PROBLEM: Regular Expressions is one of the ACSL categories that is usually only tested at the ACSL Invitational All-Star Contest. However, it is being used here as a programming problem to show how a long string or multiple strings can be written in condensed format. The concept of regular expressions was first formalized by Stephen Kleene in the 1950s. The following operations are used:

.	Matches any single character. Ex: a.c matches aac, abc, acc, adc, a#c, a4c, …etc.
[ ]	Matches a single character contained within the brackets. Ex: [abc] matches a, b, or c.
[^]	Matches any single character not contained within the brackets. [^abc] matches any character other than a, b, or c.
*	Matches the preceding character zero or more times. Ex; a*b matches b, ab, aab, aaab, …etc.
{m,n}	Matches the preceding character at least m but not more than n times. Ex: a{3,5} matches aaa, aaaa, or aaaaa.

INPUT: There will be 6 lines of input. The first line will contain 10 character strings. The last 5 lines will contain a valid regular expression string. Each regular expression will have at most two (2) operators.

OUPUT: For each regular expression print all the character strings that are matches to the strings on Line #1. If none match, then print NONE. # is used here to represent the empty string.

SAMPLE INPUT
1. #, aac, acc, abc, ac, abbc, abbbc, abbbbc, aabc, aabbc
2. a.c
3. a[ab]c
4. a[^ab]c
5. ab*c
6. a.b{2,4}c

SAMPLE OUTPUT

1. aac, acc, abc
2. aac, abc
3. acc
4. ac, abc, abbc, abbbc, abbbbc
5. aabbc, abbbc, abbbbc