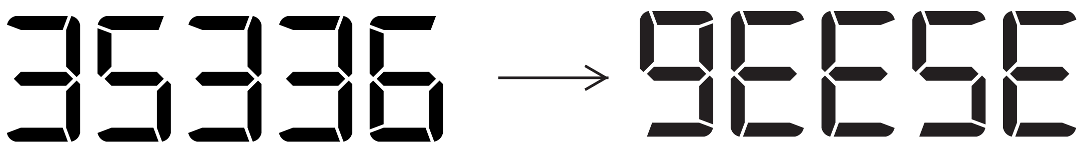
CCC 2012 Stage 2
Day 1, Problem 1: Choose Your Own Arithmetic

Problem Description
In Waterloo, you probably have seen some geese. How can you see geese with your calculator? Start with 6, add 7, multiply by 6, multiply by 8, add 7, multiply by 8, and multiply by 7, giving 35336. Then if you flip your calculator upside down, it says gEESE:



You want to write a program to help automatically build tricks of this type. However, your calculator has a lot of broken buttons: the only mathematical operators that work are $+$ and $\times$, and only a few of the digits work. Your goal is to figure out whether your half-broken calculator can achieve a given target value, using single-digit inputs and a fixed number of operations.

Note: the calculator performs the operations as soon as they are entered, rather than following any rules for order of operations (see Sample Input 2).

Input Specification
The first line of input is W, the exact number of operations you must use. W will be an integer between 0 and 6. The second line of input is $1 \leq D \leq 10$, the number of working digit keys. On each of the D following lines, a working digit is given; these values are distinct integers from 0 to 9. Finally, an integer $1 \leq V \leq 5$ is given, the number of target values; on each of the following V lines there is an integer between 0 and 5000000 (inclusive) giving a target value which you'd like to achieve on your calculator.

Output Specification
The output consists of V lines corresponding to the target values; each line contains "Y" if that target value can be achieved, and "N" if it cannot be achieved, using exactly W operations with the D given digits.

Precisely, a target value T can be achieved if, starting with one of the D digits, and then by adding or multiplying exactly W times by one of the digits, you end up with T. Digits can be re-used, and you do not need to use all of the digits. You cannot enter multi-digit numbers.

Sample Input 1
```
6
3
```

```
6
7
8
1
35336
```

Output for Sample Input 1

```
Y
```

Sample Input 2

```
3
2
4
9
2
97
88
```

Output for Sample Input 2

```
N
Y
```

Explanation

First line: we cannot achieve 97 using the rules of this calculator, so the output is `N` (even despite that $4 \times 4 + 9 \times 9 = 97$, when the typical order of operations rules are taken into account). Second line: start with 9, add 9, add 4, and multiply by 4; this gives 88.

CCC 2012 Stage 2
Day 1, Problem 2: The Hungary Games

Problem Description
Welcome to the Hungary Games! The streets of Budapest form a twisted network of one-way streets. You have been forced to join a race as part of a "Reality TV" show where you race through these streets, starting at the Széchenyi thermal bath (s for short) and ending at the Tomb of Gül Baba (t for short).

Naturally, you want to complete the race as quickly as possible, because you will get more promotional contracts the better you perform. However, there is a catch: any person who is smart enough to take a shortest s-t route will be thrown into the Pálvölgyi cave system and kept as a national treasure. You would like to avoid this fate, but still be as fast as possible. Write a program that computes a strictly-second-shortest s-t route.

Sometimes the strictly-second-shortest route visits some nodes more than once; see Sample Input 2 for an example.

Input Specification
The first line will have the format N M, where N is the number of nodes in Budapest and M is the number of edges. The nodes are $1, 2, \ldots, N$; node 1 represents s; node N represents t. Then there are M lines of the form A B L, indicating a one-way street from A to B of length L. You can assume that $A \neq B$ on these lines, and that the ordered pairs (A, B) are distinct.

Output Specification
Output the length of a strictly-second-shortest route from s to t. If there are less than two possible lengths for routes from s to t, output -1.

Limits
Every length L will be a positive integer between 1 and 10000. For 50% of the test cases, we will have $2 \leq N \leq 40$ and $0 \leq M \leq 1000$. All test cases will have $2 \leq N \leq 20000$ and $0 \leq M \leq 100000$.

Sample Input 1
```
4 6
1 2 5
1 3 5
2 3 1
2 4 5
3 4 5
1 4 13
```

Output for Sample Input 1

```
11
```

Explanation

There are two shortest routes of length 10 ($1 \rightarrow 2 \rightarrow 4, 1 \rightarrow 3 \rightarrow 4$) and the strictly-second-shortest route is $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$ with length 11.

Sample Input 2

```
2 2
1 2 1
2 1 1
```

Output for Sample Input 2

```
3
```

Explanation

The shortest route is $1 \rightarrow 2$ of length 1, and the strictly-second route is $1 \rightarrow 2 \rightarrow 1 \rightarrow 2$ of length 3.

CCC 2012 Stage 2
Day 1, Question 3: Mhocskian Languages

Problem Description

Linguists are currently studying Mhocskian, the language of the native inhabitants of Mhocsky island. The linguists have found a description of how the natives construct words in Mhocskian, and a list of words. The linguists would now like to know which of the words in the list are **valid** Mhocskian words.

Rules

Words in Mhocskian are constructed according to a set of **rules**. These rules involve two types of components: **variables** and **terminals**. A variable is an uppercase letter used in the description of the rules. A terminal is a lowercase letter that is part of a Mhocskian word.

There are two types of rules. The first type of rule allows you to replace a variable V by two variables V_1V_2 in that order, and we write $V \to V_1V_2$ as a short form for this type of rule. The second type of rule allows you to replace a variable V by a terminal t, and we write $V \to t$ as a short form for this type of rule.

One of the variables is the start variable. A word w is composed of lowercase letters from the English alphabet. It is a **valid** Mhocskian word if, starting from the start variable, it is possible to follow a sequence of rules to obtain w.

Example

Suppose we have variables $\{S, A, B\}$, terminals $\{a, b\}$, and rules

$$\{S \to AB, S \to a, A \to a, B \to b\}.$$

The word "ab" is a valid Mhocskian word because it can be constructed in the following way: $S \to AB \to aB \to ab$. The word "a" can be constructed simply by $S \to a$. The word "b" cannot be constructed.

Input

On the first line, two integers V and T in that order.

On the second line, V space separated uppercase letters, the variables. The **first** variable on the line is always the start variable.

On the third line, T space separated lowercase letters, the terminals.

On the fourth line, there is an integer R_1. R_1 lines follow, each of which is of the form V t, representing a rule $V \to t$.

On the next line, there is an integer R_2. R_2 lines follow, each of the form V V_1 V_2, representing the rule $V \to V_1V_2$.

On the next line, there is an integer W. W lines follow, each contains a single word made entirely of lowercase letters.

Output
The output must contain W lines. On line i, output a 1 if the i^{th} word is a valid Mhocskian word, and 0 otherwise.

Constraints
$1 \leq V, T \leq 26$
$1 \leq R_1 + R_2 \leq 30$
$1 \leq W \leq 20$
Each of the words in the linguists' list will have length between 1 and 30.

Sample Input
```
5 2
I S A B C
a b
2
A a
B b
7
I A B
I A C
C S B
S A B
S A C
I S S
S S S
4
abababaaabbbaabbaabb
abab
bbaa
aaabababbaaabbbb
```

Sample Output
```
1
1
0
1
```