Задача A. Юный гитарист

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

Ваня решил заняться музыкой. Но какой музыкальный инструмент выбрать для изучения, он пока еще не решил.

Его друг Женя, будучи немного постарше и поопытнее, уже хорошо освоил гитару. Он хочет продемонстрировать Ване возможности игры на гитаре.

На гитаре шесть струн, каждая из них может быть зажата пальцем на определенном ладу или оставлена открытой (неприжатой ни на каком ладу), от этого изменяется нота, которая звучит, когда струна колеблется.

Чтобы продемонстрировать возможности инструмента, Женя ведет с Ваней такую беседу. Ваня называет некоторый аккорд, а Женя говорит ему, сколькими способами можно этот аккорд взять на гитаре, то есть выбрать на каждой струне лад и зажать его (или оставить ее открытой), чтобы этот аккорд зазвучал. Женя любит громкий звук, поэтому звучать должны все струны.

Нот в музыке, конечно, семь, но помимо них существуют еще полутона. Поэтому для удобства мы будем считать, что нот всего 12. Их обозначения такие: `A`, `Bb`, `B`, `C`, `C#`, `D`, `D#`, `E`, `F`, `F#`, `G`, `G#`.

Кроме этого, `C#` может быть обозначен как `Db`, `Bb` как `A#`, `D#` как `Eb`, `F#` как `Gb`, а `G#` как `Ab`.

Обозначения нот приведены в порядке их следования, то есть за `A` идет `Bb`, за `Bb` идет `B`, …, за `G#` идет снова `A`. Расстояние между любыми двумя соседними нотами равно одному `полутону`.

Струн у гитары шесть, они занумерованы с 1 по 6. Каждая струна издает определенную ноту, когда звучит и не прижата ни на каком ладу. Совокупность нот, соответствующих открытым струнам, называется *строем* гитары.

Если открытую струну, издающую ноту номер i, зажать на j-том ладу (лады нумеруются с единицы), то этой струной будет издаваться нота $i + j$, то есть, от ноты i надо сместиться на j нот по циклу. Например, если открытая струна издает ноту `D`, то зажав ее на третьем ладу, мы получим ноту `F`, а на восьмом — ноту `Bb`.

На гитаре Вани N ладов, поэтому $1 \le j \le N$.

Аккордов существует большое множество, но строятся они по общей схеме. Вначале выбирается так называемая *тоника* аккорда — нота, от которой он будет строиться. Затем остальные ноты аккорда выписываются уже относительно нее.

Запись аккорда состоит из двух частей.

Первая часть — это всегда обозначение тоники аккорда.

Вторая часть описывает само звучание аккорда. Мы ограничимся несколькими частными случаями.

- *Мажорный аккорд.*

  К тонике добавляются ноты, отстоящие на 4 и 7 полутонов.

  Вторая часть записи этого аккорда пуста.

  Пример: `Eb`, состоит из нот `Eb`, `G`, `Bb`.

- *Минорный аккорд.*

  К тонике добавляются ноты, отстоящие на 3 и 7 полутонов.

  Вторая часть записи этого аккорда состоит из маленькой латинской буквы m.

  Пример: `Am`, состоит из нот `A`, `C`, `E`.

- *Мажорный септаккорд.*

  Образуется из мажорного аккорда путем добавления ноты, отстоящей от тоники на 10 полутонов.

Вторая часть состоит из цифры 7.

Пример: `E7`, состоит из нот `E`, `B`, `G#`, `D`.

- *Минорный септаккорд.*

  Образуется из минорного аккорда путем добавления ноты, отстоящей от тоники на 10 полутонов.

  Вторая часть состоит из буквы m и цифры 7.

  Пример: `Gm7`, состоит из нот `G`, `D`, `Bb`, `F`.

Напишите программу, помогающую Жене отвечать на вопросы Вани.

Учтите, что аккорд считается взятым на гитаре, если ни одна нота аккорда не пропущена и не взята ни одна нота, не принадлежащая аккорду.

В отличие от реальной игры на гитаре, расстояние между ладами, на которых зажата хотя бы одна струна, может быть любым.

Формат входного файла

В первой строке файла содержится N — число ладов гитары ($0 \leq N \leq 9$).

Во второй строке записан строй гитары — шесть нот, соответствующие шести струнам гитары. Ноты обозначены, как в тексте задачи, и отделены друг от друга пробелами. Регистр букв важен.

В третьей строке записан аккорд, который надо взять на гитаре.

Формат выходного файла

Выведите количество способов взять данный аккорд на данной гитаре.

Примеры

<table>
<tr><th>guitar.in</th><th>guitar.out</th></tr>
<tr><td><code>3
E B G D A E
C</code></td><td><code>4</code></td></tr>
<tr><td><code>0
A C E A C E
Am</code></td><td><code>1</code></td></tr>
</table>

Если обозначить вариант взятия аккорда шестью цифрами — номерами ладов, на которых зажаты струны с первой по шестую, считая открытую струну зажатой на нулевом ладу — то в первом примере четыре способа таковы:

$$(0;1;0;2;3;0)$$
$$(3;1;0;2;3;0)$$
$$(0;1;0;2;3;3)$$
$$(3;1;0;2;3;3)$$

Во втором примере можно играть только на открытых струнах, зато гитара настроена ровно на требуемый аккорд.

Задача B. Доказательство в HOL

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

Студент Дима учится на втором курсе университета. На втором курсе в его университете читается курс математической логики. В этом курсе особое внимание акцентируется на *автоматических доказателях* — программах, позволяющих с их помощью доказывать различные сложные теоремы.

Курсовая работа по математической логике у Димы такая: необходимо доказать с помощью автоматического доказателя *HOL*, что шахматную доску размером $2^N \times 2^N$, из которой вырезана одна клетка, можно покрыть в один слой уголками из трех клеток.

Дима не верит в то, что это правда, и пытается составить контрпример.

Ваша задача — доказать Диме, что он неправ, и решить задачу для Диминых входных данных.

Формат входного файла

В первой и единственной строке находятся три числа N, X, Y, $1 \leq N \leq 6$, $1 \leq X \leq 2^N$, $1 \leq Y \leq 2^N$.

Этими числами задана доска $2^N \times 2^N$, из которой вырезана клетка с координатами (X, Y). X — координата по горизонтали, Y — по вертикали, $(1, 1)$ — верхний левый угол доски.

Формат выходного файла

Каждый уголок характеризуется своим уникальным номером. Уголки пронумерованы начиная с единицы, без пропусков.

В выходной файл выведите 2^N строк по 2^N чисел — номера уголков, покрывающих соответствующие клетки.

Вырезанную клетку обозначьте нулем.

Примеры

hol.in	hol.out
1 1 1	0 1 1 1
2 2 2	2 2 3 3 2 0 1 3 4 1 1 5 4 4 5 5

Задача C. Осмотр королевства

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

Давным давно в одном королевстве правил мудрый король. В том королевстве было n городов, любые два из которых были соединены дорогой. С целью безопасности дорожного движения, по каждой дороге разрешалось перемещаться только в одну сторону.

Король любил свое королевство. Каждый год он k раз осуществлял осмотр королевства. Каждый осмотр начинался в столице, затем король, перемещаясь по дорогам, посещал некоторые города, и, наконец, прибывал в курортный город на берегу моря, где он отдыхал после нелегкой работы. Таким образом король посещал каждый город в королевстве ровно один раз в течении года (за исключением столицы, где каждый осмотр начинался, и морского курорта, где каждый осмотр заканчивался).

Но годы уходили и король становится стар. И ему было все сложнее делать эти k осмотров. Так что однажды он позвал своего министра транспорта и приказал ему сделать новую программу для осмотра королевства. Осмотр должен начинаться в столице и заканчиваться на морском курорте. Однако теперь весь этот путь должен посещать все города, которые ранее посещались в процессе k осмотров. И более того, если ранее некоторые города A и B посещались в процессе одного осмотра и причем A посещался до B, то новый осмотр также должен посещать A до B.

Помогите королю сделать осмотр королевства менее утомительным, разработайте новый маршрут для осмотра.

Формат входного файла

Первая строка входного файла содержит два целых числа: n и k ($3 \le n \le 400$, $1 \le k \le 100$). Следующие n строк описывают дороги. Каждая строка содержит n символов, i-й символ i-й из этих строк равен '`.`'. Для всех $j \ne i$ j-й символ i-й строки равен '`+`', если дорога идет из j-го города в i-й, либо '`-`', если дорога идет из i-го города в j-й. Столица имеет номер 1, морской курорт имеет номер n.

Следующие k строк содержат описание маршрутов, по которым король традиционно осуществлял осмотр. Каждый маршрут описывается на одной строке. Описание содержит номера городов в том порядке, в котором король посещал города в процессе соответствующего осмотра. Гарантируется, что каждый осмотр начинается в столице, заканчивается в морском курорте, и следует по всем дорогам в корректном направлении. Каждый город, за исключением столицы и морского курорта, посещается ровно в одном маршруте. Ни один маршрут не идет непосредственно из столицы в морской курорт. Ни один маршрут не посещает в качестве промежуточного города столицу или морской курорт.

Формат выходного файла

Выведите n чисел — порядок, в котором следует посещать города. Если выполнить требования короля невозможно, выведите -1 в выходной файл.

Пример

inspection.in	inspection.out
7 2 .------ +.+---- +-.-+++ +++.--- ++-+.-- ++-++.- ++-+++. 1 3 4 7 1 2 5 6 7	1 3 2 4 5 6 7

Задача D. Защита деревьев

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

Андрюша получил на день рождения подарок от родителей — маленького щенка. Он назвал его Весельчак. Весельчак еще очень маленький, но у него уже есть зубы. Они очень острые, и должно быть еще растут, потому что Весельчак очень любит погрызть все, что попадает в его поле зрения.

Летом Андрюша с родителями приехал на дачу и Весельчака поселили во дворе. Папа у Андрюши — садовод любитель, и он выращивает во дворе n плодово-ягодных деревьев. Теперь он опасается, что Весельчак может погрызть его любимые деревья, поэтому он решил их защитить.

Он нашел в сарае три длинные доски и теперь собирается соорудить из них забор. Забор будет треугольным, причем никакая часть доски не должна выступать за границу забора, поскольку в этом случае кто-нибудь может запнуться об нее и упасть. Распиливать доски нельзя — они могут в дальнейшем пригодиться в хозяйстве.

Однако может получиться так, что нельзя огородить треугольным забором все деревья. В этом случае папа Андрюши хочет огородить как можно больше деревьев. Помогите ему определить, сколько деревьев он может защитить.

Формат входного файла

Первая строка входного файла содержит n — количество деревьев во дворе ($1 \le n \le 70$). Вторая строка содержит три целых числа a, b и c — длины досок, которые будут использованы для забора ($1 \le a, b, c \le 10^4$, существует невырожденный треугольник со сторонами a, b и c). Следующие n строк содержат по два целых числа — координаты деревьев. Координаты не превышают 10^4 по абсолютной величине. Вы можете пренебречь толщиной деревьев. Никакие два дерева не совпадают.

Формат выходного файла

Выведите одно целое число — количество деревьев, которое можно защитить.

Пример

protect.in	protect.out
5 3 4 6 6 0 0 0 1 2 2 3 6 1	3