# Tokenization

Time Limit: 1 Second
Memory Limit: 256 MB

In natural language processing (NLP) tasks like question answering and summarization, the input to the language model is usually a sequence of words consisting of a context and a query. Since most modern deep learning models take numbers as input, a preprocessing step is required to map the sequence of words to a sequence of numbers.

Tokenization is a classic technique for addressing this issue. Instead of mapping each word in the sequence to a unique integer identifier, it first decompose the word into prefix, root, and suffix (known as tokens), encoding each separately. For instance, the word "department" will be decomposed into "de", "part", and "ment". This encoding method will significantly decrease the size of the vocabulary needed to cover the majority of English words.

The remaining challenge is to convert the outputting tokens back to human readable language, known as decoding. Since tokenization ignores spaces between words, it is difficult to group tokens into words after the language model has generated the output. As an NLP researcher, you are working on a project that involves implementing the decoding step of tokenization. You are provided with a dictionary containing all valid words, and the input to your program will be a sequence of string tokens.

## Input

The first line of input contains two integers $n$ and $m$ ($1 \leq n, m \leq 100$) - the number of tokens in the input and the number of words in the dictionary.

The following $m$ lines describe the dictionary. Each line contains a single string $s$ ($1 \leq |s| \leq 100$), denoting a word in the dictionary. It is guaranteed that the string only contain letters.

The last line contains $n$ strings $a_1, \ldots, a_n$ ($1 \leq |a_i| \leq 100$) separated by space, denoting the input to the program. It is guaranteed that the strings only contain letters.

## Output

Output a sequence of strings separated by space, such that the sequence is obtained by deleting any (possibly 0) spaces from the input and each string is a word in the provided dictionary. If there are more than one solutions, you can output any of them. Output $-1$ if it is impossible to obtain such a sequence.

## Sample Inputs

```
13 9
A
brown
dog
fox
jumps
lazy
over
quick
the
A qu ick br own fox jump s over the la zy dog
```

## Sample Outputs

```
A quick brown fox jumps over the lazy dog
```