Задача A. Сокращение многоугольника

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

Пусть задан некоторый выпуклый n-угольник. *Сокращенным многоугольником* назовем невырожденный выпуклый многоугольник, множество вершин которого является подмножеством множества вершин исходного многоугольника. Напомним, что многоугольник называется *невырожденным*, если его площадь больше нуля.

Ваша задача — найти сокращенный многоугольник наибольшей площади среди тех сокращенных многоугольников, периметр которых не превышает P.

Формат входного файла

Первая строка входного файла содержит количество вершин многоугольника n ($3 \le n \le 20$) и целое число P ($0 \le P \le 10^4$).

Каждая из последующих n строк содержит по два целых числа — координаты соответствующей вершины многоугольника. Вершины заданы в порядке обхода по часовой стрелке, их координаты не превосходят по модулю 100.

Никакие три последовательные вершины многоугольника не лежат на одной прямой.

Формат выходного файла

В выходной файл выведите наибольшую площадь искомого многоугольника. При проверке ответ будет считаться правильным, если он отличается от правильного не более, чем на 10^{-4}.

Если ни одного удовлетворяющего ограничениям сокращенного многоугольника не существует, выведите в выходной файл число `0`.

Примеры

reduce.in	reduce.out
4 0 0 0 0 1 1 1 1 0	0
4 11 0 0 0 3 3 3 3 0	4.5

Задача B. Треугольное путешествие

Имя входного файла:	`triatrip.in`
Имя выходного файла:	`triatrip.out`
Ограничение по времени:	4 секунды
Ограничение по памяти:	64 мегабайта

Туристическое агентство «Четыре русских» предлагает своим клиентам новую услугу. В отличии от обычных туристических агентств, которые обычно продают билеты на самолет либо *в один конец*, либо *туда и обратно*, «Четыре русских» предлагают абсолютно новый вариант: треугольное путешествие.

Треугольное путешествие происходит следующим образом: путешественник начинает свой путь в некотором городе A, перелетает в некоторый город B, затем перелетает в некоторый город C, и наконец возвращается в A.

Теперь руководство компании хочет выяснить, сколько различных треугольных путешествий они могут предложить своим клиентам. Помогите им это выяснить по заданному множеству доступных рейсов.

Формат входного файла

Первая строка входного файла содержит целое число n — количество городов, которые обслуживаются авиакомпаниями, которые продают билеты через агентство ($3 \le n \le 1500$). Следующие n строк содержат по n символов каждая, j-й символ i-й из этих строк равен '`+`', если можно напрямую перелететь из города i в город j, и '`-`', если нельзя. Для удобства i-й символ i-й строки равен '`-`'.

Формат выходного файла

Выведите одно целое число — количество различных треугольных путешествий.

Пример

triatrip.in	triatrip.out

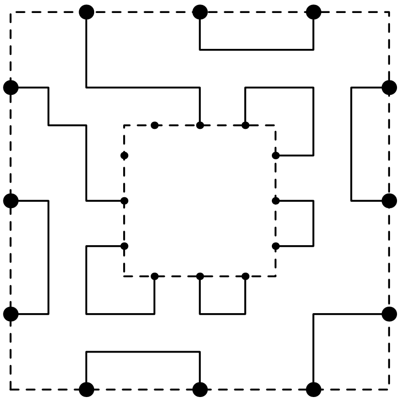
`4`
`--+-`
`+--+`
`-+--`
`--+-`
	`6`

Задача C. Фрактальный Лабиринт

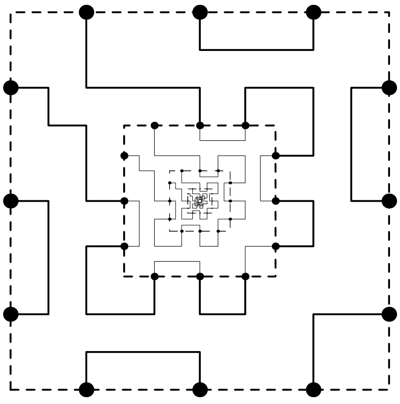
Имя входного файла:	`fractal.in`
Имя выходного файла:	`fractal.out`
Ограничение по времени:	2 seconds
Ограничение по памяти:	64 мегабайта

Роман обожает всевозможные лабиринты. С детства он занимается тем, что ищет из них выход. Вчера он нашёл новый, ещё более увлекательный вид лабиринтов. Он назвал его «фрактальный лабиринт».

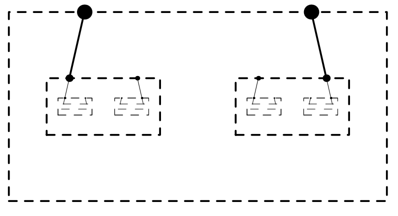
Представьте себе дом с N дверями. Внутри этого дома находятся ещё K домов, каждый из которых представляет собой точную копию "внешнего" дома. Некоторые из этих дверей соединены тропами. Если нарисовать эти тропы, то получится:



Вспоминая, что каждый из «внутренних» домов точная копия «внешнего», то получаем следующая картина:



Одна из дверей «внешнего» дома помечена как вход, а другая — выход. Полагая длину каждой тропы равной 1, требуется найти длину кратчайшего пути от входа до выхода в таком лабиринте.



Формат входного файла

В первой строке входного файла числа N ($2 \le N \le 20$) и K ($0 \le K \le 5$). Следующая строка содержит M, число троп. Следующие M строк содержат описание троп, по одному на строке. Каждая тропа задаётся следующим образом:

`<номер дома>.<номер двери> - <номер дома>.<номер двери>`

Здесь левая и правая части описания обозначают соединяемые тропой двери (дверь описывается своим номером и номером дома, которому она принадлежит). По каждой тропе можно двигаться в обоих направлениях. Номер «внешнего» дома — 0, а «внутренние» нумеруются числами от 1 до K. Двери нумеруются, начиная с 0. Никакие две тропы не соединяют одну и ту же пару дверей. Никакая тропа не соединяет дверь саму с собой. В последней строке содержатся номера входа и выхода D_i и D_o. Эти числа могут совпадать.

Формат выходного файла

Если существует путь от входа к выходу, то выведите длину кратчайшего пути. Иначе, выведите «`no solution`».

Пример

fractal.in	fractal.out
12 1 11 0.0 - 1.1 0.1 - 0.2 1.2 - 1.3 0.3 - 0.4 1.4 - 1.5 0.5 - 0.6 1.6 - 1.7 0.7 - 0.8 1.8 - 1.9 0.9 - 0.10 1.10 - 0.11 0 11	11

Задача D. Транспортная проблема

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

Компания Флатнефть выиграла тендер на строительство системы нефтепроводов между a нефтяными скважинами и b нефтеперерабатывающими заводами Флатландии. В результате было построено множество E, состоящее из m нефтепроводов. Все построенные нефтепроводы однонаправленные, нефть может перемещаться по ним лишь в одном (заранее определенном для каждого нефтепровода) направлении.

Чтобы упростить процесс планирования транспортировки нефти (а также поскольку финансирование в рамках тендера было вполне приличное) все построенные нефтепроводы имеют огромную пропускную способность. Каждый нефтепровод способен в сутки пропускать c баррелей нефти, где c — суммарное количество нефти, которое производится во Флатландии. Система нефтепроводов дополнительно удовлетворяет свойству, что из любого узла сети (скважины или завода) можно доставить нефть до любого другого узла сети.

Однако когда дело дошло до собственно планирования транспортировки нефти, выяснилось, что из-за огромного размера сети спланировать куда и сколько нефти направить достаточно сложно. Компания обратилась к вам за помощью.

Для каждой нефтяной скважины u известно количество добываемой в ней нефти p_u . Для каждого нефтеперерабатывающего завода u известно количество нефти q_u, которое он может перерабатывать в день. Сумма всех p_u равна сумме всех q_u и равна пропускной способности каждого из нефтепроводов c. Любой узел сети (скважина или завод) также может быть транзитным.

План транспортировки нефти сопоставляет каждому нефтепроводу uv количество перекачиваемой по нему в сутки нефти $f(uv)$, при этом должны выполняться следующие условия:

- для каждой трубы uv: $0 \le f(uv) \le c$;
- если u — нефтяная скважина, то, $\sum\limits_{uv \in E} f(uv) - \sum\limits_{vu \in E} f(vu) = p_u$;
- если u — нефтеперерабатывающий завод, то, $\sum\limits_{vu \in E} f(vu) - \sum\limits_{uv \in E} f(uv) = q_u$.

По заданной транспортной сети разработайте план транспортировки нефти.

Формат входного файла

Первая строка входного файла содержит три целых числа: a, b и m ($1 \le a, b \le 50\,000$, $1 \le m \le 100\,000$). Вторая строка содержит a целых чисел: p_i ($1 \le p_i \le 10^4$). Третья строка содержит b целых чисел: q_i ($1 \le q_i \le 10^4$). Сумма всех p_i равна сумме всех q_i.

Следующие m строк описывают нефтепроводы, каждый нефтепровод описывается двумя числами, каждое из которых принимает значение в диапазоне от $-a$ до b, кроме 0 — номер узла, где начинается нефтепровод, и номер узла, где он заканчивается. Нефтедобывающие скважины пронумерованы от -1 до $-a$. Нефтеперерабатывающие заводы пронумерованы от 1 до b.

Формат выходного файла

Если план транспортировки нефти существует, выведите “`YES`” на первой строке выходного файла. Вторая строка в этом случае должна содержать m целых чисел — каждое из этих чисел задает количество нефти, перекачиваемой по соответствующему нефтепроводу.

Если плана не существует, выведите “`NO`”.

Пример

transport.in	transport.out
2 2 6 5 3 4 4 -1 1 -2 1 -2 2 1 2 2 -1 2 -2	YES 5 0 3 1 0 0