Задача A. Кубики

Имя входного файла:	`cubes.in`
Имя выходного файла:	`cubes.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Родители подарили Сене набор детских кубиков. Поскольку Сеня скоро пойдет в школу, они купили ему кубики с буквами. На каждой из шести граней каждого кубика написана буква.

Теперь Сеня хочет похвастаться перед старшей сестрой, что научился читать. Для этого он хочет сложить из кубиков ее имя. Но это оказалось довольно сложно сделать —- ведь разные буквы могут находиться на одном и том же кубике и тогда Сеня не сможет использовать обе буквы в слове. Правда одна и та же буква может встречаться на разных кубиках. Помогите Сене!

Дан набор кубиков и имя сестры. Выясните, можно ли выложить ее имя с помощью этих кубиков и если да, то в каком порядке следует выложить кубики.

Формат входных данных

На первой строке входного файла находится число n ($1 \leqslant n \leqslant 100$) — количество кубиков в наборе у Сени. На второй строке записано имя Сениной сестры —- слово, состоящие только из больших латинских букв, не длиннее 100 символов. Следующие n строк содержат по 6 букв (только большие латинские буквы), которые написаны на соответствующем кубике.

Формат выходных данных

На первой строке выходного файла выведите «`YES`», если выложить имя Сениной сестры данными кубиками можно, «`NO`» в противном случае.

Если ответ «`YES`», на второй строке выведите m различных чисел из диапазона от 1 до n, где m - количество букв в имени Сениной сестры, i-е число должно быть номером кубика, который следует положить на i-е место при составлении имени Сениной сестры. Кубики нумеруются с 1, в том порядке, в котором они заданы во входном файле. Если решений несколько, выведите любое. Разделяйте числа пробелами.

Примеры

cubes.in	cubes.out
4 ANN ANNNNN BCDEFG HIJKLM NOPQRS	NO
5 HELEN ABCDEF GHIJKL MNOPQL STUVWN EIUOZK	YES 2 5 3 1 4

Задача B. Эйлеров путь

Имя входного файла:	`euler.in`
Имя выходного файла:	`euler.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Дан неориентированный связный граф, не более трех вершин имеет нечетную степень. Требуется определить, существует ли в нем путь, проходящий по всем ребрам.

Если такой путь существует, необходимо его вывести.

Формат входных данных

Первая строка входного файла содержит натуральное число n — количество вершин графа ($1 \leqslant n \leqslant 100\,000$). Далее следуют n строк, задающих ребра. В i-й из этих строк находится число m_i — количество ребер, инцидентных вершине i. Далее следуют m_i натуральных чисел — номера вершин, в которые ведут ребро из i-й вершины.

Граф может содержать кратные ребра, но не содержит петель.

Граф содержит не более 300 000 ребер.

Формат выходных данных

Если решение существует, то в первую строку выходного файла выведите одно число k — количество ребер в искомом маршруте, а во вторую $k + 1$ число — номера вершин в порядке их посещения.

Если решений нет, выведите в выходной файл одно число `-1`.

Если решений несколько, выведите любое.

Примеры

euler.in	euler.out
4 2 2 2 4 1 4 3 1 2 2 4 2 3 2	5 1 2 3 4 2 1

Задача C. Замощение доминошками

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

Дано игровое поле размера $n \times m$, некоторые клетки которого уже замощены. Замостить свободные соседние клетки поля доминошкой размера 1×2 стоит a условных единиц, а замостить свободную клетку поля квадратиком размера 1×1 — b условных единиц.

Определите, какая минимальная сумма денег нужна, чтобы замостить всё поле.

Формат входных данных

Первая строка входного файла содержит 4 целых числа n, m, a, b ($1 \leqslant n, m \leqslant 100$, $|a| \leqslant 1\,000$, $|b| \leqslant 1\,000$). Каждая из последующих n строк содержит по m символов: символ '"."' (точка) обозначает занятую клетку поля, а символ '"*"' (звёздочка) — свободную.

Формат выходных данных

В выходной файл выведите одно число — минимальную сумму денег, имея которую можно замостить свободные клетки поля (и только их).

Пример

dominoes.in	dominoes.out
2 3 3 2 .** .*.	5

Задача D. Студентам — бесплатно!

Имя входного файла:	students-free.in
Имя выходного файла:	students-free.out
Ограничение по времени:	0.5 секунд
Ограничение по памяти:	64 мегабайта

Зал Большого галактического театра состоит из S рядов, по S мест в каждом ряду. Продажа билетов на каждый спектакль происходит по следующему принципу: первые $S^2 - N$ ценителей прекрасного приобретают билеты на любые места по их вкусу, а оставшиеся N кресел администрация бесплатно выделяет студентам, отдавая дань сложившимся традициям.

Во избежание обвинений в дискриминации по половому признаку, рассаживать студентов по этим N местам необходимо таким образом, чтобы:

- в каждом ряду количество девушек-студенток и количество юношей-студентов различалось бы не более чем на 1;
- на каждой "вертикали мест" (т. е. местах, имеющих один и тот же номер, но расположенных в разных рядах) количество девушек-студенток и количество юношей-студентов также различалось бы не более чем на 1.

Таким образом, после продажи билетов ценителям прекрасного билетёры должны распределить оставшиеся N кресел на женские и мужские с соблюдением этих правил. Каждое место в зале определяется двумя числами от 1 до S — номером ряда и номером самого места в этом ряду. Студенческое кресло номер i расположено в a_i-м ряду и имеет в нём номер b_i. Поскольку ценители прекрасного могли занять совершенно любые места, числа a_i и b_i могут принимать любые значения от 1 до S. В частности, может оказаться так, что в каком-нибудь ряду не будет ни одного студенческого места.

Ради упрощения работы билетёров администрация обращается к вам с заданием написать программу, которая автоматизирует процесс распределения студенческих мест на мужские и женские.

Формат входных данных

Сначала вводятся два целых числа S и N ($1 \leqslant S \leqslant 100\,000, 1 \leqslant N \leqslant \min\{100\,000 S^2\}$). Далее расположены N пар натуральных чисел (a_i, b_i), не превосходящих S. Гарантируется, что все места различные.

Формат выходных данных

Если искомого способа не существует, выведите Impossible. Иначе выведите единственную строку из N символов M (мужское) и W (женское). Символ на i-й позиции соответствует статусу i-го места в той же нумерации, в которой они были перечислены во входных данных.

Примеры

students-free.in	students-free.out
2 2 2 1 1 2	WM
3 5 1 2 2 3 1 3 2 1 1 1	MMWWM

Задача E. Встреча лисичек

Имя входного файла:	`fox-meeting.in`
Имя выходного файла:	`fox-meeting.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

В стране Лисляндии N городов, соединённых дорогами. Дорожная сеть Лисляндии является деревом.

В Лисляндии в разных городах живёт M лисичек. Лисички любят ездить в гости друг к другу. Однако, лисички — очень осторожные зверьки. Лисичка отправляется в гости к другой лисичке только в том случае, если во всех городах на её пути также живут лисички.

Помогите некоторым лисичкам переехать в новые города так, чтобы каждая пара лисичек могла ездить друг к другу в гости. Так как лисички не любят переезжать далеко, то требуется найти такой план переезда, в котором максимальное из расстояний, которые проедут лисички во время переезда, было как можно меньше.

После переезда в одном городе может жить более одной лисички.

Формат входных данных

В первой строке содержится целое число N — количество городов ($1 \leqslant N \leqslant 50$).

В следующих $N-1$ строках содержатся по три целых числа a_i, b_i, l_i — номера городов, соединённые i-й дорогой и длина i-й дороги соответственно ($1 \leqslant l_i \leqslant 100\,000$).

В следующей строке содержится целое число M — количество лисичек ($1 \leqslant M \leqslant N$). В следующей строке содержатся M целых чисел — номера городов, в которых живут лисички. Гарантируется, что все лисы живут в различных городах.

Формат выходных данных

Выведите одно число — максимальное пройденное лисичкой расстояние.

Примеры

fox-meeting.in	fox-meeting.out
2 1 2 5 2 1 2	0
3 1 2 1 2 3 1 2 1 3	1
3 1 2 1 2 3 1 3 1 2 3	0
10 10 5 71846 8 5 10951 3 8 42265 7 10 37832 2 5 29439 6 5 95676 9 6 83661 1 10 28186 4 3 21216 10 1 2 3 4 5 6 7 8 9 10	0

Задача F. Блуждания на зарядке

Имя входного файла:	`dancing-forever.in`
Имя выходного файла:	`dancing-forever.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Как известно, с утра в ЛКШ проходит *танцевальная зарядка*. Сегодня на ней случилось невероятное: на зарядку пришло одинаковое количество мальчиков и девочек.

Известно, что каждый мальчик будет танцевать только с той девочкой, которая ему нравится, но ему могут нравиться сразу несколько девочек. При этом если во время танца он заметит, что какая-то девочка, которая ему нравится, не танцует, то он побежит звать её на танец.

Серёже не нравятся эти блуждания, поэтому он хочет, чтобы после танца ни один маль-

чик не захотел менять своего партнёра. Помогите ему выбрать пары для такого танца. В танце могут участвовать не все мальчики и девочки, но должна участвовать хотя бы одна пара.

Формат входных данных

Единственная строка входного файла состоит из N^2 символов `Y` и `N`.

Если $(i \cdot N + j)$-й символ этой строки — `Y`, то i-му мальчику нравится j-я девочка, и не нравится, если `N`.

Гарантируется, что строка не пустая и что N не превосходит 100.

Формат выходных данных

Если такой танец возможен, выведите в отдельных строчках танцующие пары (первое число — номер мальчика, второе — номер девочки), иначе выведите `-1`.

Примеры

dancing-forever.in	dancing-forever.out
YYNNNYYNNNYYYNNY	4 1 1 2 2 3 3 4
YNNNYYNNYYNNNNYY	1 1 2 2
YNYYNYYNY	2 1 1 3
YYYNNYYYNNNNNYYNNNYYNNNYY	4 4 3 5

Задача G. Сжатие обратной польской нотации

Имя входного файла:	`postfix.in`
Имя выходного файла:	`postfix.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Обратная польская нотация — форма записи математических выражений, в которой операнды расположены перед знаками операций. Она позволяет, не используя скобок, определить порядок операций.

Польская нотация для выражения, состоящего из одной переменной или числа, состоит только из этой переменной или числа. Польская нотация для выражения $A \circ B$ (где A и B — выражения, а $\circ$ — последняя вычисляемая операция выражения) получается конкатенацией польских нотаций для выражений A и B и символа оцерации $\circ$. Например, обратная польская нотация для выражения $((a + f) \times b) \times (c \times d)$ есть $af + b \times cd \times \times$.

Вам дано выражение в обратной польской нотации, состоящее только из переменных и операций. Все операторы бинарные (функции от двух аргументов), ассоциативные и коммутативные. То есть для любых операций $\circ$ и выражений A, B, C верно, что

- $A \circ (B \circ C) = (A \circ B) \circ C$ *(ассоциативность)*.
- $A \circ B = B \circ A$ *(коммутативность)*.

Ваша задача — найти обратную польскую нотацию выражения, которое может быть получено из изначального ассоциативными и коммутативными перестановками, имеющую наименьший *блочный размер*.

Блочным размером строки будем называть количество блоков последовательных равных символов в ней. Например, блочный размер строки «$xx + yy + zz + \times\times$» равен семи.

Формат входных данных

В единственной строке содержится обратная польская нотация выражения. Переменные являются строчными буквами латинского алфавита, возможные операции — «`+`», «`*`», «`#`», «`!`», «`@`», «`$`», «`+`» и «`^`».

Гарантируется, что обратная польская нотация корректная, не пустая, и её длина не превосходит 2500 символов.

Формат выходных данных

Выведите минимальный возможный блочный размер обратной польской нотации.

Примеры

postfix.in	postfix.out
af+b*cd**	7
xy*x*y*x*y*	3
xy@z@ab@c@yc	9
abc++abc++abc++abc*****	13