Problem D. Lines

Input file:	`lines.in`
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

На плоскости даны n прямых. Ваша задача — выбрать наибольшее возможное количество прямых такое, что среди них никакие две не совпадают, никакие две не параллельны и никакие две не пересекаются в точке с $x = 0$.

Input

Первая строка входного файла содержит одно целое число T — количество тестовых примеров.

Каждый тестовый пример начинается строкой, содержащей целое число n — количество строк ($1 \le n \le 3000$). Каждая из последующих n строк содержит по три целых числа A, B и C, задающих прямую как множество точек (x, y), для которых $Ax + By + C = 0$ ($-10^9 \le A, B, C \le 10^9$, $A^2 + B^2 > 0$).

Гарантируется, что сумма всех n во входном файле не превосходит 3000.

Output

Для каждого тестового примера сначала выведите в отдельной строке целое число k — наибольшее количество прямых, которые могут быть выбраны. В следующей строке выведитеk целых чисел — номера выбранных прямых (прямые занумерованы с единицы в порядке их следования во входном файле).

Если ответов несколько, выведите любой из них. Номера прямых можно выводить в произвольном порядке.

Example

lines.in	standard output
2 2 1 1 0 1 1 1 2 -1 1 1 -2 1 1	1 1 1 1

Problem E. Yet Another Problem About Permutations

Input file:	`permutation.in`
Output file:	*standard output*
Time limit:	3 seconds
Memory limit:	256 mebibytes

Эта задача посвящена перестановкам. Если Вам неизвестны некоторые обозначения, указанные ниже, прочитайте пояснение к примеру.

Перестановка p называется *простой*, если длина любого её цикла не превосходит двух. Например, перестановка $2, 1, 4, 3$ является *простой*, в то время как перестановка $3, 1, 2$ таковой не является.

Вам дана перестановка p. Требуется представить её в виде произведения минимального количества простых перестановок.

Input

Первая строка входного файла содержит одно целое число T — количество тестовых примеров ($1 \le T \le 10^5$).

Каждая из последующих T строк задаёт один тестовый пример. Описание каждого тестового пример начинается целым числом n — длиной перестановки p ($1 \le n \le 10^5$), за которым идёт n попарно различных целых чисел $p_1, p_2, \ldots, p_n$ — сама перестановка p. ($1 \le p_i \le n$)

Сумма длин всех перестановок в одном тесте не превосходит 10^6.

Output

Для каждого тестового примера сначала выведите одно целое число k — минимальное количество простых перестановок в произведении. Следующие k строк должны содержать описания простых перестановок $q^{(1)}$, $q^{(2)}$, ..., $q^{(k)}$, по одной перестановке на строку. i-я из этих строк должна содержать n попарно различных целых чисел от 1 до n, задающих перестановку $q^{(i)}$. Произведение $q^{(1)} \circ q^{(2)} \circ \ldots \circ q^{(k)}$ должно быть равно p.

Если задача допускает несколько решений, выведите любое из них.

Example

permutation.in	standard output
2 4 2 1 4 3 3 3 1 2	1 2 1 4 3 2 3 2 1 1 3 2

Note

Перестановкой длины n называется последовательность из n целых чисел такая, что каждое из чисел от 1 до n встречается в этой последовательности ровно один раз.

Цикл в перестановке p — это последовательность попарно различных целых чисел $i_1, i_2, \ldots, i_t$, каждое из которых находится в диапазоне от 1 до n таких, что $p_{i_1} = i_2$, $p_{i_2} = i_3$, ..., $p_{i_{t-1}} = i_t$ и $p_{i_t} = i_1$. Число $t \ge 1$ называется *длиной* цикла.

Произведение $a \circ b$ двух перестановок a и b — это такая перестановка c, что для каждого i, $c_i = a_{b_i}$. Например, если $a = 3\,2\,1$ and $b = 1\,3\,2$, то произведение $a \circ b = 3\,1\,2$. Произведение трёх или более перестановок не зависит от порядка выполнения операций, то есть $a \circ b \circ c = (a \circ b) \circ c = a \circ (b \circ c)$.

Problem G. Shortest Accepted Word

Input file:	`shortest-accepted.in`
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

В этой задаче будут рассматриваться строки, состоящие из строчных латинских букв "`a`", "`b`" и "`c`".

Также *регулярное выражение* в данной задаче определяется рекурсивно следующим образом:

1. Строка из одного символа "`$`" является регулярным выражением, которому соответствует пустая строка.
2. Строки из одного символа "`a`", "`b`" и "`c`" являются регулярными выражениями, которым соответствуют строки "`a`", "`b`" и "`c`" соответственно.
3. Если P — регулярное выражение, то "`(`P`)`" — это также регулярное выражение, которому соответствуют все строки, соответствующие выражению P.
4. Если P — регулярное выражение, которое является **непосредственным** результатом применения правил 1–4, его *итерация*, обозначаемая как "P`*`", также является регулярным выражением, которому соответствуют строки типа $s = u_1u_2 \dots u_k$ (конкатенация некоторого, возможно пустого, множества строк), где k — любое неотрицательное число и каждая строка u_i соответствует регулярному выражению P.
5. Если P и Q — регулярные выражения, которые являются **непосредственным** результатом применения правил 1–5, их *конкатенация*, обозначаемая как "PQ", также является регулярным выражением, которому соответствуют строки в форме $s = uv$ (конкатенация строк u и v, каждая из которых может быть пустой), где префикс u соответствует выражению P, а суффикс v — выражению Q.
6. Если P и Q — регулярные выражения, которые являются **непосредственным** результатом применения правил 1–6, их *объединение*, обозначаемое как "P`|`Q", также является регулярным выражением, которому соответствуют строки, соответствующие как P, так и Q.

Ограничения в правилах 4–6 введены для того, чтобы предотвратить неоднозначность и расставить приоритеты операций: при разборе регулярного выражения сначала надо раскрывать итерации, затем конкатенации, затем объединения. Скобки играют обычную роль при изменении приоритетов операций: например, регулярное выражение "`a(bac|ac*)`" соответствует строкам вида "`a`, за которой следует ((`b`, за ним `a`, за ним `c`) или (`a`, за которым следует один или более экземпляров `c`)".

По заданному регулярному выражению r найтите кратчайшую строку s, которая соответствует этому регулярному выражению. Если таких строк несколько, выведите лексикографически наименьшую.

Input

Первая строка входа содержит целое число T — количество тестовых примеров ($1 \le T \le 300$). Каждая из последующих T строк задаёт один тестовый пример. Описание тестового примера состоит из регулярного выражения r, являющегося непустой строкой длины не более 300 символов, построенной в соответствии с вышеописанными правилами.

Сумма длин всех регулярных выражений в одном тесте не превосходит 300.

Output

Для каждого тестового примера выведите наиболее короткую строку, которая соответствует заданному регулярному выражению r. Если таких строк более одной, выведите лексикографически наименьшую.

Если ответом является пустая строка, выведите вместо неё строку из одного символа "`$`".

Example

shortest-accepted.in	standard output
3 a ab|ac*(ca|cb) ((ab|ac)a)*	a ab $

Problem H. Work

Input file:	work.in
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Есть N сотрудников и M различных заданий для них. Вам дана матрица A, в которой $A_{i,j} = 1$, если i-й сотрудник может выполнить j-е задание; в противном случае $A_{i,j} = 0$. Сотруднику может быть поручено задание только в том случае, если он его может выполнить.

Требуется распределить задания по сотрудникам таким образом, чтобы распределение количества работ, сделанных сотрудниками, было как можно ближе к равномерному (в евклидовой метрике), то есть чтобы N-мерный вектор, i-й элемент которого является количеством заданий, назначенное i-му сотруднику, был как можно ближе к N-мерному вектору, каждый элемент которого был равен вещественному числу M/N.

При этом каждое задание, которое может быть выполнено, должно быть поручено ровно одному сотруднику.

Input

Первая строка входного файла содержит два целых числа N и M ($1 \le N, M \le 300$). Далее следуют N строк, каждая из которых содержит по M символов. Каждый из этих символов является или '0', или '1'. Эти строки задают матрицу A.

Output

Выведите N строк; в i-й строке сначал выведите k_i — количество заданий, порученных i-му сотруднику. После этого выведите номера этих заданий.

Если оптимальных решений несколько, выведите любое из них.

Examples

work.in	standard output
3 3 111 111 111	1 1 1 2 1 3
2 4 0100 1100	1 2 1 1

Note

Евклидовым расстоянием между двумя векторами $(u_1, u_2, \ldots, u_N)$ и $(v_1, v_2, \ldots, v_N)$ является вещественное число

$$\sqrt{(v_1 - u_1)^2 + (v_2 - u_2)^2 + \ldots + (v_N - u_N)^2}.$$

Problem I. Jam

Input file:	jam.in
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Два программиста сидят в автомобиле, который едет за автобусом в пробке. На остановке два человека зашли в автобус, а три вышли. После чего один из программистов заметил, что если ещё один человек зайдёт в автобус, то автобус будет пустым.

Так как пробка была длинной, то программисты сделали несколько наблюдений, на каждой следующей остановке записывая количество вошедших и вышедших пассажиров.

Ваша задача — написать программу, которая вычислит наименьшее количество пассажиров в автобусе перед тем, как автомобиль программистов попал в пробку.

Input

Первая строка входного файла содержит одно целое число T ($1 \le T \le 50$) — количество тестовых примеров.

Каждый тестовый пример начинается строкой, содержащей целое число M ($1 \le M \le 100$) — количество остановок. Далее следует M строк, содержащих результаты наблюдений. Каждое наблюдение задано двумя целыми числами P_1 и P_2, разделёнными пробелом — количество человек, вошедших в автобус на соответствующей остановке, и количество человек, покинувших автобус, соответственно ($0 \le P_1, P_2 \le 1000$). При этом сначала P_1 человек входят в автобус, затем двери «на выход» открываются и P_2 человек выходят из него.

Output

Для каждого тестового примера выведите в отдельной строке минимальное количество пассажиров, находившихся в салоне автобуса до того, как программисты попали в пробку.

Example

jam.in	standard output
1 3 4 6 5 6 2 0	3

Problem J. King of Guess

Input file:	kingofguess.in
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Игра «Угадай число» играется следующим образом: загадано целое число N и заданы два целых числа X и Y, строго между которыми оно находится. Ход состоит в том, что участник называет некоторое целое число между X и Y, соответственно, ответы могут быть «больше», «меньше» и «угадал» (после ответа «угадал» игра заканчивается). В случае, если число не угадано, одно из чисел X или Y заменяется на названное число (в зависимости от ответа), и игра продолжается.

Пусть участники всякий раз называют среднее целое число в текущем интервале (если таких чисел два, они называют наименьшее). За какое количество ходов закончится игра?

Input

Первая строка входа содержит три целых числа N, X и Y ($0 \le N, X, Y \le 10^4$, $X < Y$).

Output

Выведите одно число — количество ходов, за которое закончится игра.

Examples

kingofguess.in	standard output
42 20 80	3

Problem K. Lesson

Input file:	`lesson.in`
Output file:	*`standard output`*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Алиса и Боб играют в «Морской бой» на уроке в школе. Игра идёт по следующим правилам:

У каждого игрока есть поле $N \times N$, на котором он располагает четыре неперекрывающихся корабля; каждый корабль является цепочкой клеток, параллельных одной из сторон поля. Требуется разместить по одному из следующих кораблей:

Name	**Length**
Sail	1
Frigate	2
Cruiser	3
Dreadnought	4

После этого игроки делают ходы по очереди, называя клетки на игровом поле. Если клетка, названная игроком X, занята на игровом поле игрока Y кораблём, игрок Y сообщает, что корабль ранен; если же игрок X своим ходом назвал последнюю из не названных им ранее клеток корабля, то игрок Y сообщает, что корабль потоплен, после чего игрок X получает дополнительный ход; если же игрок X своим ходом не потопил корабль игрока Y, то очередь хода переходит к игроку Y. Игра заканчивается, когда один из игроков потопил все корабли противника; в этом случае тот игрок, у которого остались корабли, объявляется победителем.

Чтобы учитель не заметил происходящего, Алиса и Боб модифицировали игру: сейчас после расстановки кораблей они записывают на бумаге последовательность ходов; разумеется, все ходы корректны и один и тот же игрок не записывает одно и то же поле дважды (ибо незачем).

По заданным ходам каждого игрока выведите, какие корабли и в каком порядке будут потоплены и кто в итоге выиграет, если первый ход делает Алиса.

Input

Первая строка входного файла содержит целое число T — количество тестовых примеров ($1 \le T \le 20$).

Далее следуют тестовые примеры. Каждый тестовый пример начинается строкой, содержащей целое число N ($4 \le N \le 10$) — размер игрового поля. Далее следуют N строк, задающих расстановку кораблей на поле Алисы. Каждая из этих строк состоит из 4 символов. Это один из пяти символов '`.`', '`1`', '`2`', '`3`' или '`4`'. Точка ('`.`') обозначает, что данная клетка пуста. '`1`', '`2`', '`3`' или '`4`' обозначают, что эта клетка занята одним из кораблей Алисы. Клетки с одинаковыми числами принадлежат одному кораблю; корабли параллельны сторонам поля. Обратите внимание, что цифра, обозначающая корабль, не обязана совпадать с его длиной.

Следующие N строк задают расстановку кораблей на поле Боба в аналогичном формате. Следующие $N \cdot N$ строк задают ходы Алисы. Каждая из строк содержит два целых числа R_i и C_i — строка и столбец, которые она планирует называть для этого хода (если до этого хода дойдёт очередь). Следующие $N \cdot N$ строк задают ходы Боба в аналогичном формате ($1 \le R_i, C_i \le N$).

Output

Для каждого тестового примера выведите для каждого потопленного корабля сообщение в формате "`PlayerX sank PlayerY's ShipName`", в порядке, в котором корабли были потоплены. Последняя строка вывода для каждого тестового примера должна содержать имя победителя.

Examples

lesson.in	standard output
1 4 1..4 22.4 3334 ...4 112. ..2. .32. 4444 1 1 1 2 1 4 2 3 3 2 4 3 2 1 3 3 4 1 4 2 3 4 4 4 1 3 2 2 2 4 3 1 1 3 4 4 3 4 1 1 4 3 4 1 2 4 2 2 1 2 4 2 3 1 3 2 2 3 1 4 3 3 2 1	Alice sank Bob's Frigate Alice sank Bob's Sail Bob sank Alice's Sail Alice sank Bob's Dreadnought Alice sank Bob's Cruiser Alice

Problem L. Maze

Input file:	`maze.in`
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

На некотором астероиде был найден лабиринт ровно с одним входом, не содержащий циклов и пустот, окружённых стенами. Для исследования лабиринта был отправлен робот.

Робот всегда смотрит в том направлении, в котором он движется. На каждом шаге робот будет пытаться повернуть направо. Если на этом месте стена, он будет пытаться сделать шаг вперёд. Если и это невозможно — пытаться повернуть налево, если же и это невозможно — развернуться назад.

Робод записывает лог, который содержит его путь от момента входа в лабиринт до момента выхода. Передвижения записаны единичными буквами: '`F`' для движения вперёд, '`L`' для движения налево, '`R`' для движения направо и '`B`' для движения назад.

Каждая из букв '`L`', '`R`' и '`B`' обозначает не только поворот робота, но и продвижение на одно поле в данном направлении. Изначально робот ориентирован на восток. Путь робота всегда заканчивается в точке выхода.

Вам задан лог. Требуется по нему восстановить план лабиринта.

Input

Первая строка входа содержит одно целое число T — количество тестовых примеров ($1 \le T \le 100$). Каждый тестовый пример представляет собой одну строку — корректный лог пути робота. Гарантируется, что исходный лабиринт имел размеры не более, чем 100×100.

Output

Для каждого тестового примера выведите в первой строке два целых числа h и w ($3 \le h, w \le 100$) — высоту и ширину лабиринта, соответственно. Далее выведите h строк, каждая из которых содержит w символов, задающих лабиринт, где '`X`' обозначает стену, а '`.`' — пустое поле.

Контур лабиринта должен состоять из стен, за исключением одного квадрата с левой стороны — входа в лабиринт. Лабиринт не должен содержать циклов (то есть путей, по которым робот сможет придти в какую-то клетку не с той стороны, с которой в неё входил) и пустых полей, которые не могут быть достигнуты от входа. Каждая строка и каждый столбец (кроме верхней и нижней строки и правого столбца) должны содержать как минимум одно пустое поле.

Example

maze.in	standard output
3 FFRBLF FFRFRBRFBFRBRFLF FRLFFFLBRFFFRFFFRFRFBRFLBRFRLFLFFR	4 4 XXXX ...X XX.X XXXX 7 5 XXXXX ...XX XX.XX X...X XX.XX XX.XX XXXXX 7 7 XXXXXXX X...X.X X.X...X X.X.XXX ..XXX.X X.....X XXXXXXX