Consider a collection of words. Sometimes, it is possible to identify each word uniquely from its prefix. For instance, if the collection is \{love, hero, here, half, yet\}, a word that starts with 1 must be love. In the same way, if we know that a word starts with y, it must be yet. However, to identify the word half we need the prefix ha, to identify the word hero as well as here we need to write the whole words.

Your program must print the words in increasing order according to the length of the shortest prefix that identifies them. In a event of a tie, it must print the words in lexicographical order.

**Input**

The input consists of various cases. Each case starts with its number of words \(n \geq 2\). \(n\) words follow, each one contains between 1 and 20 lowercase letters. All the words of a collection will be identifiable with some of its prefixes (eventually, the whole word). That is, no given word will not be prefix of other word of the same case.

**Output**

For each case, your program must print the prefixes (with the words that they indentify to) in increasing order by length and, in a event of a tie, in lexicographical order. it must separate the output of different cases with a line with 10 dashes.

**Scoring**

- **Test1**: Tests with \(n \leq 10\).  
  45 Points

- **Test2**: Tests with \(n \leq 10000\).  
  55 Points

**Sample input**

```
5
hero
here
half
love
yet
```

**Sample output**

```
1 love
y yet
h ha half
here here
hero hero
----------
b bye
h hello
```

**Problem information**

Author: Salvador Roura  
Translator: Carlos Molina  
Generation: 2014-01-29 13:01:56