Задача A. Рассядьтесь удобно!

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

Знаете ли вы, что то, как Вы сидите во время соревнования, может серьезно повлиять на Ваши результаты?

В частности, положение стульев относительно столов и компьютера очень сильно влияет на взаимоотношения в команде.

С другой стороны, само расположение столов, за которыми сидят участники на больших соревнованиях, не менее важно. Как правило, они не должны стоять близко друг к другу, чтобы, с одной стороны, участники могли не сдерживать громкость голоса при рассказывании очередного оригинального решения, и, с другой стороны, не могли подслушать идеи других команд.

И если о первом должны заботиться участники, то о втором — организаторы соревнований.

Представьте себе, что Вам поручили организовать расстановку столов в одном из залов. План зала представляет собой лист клетчатой бумаги размером N на M, на котором каждая клетка или занята, или свободна. Между свободными клетками можно перемещаться, переходя по общей стороне (ходом ладьи). Зал устроен так, что из любой свободной клетки в любую другую свободную можно дойти, причем ровно одним способом.

Стол для команды занимает ровно две смежных клетки, причем они могут находиться друг относительно друга как по горизонтали, так и по вертикали. Учитывая вышеуказанные соображения, расстояние между столами не должно быть менее одной клетки (переходя по смежным клеткам).

Расставьте в свободном пространстве зала наибольшее число столов, соблюдая вышеуказанные требования.

Формат входного файла

В первой строке находятся два целых числа N, M ($1 \le N \cdot M \le 10^4$).

В каждой из последующих N строк находится по M символов. Вместе они составляют план зала. Каждый из этих символов является либо точкой «`.`», обозначающей свободную клетку, либо звездочкой «`*`», обозначающей занятую клетку.

Формат выходного файла

Выведите план расстановки столов в зале. Формат вывода аналогичен формату ввода (размеры выводить не надо), только к символам добавляются следующие:

- левая квадратная скобка («`[`»), обозначает левую половину стола.
- правая квадратная скобка («`]`»), обозначает правую половину стола.
- большая латинская буква «`M`», обозначает верхнюю половину стола.
- большая латинская буква «`W`», обозначает нижнюю половину стола.

Если ответ неоднозначен, выведите любой.

Примеры

sitdown.in	sitdown.out

`4 4`
`****`
`*..*`
`**.*`
`****`
	`****`
	`*.M*`
	`**W*`
	`****`

`1 5` `.....`	`[].[]`

Задача B. Операция Xor

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

Пусть есть два неотрицательных целых числа a и b. Представим их в двоичной системе счисления $a = (a_1 a_2 \ldots a_l)_2$ и $b = (b_1 b_2 \ldots a_l)_2$ $(a_i, b_i \in \{0, 1\})$. Без ограничения общности можно полагать, что они имеют одинаковую длину, так как к меньшему по длине можно подписать слева необходимое количество нулей.

Результат операции $c = a \oplus b$ — число, i-тая цифра двоичной записи которого равна $(a_i + b_i) \mod 2$. В программировании такая операция называется операцией **xor**.

Ваша задача найти набор из n чисел a_i, удовлетворяющих условиям:

1. $a_i \neq a_j$ ни для каких $i \neq j, 1 \leq i, j \leq n$
2. $0 < a_i \leq 2^{31} - 1$
3. $a_1 \oplus a_2 \oplus \ldots \oplus a_n = A$

Формат входного файла

Во входном файле содержаться два натуральных числа n $(1 \leq n \leq 100000)$ и A $(0 \leq A \leq 2^{31} - 1)$.

Формат выходного файла

В выходной файл через пробел запишите n чисел, удовлетворяющих вышеописанному условию или «-1», если такого набора не существует.

Пример

xor.in	xor.out
1 5	5
7 1	12 71 35 3 54 138 214

Задача C. Матрица минимальных разрезов

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

Рассмотрим взвешенный неориентированный граф G. *Минимальным реберным разрезом* между вершинами u и v называют множество ребер $C_{u,v}$ с минимальным весом, такое что любой путь из u в v содержит хотя бы одно ребро из $C_{u,v}$. Обозначим вес минимального реберного разреза между вершинами u и v как $c_{u,v}$.

Матрица $c_{u,v}$ называется *матрицей минимальных разрезов* графа G. Вам дана матрица $c_{u,v}$. Найдите граф G, для которого $c_{u,v}$ является матрицей минимальных разрезов, либо выясните, что такого графа нет.

Формат входного файла

Первая строка входного файла содержит n — размер матрицы ($2 \le n \le 50$). Следующие n строк содержат по n целых чисел — значения элементов матрицы. Гарантируется, что матрица симметрична и содержит нули на главной диагонали. Все остальные вхождения положительны и не превышают 1000.

Формат выходного файла

Если искомый граф G существует, выведите на первой строке выходного файла слово «`YES`». Затем выведите m — количество ребер в графе. После этого выведите m строк по три числа — описания ребер: вершины, которые соединены ребром и его вес. Граф не должен содержать параллельных ребер и петель. Вес каждого ребра не должен превышать 1000.

Если искомого графа не существует, выведите «`NO`» на первой строке выходного файла.

Пример

cut.in	cut.out
3 0 2 2 2 0 2 2 2 0	YES 3 1 2 1 1 3 1 2 3 1

Задача D. Умножение матриц

Имя входного файла:	`matrix.in`
Имя выходного файла:	`matrix.out`
Ограничение по времени:	3 секунды
Ограничение по памяти:	64 мегабайта

Петя учится умножать матрицы. Пусть нужно умножить две матрицы A и B размера $n \times n$. Результатом умножения будет матрица C такого же размера, элементы которой вычисляются по следующей формуле: $C_{ij} = \sum\limits_{k=1}^{n} A_{ik} \cdot B_{kj}$. В качестве тренировки, Петя перемножает *двоичные* матрицы — то есть матрицы из нулей и единиц, и выполняет все действия по модулю два.

Недавно Петя перемножил две матрицы A и B и утверждает, что результат равен C. Однако его сестра Ира ему не верит. Она говорит, что он ошибся.

Помогите детям разрешить спор, проверьте, действительно ли произведение матриц A и B равно C.

Формат входного файла

Первая строка входного файла содержит число n — размер матриц ($1 \le n \le 4000$). Следующие три строки описывают матрицы A, B и C.

Каждая матрица описывается строкой, которая содержит n блоков по $\lceil n/4 \rceil$ шестнадцатеричных цифр. Если выписать эти цифры в двоичной записи (от старших битов к младшим) и откинуть лишние цифры в конце, то получится соответствующая строка матрицы. Например, матрица

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

описывается блоками `28`, `A8`, `68`, `78`, `D0`, `88`.

Формат выходного файла

Выведите “`YES`”, если $AB = C$, и “`NO`’, если данное равенство не выполняется.

Пример

matrix.in	matrix.out
6 28 A8 68 78 D0 88 80 40 20 10 08 04 28 A8 68 78 D0 88	YES
6 28 A8 68 78 D0 88 80 40 20 10 08 04 28 B8 68 78 D0 88	NO