, перемещался по очереди в каждый год Великой Дорожной Реформы и совершал в нём путешествие из города s в город f, посещая при этом все города по пути, включая s и f. После этого он выписывал количество городов, которые были общими во всех k путешествиях в этот день.

К сожалению, ровно в тот день, когда он закончил изучение Великой Дорожной Реформы, совершив путешествие для каждой возможной пары городов s и f, все записи историка потерялись. Остались карты системы дорог Берляндии в каждый из k лет Великой Дорожной Реформы.

Помогите Василию Андреевичу и восстановите числа, которые бы он выписал для всех возможных пар городов s и f.

Формат входных данных

В первой строке находятся два целых числа n и k — количество городов Берляндии и количество лет, изучаемых Василием Андреевичем ($1 \leqslant n, k \leqslant 500$).

Далее следует k описаний систем дорог Берляндии в различные года в следующем формате: в $n-1$ строке очередного описания находится два целых числа a и b — номера городов, соединенных очередной дорогой ($1 \leqslant a, b \leqslant n$, $a \neq b$).

Гарантируется, что в каждом из k изучаемых лет в Берляндии от любого города до любого существовал единственный путь, не проходящий через один и тот же город дважды.

Формат выходных данных

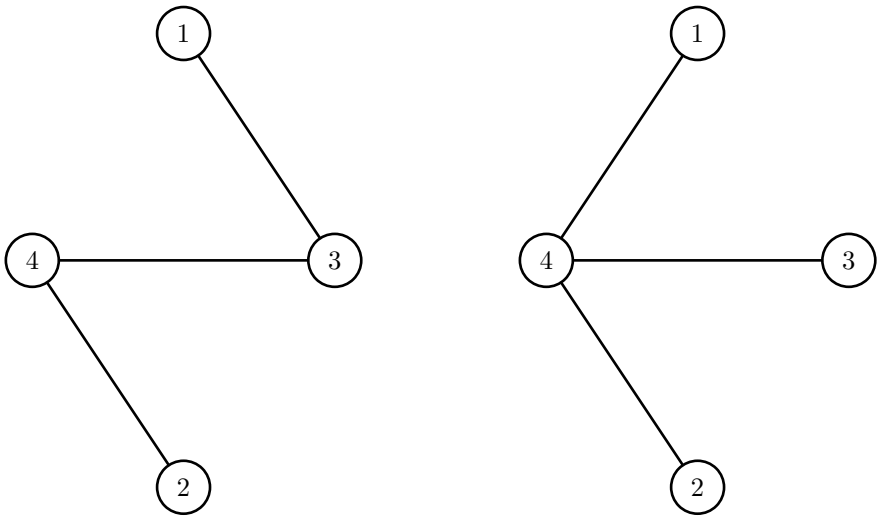
Выведите n строк по n чисел в каждой. В i-й строке j-е число должно быть равно числу, которое выпишет Василий Андреевич, если $s = i$ и $f = j$.

Примеры

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

Замечание

В первом тесте в Берляндии 4 города. Всего есть 2 года, в которые перемещался Василий Андреевич. Карта дорожной системы в первом году нарисована слева, карта дорожной системы во второй год нарисована справа.



Рассмотрим, например, пару городов $s = 1$ и $f = 2$. Переместившись в первый год, Василий Андреевич посетит города 1, 2, 3, 4. Переместившись во второй год, Василий Андреевич посетит города 1, 2, 4. Таким образом, он посетит 3 города 1, 2, 4 во всех путешествиях и выпишет число 3 для этой пары городов.

Рассмотрим пару городов $s = 3$ и $f = 1$. Переместившись в первый год, Василий Андреевич посетит города 1, 3. Переместившись во второй год, Василий Андреевич посетит города 1, 3, 4. Таким образом, он посетит 2 города 1, 3 во всех путешествиях и выпишет число 2 для этой пары городов.