# Food Finder

You will be given **two sequences of characters, representing vowels and consonants**. Your task is to start checking if the following words could be created:

* **"pear"**
* **"flour"**
* **"pork"**
* **"olive"**

Start by taking the **first character** of the **vowels collection** and the **last character** from the **consonants collection.** Then check if these letters are present in one or more of the given words. If these letters are present, you should store the information. Then process to the next couple of letters until there are no more **consonant** letters left.

A **letter (vowels or consonants) could participate in more than one word, for example:**

The letter **'**o**'** is present in **"flour", "pork",** and **"olive".**

The letter **'**l**'** is present in **"flour",** and **"olive".**

**Keep in mind that:**

* A **vowel** letter is always returned to the collection, whether used or not.
* A **consonant** letter is always removed from the collection, whether used or not.

As a result, you should **check how many of the given words were** found and print:

**"Words found: {numberOfWordsFound}**

**{wordOne}**

**{wordTwo}**

**…**

**"**

**Look at the provided examples for a better understanding of the problem.**

## Input

* On the **first line**, you will receive characters representing the **vowels**, **separated** by a single space (**" "**).
* On the **second line**, you will receive characters representing the **consonants**, **separated** by a single space (**" "**)**.**

## Output

* As a result, print on the first line how many words have been found and on the next N lines, every word:

**"Words found: {numberOfWordsFound}**

**{wordOne}**

**{wordTwo}**

**…**

**"**

**Print words in the same order as in the problem's description.**

## Constraints

* All letters will be lowercase.
* All letters in the given words are unique.
* There will be no case where no word is matched.
* The letter **'**y**' will be always a vowel.**

## Examples

|  |  |
| --- | --- |
| ****Input**** | ****Output**** |
| **e a u o**  **p r l x f** | **Words found: 2**  **pear**  **flour** |
| ****Comment**** | |
| We start by taking the first two letters **'e' and 'f'. We have concurrence, 'e' is found into "olive" and "pear", 'f' is found into "flour".**  **We add 'e' to the end of the vowel collection (a u o e) and remove 'f' out of the consonant collection (p r l x).**  Onto the next iteration we continue **'a' and 'x', where 'a' is found in "pear", and 'x' is not located into any word, 'a' is added at the end of the vowels collection (u o e a) and remove 'x' (p r l).**  **Next, we have 'u' and 'l', where both letters are found in "flour" and 'l' is found in "**olive**", we add 'u' back in the collection (o e a u) and remove 'l' (p r).**  **Next, we have 'o' and 'r', 'o' is found in "**flour**"**, **"**pork**"**, and **"**olive**"**, and **'r' is found in "**pear**"**, **"**flour**"**, **"**pork**"**, we add **'o' back in the collection (e a u o) and remove 'r' (p).**  **In this iteration, one word was found: "**flour**"**.  In the last iteration, we take **'e' and 'p'. We have already found all 'e' letters so far, so we only search for 'p', which is found in "**pear**"** and **"**pork**"**.  As a result, we found two words **"**pear**"** and **"**flour**",** so we print the corresponding output.  Final result: **"**pear**"**, **"**flour**"**, **"**pork, **"**olive**"**. | |

|  |  |
| --- | --- |
| ****Input**** | ****Output**** |
| **a o y**  **b h p j r n k** | **Words found: 1**  **pork** |