Problem A. Bermutation

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

Дана перестановка первых n целых положительных чисел p. Разрешено модифицировать эту перестановку следующим образом: выбирается отрезок из $2b$ последовательных элементов, после чего его половины меняются местами.

Более формально, если вы выбираете отрезок $a_i, a_{i+1} \ldots, a_{i+2b-1}$, то после модификации получите отрезок $a_{i+b}, a_{i+b+1}, \ldots, a_{i+2b-1}, a_i, a_{i+1}, \ldots, a_{i+b-1}$.

Рассмотрим S — множество всех перестановок, которые могут быть получены из заданной перестановки p применением этой модификации некоторое (возможно, нулевое) количество раз.

Отрезок из $2b$ последовательных элементов для каждой модификации выбирается независимо от отрезков, выбранных для других модификаций. Запишем все эти перестановки в лексикографическом порядке и занумеруем их, начиная с единицы. Ваша задача — найти в этом списке номер изначальной перестановки p. Ответ выведите по модулю 120 586 241.

Input

Первая строка входного файла содержит одно целое положительное число T — количество тестовых примеров. Далее задаются тестовые примеры.

Каждый тестовый пример задан в двух строках. Первая строка кажжого тестового примера содержит два целых числа n и b ($2 \le n \le 10^5$, $1 \le b$ и $2b \le n$). Вторая строка каждого тестового примера содержит n целых чисел, задающих перестановку p. Каждое из целых положительных чисел встречается во второй строке ровно один раз.

Гарантируется, что сумма всех n во входном файле не превосходит 10^5.

Output

Выведите остаток от деления на 120 586 241 перестановки p в лексикографически упорядоченном списке всех перестановок, которые могут быть получены из перестановки p с помощью описанной в задаче модификации/

Example

bermutation.in	standard output
2 5 1 2 3 1 4 5 5 2 2 3 1 4 5	31 3

Problem B. Grid

Input file:	`grid.in`
Output file:	*standard output*
Time limit:	1.5 seconds
Memory limit:	256 mebibytes

В центре одного из байтландских городов находится прямоугольная площадь, замощённая квадратными плитками со стороной 1 таким образом, что её можно представить как прямоугольник $N \times M$. До недавнего времени в этом городе ходили трамваи; трамвайное движение было удобно большинству жителей. Однако участок, на котором расположено депо, по мнению одного из близких к мэру предпринимателей, отлично подойдёт под казино. Так что среди горожан была запущена кампания «Уберём трамваи с улиц»; правда, участники этой кампании получали плату фишками казино.

Финальной точкой этой кампании должен стать сход на главной площади. Все участники схода собрались на площади и встали таким образом, что на каждой плитке стоит ровно один человек. Каждый участник схода смотрит в одну из четырёх сторон — на север, на восток, на юг или на запад. При этом обнаружился некий эффект: если после начала схода два его участника смотрят друг на друга, им становится стыдно, так как оба знают, что ничего не имеют против трамвая, а пришли сюда за фишки; после чего они решают, что будут требовать убрать не трамвай, а мэра...

Организатор схода прекрасно понимает ситуацию и не хочет «подставить» своего покровителя. Он готов заплатить каждому участнику схода по дополнительной фишке за каждый поворот вокруг своей оси на 90 градусов. При этом если участник поворачивается несколько раз, ему каждый раз платится фишка. Организатор хочет добиться ситуации, когда никакие два участника схода не смотрят друг на друга.

Какое минимальное количество фишек он должен потратить?

Input

Первая строка входа содержит два целых числа N и M — размерность площади в плитках ($1 \le N, M \le 50$). Каждая из последующих N строк задаёт первоначальное расположение участников схода и содержит M символов. Каждый из этих символов может быть или "^" (участник смотрит на север), ">" (участник смотрит на восток), "v" (участник смотрит на юг), or "<" (участник смотрит на запад).

Output

Выведите минимальное количество фишек, которое организатор схода должен потратить, чтобы добиться расстановки, при которой никакие два участника схода не смотрят друг на друга.

Examples

grid.in	standard output
3 3 >v< >^< ^^^	2
1 10 »><«<»«	2

Problem C. Heap

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

A *d-арная куча* размера n — это массив $a_1, a_2, \ldots, a_n$, в котором для каждой пары индексов i и j такой, что $1 \le i \le n$, $1 \le j \le n$ и $(i-1) \cdot d+2 \le j \le i \cdot d+1$, выполняется строгое неравенство $a_j > a_i$. Например, для тернарной ($d = 3$) кучи размера 8 требуемыми неравенствами являются $a_2 > a_1$, $a_3 > a_1$, $a_4 > a_1$, $a_5 > a_2$, $a_6 > a_2$, $a_7 > a_2$, and $a_8 > a_3$.

Рассмотрим все d-арные кучи размера n, которые также являются перестановками чисел от 1 до n. Запишем их все в лексикографическом порядке, рассматривая перестановки как последовательности чисел. После этого, занумеруем кучи в получившемся упорядоченном списке, начиная с единицы.

Вам дана d-арная куча размера n, которая одновременно является перестановкой. Найдите её номер в построенном выше списке. Так как ответ может быть очень большим, вычислите его по модулю $(10^9 + 7)$.

Input

Первая строка входного файла содержит целое положительное число T: количество тестовых примеров. Далее следуют сами тестовые примеры.

Каждый тестовый пример состоит из двух строк. Первая из этих строк содержит два целых числа n и d ($1 \le n \le 3000$, $1 \le d \le 3000$).

Вторая строка содержит n попарно различных целых положительных чисел, не превосходящих n и задающих перестановку. При этом задаваемая строкой перестановка является одновременно d-арной кучей.

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

Output

Для каждого тестового примера, выведите остаток от деления номер соответствующей последовательности в описанном в условии списке d-арных куч, являющихся перестановками, на $(10^9 + 7)$.

Example

heap.in	standard output
2 5 2 1 3 2 4 5 5 3 1 4 3 2 5	7 12

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 F. Strange Sequence

Input file:	`sequence.in`
Output file:	*`standard output`*
Time limit:	6 seconds
Memory limit:	256 mebibytes

Рассмотрим следующую хорошо известную последовательность s, состоящую из «цифровых строк».

Пусть $s_0 =$ “2”. Каждый следующий элемент строится описанием предыдущего: разбиваем предыдущий на последовательные блоки равных цифр, и для каждой такой группы записываем размер группы, за которым идёт цифра, из которой состоит группа. Таким образом, первые несколько элементов строятся следующим образом:

String	Description
$s_0 =$ “2”	одна 2
$s_1 =$ “12”	одна 1, одна 2
$s_2 =$ “1112”	три 1, one 2
$s_3 =$ “3112”	одна 3, две 1, одна 2
$s_4 =$ “132112”	одна 1, одна 3, одна 2, две 1, одна 2
$s_5 =$ “1113122112”	...

Найдите остаток от деления длины n-го элемента этой последовательности на 7 340 033.

Input

Первая строка входа содержит одно целое число n ($0 \le n \le 10^{18}$).

Output

Выведите остаток от деления длины строки s_n на 7 340 033.

Examples

sequence.in	standard output
0	1
2	4

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

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