☆ Simple Text Queries

In this challenge, you will be given an array of *sentences* and and array of phrases. You must determine which sentences contain all of the words of a phrase.

For example, given the following *sentences*:

1. bob and alice like to text each other
2. bob does not like to ski
3. alice likes to ski

And the query phrases:

1. bob alice
2. alice
3. like

The results of the *queries* are:

1. *sentences[0]*
2. *sentences[0], sentences[2]*
3. *sentences[0], sentences[1]*

Note: The word *like* in *queries[2]* does not match *likes* in *sentences[2]*. Matches must be exact.

After each query has been processed, print out the indexes of the matching *sentences*. If all of the words of a phrase occur multiple times in a sentence, the index of that *sentences[i]* should occur that number of times.

Function Description

Complete the function *textQueries* in the editor below. The function must print the space-separated indexes of the matching *sentences* for each query on a separate line. A matching *sentences* index should occur once for each occurrence of the entire query in the sentence. If there are no matching *sentences*, print *-1*.

textQueries has the following parameter(s):

sentences[sentences[0],...sentences[n-1]]: an array of sentence strings consisting of space-separated words

Constraints

- $1 \le n \le 10^4$
- $1 \le q \le 10^4$
- The number of words in any sentence or query phrase is in the range *[1-10]*.
- Each word has at most *11* characters.
- No word appears in more than *10* sentences.
- Each word consists of uppercase and lowercase English alphabetic letters only (i.e., the character class *[a-zA-Z]*).

Input Format for Custom Testing

Sample Case 0

Sample Input 0

```
3
jim likes mary
kate likes tom
tom does not like jim
2
jim tom
likes
```

Sample Output 0

```
2
0 1
```

Explanation 0

We perform the following *q = 2* queries on *sentences = ["jim likes mary", "kate likes tom", "tom does not like jim"]*:

0. Find the indexes of sentences containing both the words "*jim*" and "*tom*". The only sentence containing both words is located at index *2*, so the array *[2]* is stored in index *0* of the results array.
1. Find the indexes of sentences containing the word "*likes*". This word appears in *sentences[0]* and *sentences[1]*, so the array *[0, 1]* is stored in index *1* of the results array.

We then print the array *[[2], [0, 1]]* as our answer.

Sample Case 1

```
are you how
it goes to
goes done are it
2
done it
it
```

Sample Output 1

```
0 3
0 2 3
```

Explanation 1

We perform the following *q = 2* queries on *sentences = ["how it was done", "are you how", "it goes to", "goes done are it"]*:

0. Find the indexes of sentences containing both the words "*done*" and "*it*". These words appear in *sentences[0]* and *sentences[3]*, so the array *[0, 3]* is stored in index *0* of the results array.
1. Find the indexes of sentences containing the word "*it*". This word appears in *sentences[0]*, *sentences[2]* and *sentences[3]*. The array *[0, 2, 3]* is stored in index *1* of the results array.

We then print the results array *[[0, 3], [0, 2, 3]]* as our answer.

Sample Case 2

Sample Input 2

```
3
it go will away
go do art
what to will east
3
it will
go east will
will
```

Sample Output 2

```
0
-1
0 2
```

Explanation 2

appear in *sentences[0]*, so the array *[0]* is stored in index *0* of the results array.

1. Find the indexes of sentences containing the words "*go*", "*east*", and "*will*". These words do not appear in any sentence, so the array *[-1]* is stored in index *1* of the results array.
2. Find the indexes of sentences containing the word "*will*". This word appears in *sentences[0]* and *sentences[2]*, so the array *[0, 2]* is stored in index *2* of the results array.

We then print *[[0], [-1], [0, 2]]* as our answer.

YOUR ANSWER

Draft saved 12:32 am Original code Python 3

```
#!/bin/python3

import math
import os
import random
import re
import sys


```

Twitter 2018-2019 University Recruiting Coding Ch...

6d 13h
to test end

Line: 29 Col: 1

Test against custom input

Run Code | Submit code & Continue

(You can submit any number of times)

- Section 1 -

1

2

- Section 2 -

3

Download sample test cases

The input/output files have Unix line endings. Do not use Notepad to edit them on windows.