TWO SIGMA

Two Sigma Campus Coding Cha

01h : 29m
to test end

1

2

☆ Missing Words

Given two sentences, *s* and *t*, *t* is a subsequence of *s* if all of the words in *t* occur in the same order within *s*. Words do not have to appear contiguously in *s*, but order must be maintained. For example, given the sentence “I like cheese”, one example of a subsequence would be “I cheese”.

In this challenge, you will be given two sentences, *s* and *t*. It is guaranteed that string *t* is a subsequence of string *s*. When reading string *s* from left to right, locate the first occurrence of subsequence *t*. Remove this subsequence and return the remaining elements of string *s* in order.

Example:

s = I like eating cheese do you like cheese
t = like cheese
Return: I eating do you like cheese

Helpful Note:
Be sure to always take the first occurrence of an element if it appears multiple times in a subsequence. For example:

s = I like soft cheese and hard cheese yum
t = like cheese yum

Notice that there are two subsequences in string *s* containing *t*

I like soft cheese and hard cheese yum
I like soft cheese and hard cheese yum

The correct return will remove the the 4th word, not the 7th word, returning “I soft and hard cheese” rather than “I soft cheese and hard”.

Function Description

Complete the function missingWords in the editor below. It must return the list of words in *s* that remain after removing the first occurrence of subsequence *t* with each returned word on its own line.

missingWords has the following parameter(s):

s: a sentence of space-separated words

t: a sentence of space-separated words

Constraints

- Strings *s* and *t* consist of English alphabetic letters (i.e., *a−z* and *A−Z*) and spaces only.
- $1 \le |t| \le |s| \le 10^6$
- *1* ≤ length of any word in *s* or *t* ≤ *15*
- It is guaranteed that string *t* is a subsequence of string *s*.

Input Format for Custom Testing

Sample Case 0

Sample Input 0

```
I use HackerRank to be a better programmer and to practice
HackerRank to practice
```

Sample Output 0

```
I
use
be
a
better
programmer
and
to
```

Explanation 0

We find the first subsequence of “HackerRank to practice” in string s, which occurs at words 3,4 and 11. Removing these three words, we return the remaining elements of string s as separate words.

YOUR ANSWER

We recommend you take a quick tour of our editor before you proceed. The timer will pause up to 90 seconds for the tour. Start tour ✖

Original code Java 8 ⌄

```
import ↔;

public class Solution {



    // Complete the missingWords function below.
    static List<String> missingWords(String s, String t) {


    }




    public static void main(String[] args) throws IOException
↔}
}

```

Line: 9 Col: 1

TWO SIGMA

Two Sigma Campus Coding Cha

08s
to test end

1

2

☆ Substrings

Consider a string, s = "*abc*". An alphabetically-ordered sequence of substrings of s would be *{"a", "ab", "abc", "b", "bc", "c"}*. If we reduce this sequence to only those substrings that *start with a vowel and end with a consonant*, we're left with *{"ab", "abc"}*. The alphabetically *first* element in this reduced list is "*ab*", and the alphabetically *last* element is "*abc*". As a reminder:

- *Vowels:* *a*, *e*, *i*, *o*, and *u*.
- *Consonants:* *b*, *c*, *d*, *f*, *g*, *h*, *j*, *k*, *l*, *m*, *n*, *p*, *q*, *r*, *s*, *t*, *v*, *w*, *x*, *y*, and *z*.

Complete the *findSubstrings* function in your editor. It has *1* parameter: a string, s, consisting of lowercase English letters ($a - z$). The function must find the substrings of s that *start with a vowel* and *end with a consonant*, then print the alphabetically *first* and alphabetically *last* of these substrings.

Input Format

The locked stub code in your editor reads a single string, s, from stdin and passes it to your function.

Constraints

- $3 \leq \text{length of } s \leq 5 \times 10^5$

Output Format

Your function must print two lines of output denoting the alphabetically first and last substrings of s that start with a vowel and end with a consonant. Print the alphabetically *first* qualifying substring on the first line, and the alphabetically *last* qualifying substring on the second line.

Sample Input 1

```
aba
```

Sample Output 1

```
ab
ab
```

Explanation 1

"ab" is the only possible substring which starts with a vowel (*a*) and ends with a consonant (*b*). Because we only have *1* qualifying substring, *"ab"* is both the alphabetically first and last qualifying substring and we print it as our first and second lines of output.

Sample Input 2

```
aab
```

Sample Output 2

```
aab
ab
```

Explanation 2

There are *2* possible substrings which start with a vowel and end with a consonant: *"aab"* and *"ab"*. When ordered alphabetically, *"aab"* comes before *"ab"*. This means that we print *"aab"* (the alphabetically first qualifying substring) as our first line of output, and we print *"ab"* (the alphabetically last qualifying substring) as our second line of output.

YOUR ANSWER

Draft saved 12:16 pm Original code Java 8