Задача A. All you need is love

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

Вам дан неориентированный граф без петель и кратных ребер. Найдите в нем максимальное паросочетание.

Формат входного файла

В первой строке даны два числа n и m ($1 \le n \le 17$, $0 \le m \le \frac{n \cdot (n-1)}{2}$).

В следующих m строках даны по два числа a_i и b_i ($1 \le a, b \le n$) — ребро между вершинами a_i и b_i.

Формат выходного файла

В первой строке выведите число k — размер максимального паросочетания.

В следующих k строках выведите ребра, принадлежащие паросочетанию. Используйте формат входных данных.

Пример

match-exp.in	match-exp.out
5 5 1 2 2 3 3 4 4 5 5 1	2 2 1 4 3

Задача B. Be the love you seek

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

Дан двудольный невзвешенный граф. Необходимо найти максимальное паросочетание.

Формат входного файла

В первой строке входного файла три целых числа n, m и k ($1 \le n, m \le 200$, $1 \le k \le n \times m$) — количество чисел в первой и второй долях, а также число ребер соответственно. Далее следуют k строк, в каждой из которых два числа a_i и b_i, что означает ребро между вершиной с номером a_i первой доли и вершиной с номером b_i второй доли. Вершины в обеих долях нумеруются с единицы.

Формат выходного файла

В первой строке выведите число k — размер максимального паросочетания.

В следующих k строках выведите ребра, принадлежащие паросочетанию. Используйте формат входных данных.

Пример

matching.in	matching.out
3 3 5 1 1 1 3 2 1 2 2 3 2	3 2 1 3 2 1 3

Задача C. Ладьи

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

Все клетки квадратной таблицы $n \times n$ пронумерованы в некотором порядке числами от 1 до n^2. Петя делает ходы по следующим правилам. Первым ходом он ставит ладью в любую клетку. Каждым последующим ходом Петя может либо поставить новую ладью на какую-то клетку, либо переставить ладью из клетки с номером a ходом по горизонтали или по вертикали в клетку с номером большим, чем a.

Каждый раз, когда ладья попадает в клетку, эта клетка немедленно закрашивается. Ставить ладью на закрашенную клетку запрещено. Какое наименьшее количество ладей потребуется Пете, чтобы он смог за несколько ходов закрасить все клетки таблицы?

Формат входного файла

На вход подается число n ($1 \le n \le 40$). Далее следуют n строк по n целых чисел, задающие нумерацию клеток таблицы.

Формат выходного файла

Выведите единственное число k — наименьшее количество ладей, которое потребуется Пете.

Примеры

rooks.in	rooks.out
2 3 4 1 2	2

Задача D. Замощение доминошками

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

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 входного файла

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 выходного файла

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 E. Шахматная доска

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

Вася любит играть в необычные шахматы. Его братишка Коля был еще очень маленький. Как-то раз, когда Вася вернулся из школы, он увидел, что его любимую шахматную доску кто-то перекрасил. Вася не сильно разозлился на Колю, потому что очень любил своего брата. Так как у них дома были только черная и белая краски, каждая клетка доски была покрашена в один из этих двух цветов.

Вася решил исправить ошибку брата, он решил покрасить доску так, чтобы она снова стала шахматной. Но Вася почему-то подумал, что хочет красить только диагонали. Причем Вася решил не тратить много времени, поэтому его интересует способ покраски, который содержит наименьшее количество действий. За одно действие Вася может покрасить полностью какую-либо диагональ, в любой из двух цветов: черный или белый. Диагонали бывают двух типов, в зависимости от направления прямой, на которой лежит диагональ. Диагональ, которая лежит на прямой, направленной влево и вниз, является диагональю первого типа, а диагональ, которая лежит на прямой, направленной вправо и вниз, — второго.

Вам предстоит помочь Васе. Задано испорченное Колей шахматное поле. Вам необходимо определить, за какое минимальное количество действий Вася сможет перекрасить доску так, чтобы она стала шахматной.

Формат входного файла

В первой строке входного файла записаны два целых числа: n и m ($1 \le n, m \le 100$) — количество строчек и количество столбцов шахматного поля соответственно.

В следующих n строках записано поле, в каждой строке по m символов. Каждая строка входного файла описывает одну строку шахматного поля. W соответствует белой клетке, B — черной.

Формат выходного файла

В выходной файл нужно вывести число p, количество действий, которое потребуется Васе, чтобы его доска снова стала шахматной. В следующих p строках описаны действия. Каждое действие описано тремя параметрами: тип диагонали, координаты клетки и цвет. Тип диагонали — это число 1 или 2. Координаты клетки — это два целых числа: строка и столбец одной из клеток, которую покрасили этим действием. Цвет — это символ W или B, белый и черный соответственно.

Примеры

chess.in	chess.out
3 3 WBB BWB BBW	1 1 3 1 W
3 3 WBW WWB WWW	1 2 2 1 B

Задача F. Массив и операции

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

Вы выписали на листок массив из n целых положительных чисел $a[1]$, $a[2]$, ..., $a[n]$ и m хороших пар целых чисел (i_1, j_1), (i_2, j_2), ..., (i_m, j_m). Каждая хорошая пара (i_k, j_k) удовлетворяет условиям: $i_k + j_k$ — нечетное число и $1 \le i_k < j_k \le n$.

За одну операцию вы можете выполнить последовательность действий:

- взять одну из хороших пар (i_k, j_k) и некоторое целое число v ($v > 1$), которое делит оба числа $a[i_k]$ и $a[j_k]$
- разделить оба числа на v, т. е. выполнить присваивания: $a[i_k] = \frac{a[i_k]}{v}$ и $a[j_k] = \frac{a[j_k]}{v}$.

Определите, какое максимальное количество операций можно последовательно совершить над данным массивом. Обратите внимание, что одну пару можно использовать несколько раз в описанных операциях.

Формат входного файла

В первой строке записано два целых числа через пробел n, m ($2 \le n \le 100$, $1 \le m \le 100$).

Во второй строке записано n целых чисел пробел $a[1]$, $a[2]$, ..., $a[n]$ ($1 \le a[i] \le 10^9$) — описание массива.

В следующих m строках задано описание хороших пар. В k-й строке содержится два целых числа через пробел i_k, j_k ($1 \le i_k < j_k \le n$, $i_k + j_k$ — нечетное число). Гарантируется, что все хорошие пары различны.

Формат выходного файла

Выведите единственное целое число — ответ на задачу.

Примеры

arrayop.in	arrayop.out
3 2 8 3 8 1 2 2 3	0
3 2 8 12 8 1 2 2 3	2

Задача G. 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 мальчик 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.

Известно, что каждому мальчику нравится хотя бы одна девочка.

Формат входного файла

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 выходного файла

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 H. Heavy Chain Clusterization

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

Группа биологов пытается найти лекарство от вирусного заболевания. Они перепробовали множество антител, которые могут потенциально победить вирусные антигены, и выбрали n антител, которые проявили себя лучшим образом в ходе экспериментов.

Каждое антитело идентефицируется его тяжелой цепью — последовательностью аминокислот.

Множество антител формирует кластер подобия, если выполняется хотя бы одно из следующих требований:

- k-префиксы (первые k аминокислот) всех их тяжелых цепей совпадают
- k-суффиксы (последние k аминокислот) всех их тяжелых цепей совпадают

В целях упрощения дальнейших исследований, биологи хотят сгруппировать антитела в кластера подобия.

Вам нужно разбить выбранные антитела на минимальное количество кластеров подобия.

Формат входного файла

В первой строке даны два числа n и k — количество тяжелых цепей и длина последовательности аминокислот, по которой определяется сходство ($1 \le n \le 5\,000$, $1 \le k \le 550$).

В следующих n строках даны последовательности аминокислот, которые формируют тяжелые цепи антител. Каждая аминокислота описывается заглавной буквой латинского алфавита. Каждая тяжелая цепь содержит не менее k и не более 550 аминокислот.

Формат выходного файла

В первой строке выведите одно целое число c — минимальное количество кластеров подобия. В следующих c строках выведите описание кластеров, по одному на каждой строке.

Каждое описание начинается с m_i — количества антител в кластере. Затем идет m_i чисел — номера этих антител. Антитела нумеруются в порядке их появления во входном файле, начиная с 1.

Каждое антитело должно находиться ровно в одном кластере.

Примеры

heavy.in	heavy.out
4 1 AA AB BB BA	2 2 1 2 2 3 4
3 2 ABA BAB XY	3 1 1 1 2 1 3

Задача I. 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 входного файла

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 выходного файла

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

Задача J. 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 любый 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 входного файла

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 выходного файла

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