As Easy As C-A-B

We all know how to alphabetize a list when you know the order of the alphabet. But can you find the order of the alphabet from an ordered list of words?

Consider the ordered list [**cab**, **cda**, **ccc**, **badca**]. It is clear that '**c**' comes before '**b**' in the underlying alphabet because '**cab**' comes before '**badca**'. Similarly, we know '**a**' comes before '**d**', because '**cab**' < '**cda**', '**a**' comes before '**c**' because '**cab**' < '**ccc**', and '**d**' comes before '**c**' because '**cda**' < '**ccc**'. The only ordering of these four letters that is possible is **adcb**.

Of course, it may not work out so well. If the word list is [**abc**, **bca**, **cab**, **abc**] there is no alphabet that works. The list is *inconsistent*. If the word list is [**dea**, **cfb**] we don't know about the relative positions of any of the letters other than '**c**' and '**d**'. The list is *incomplete*. Every list will fall into exactly one of the following three categories:

1. The list is *correct* if a single alphabet will yield the ordering
2. The list is *incomplete* if more than one alphabet will yield the ordering
3. The list is *inconsistent* if no alphabet will yield the ordering

Given a list of words, determine if the list is *correct*, *incomplete* or *inconsistent*, and if it is *correct*, give the single underlying ordered alphabet.

Input

Each input will consist of a single test case. Note that your program may be run multiple times on different inputs. The first line of input contains a lowercase letter ***last***, and an integer ***n*** (**1**≤***n***≤**100**). Each of the following ***n*** lines will have a string ***s*** (**1**≤|***s***|≤**50**) consisting only of lowercase letters '**a**'-***last***.

Output

If the list is *correct*, and it is possible to uniquely determine the ordering of the letters '**a**'-***last***, output that ordering as a single string. If the list is *incomplete*, and there's not enough information to determine the positions of all the letters, output **0** (zero). If the list is *inconsistent* in any way then output **1**.

Sample Input	Sample Output

`d 4`
`cab`
`cda`
`ccc`
`badca`
	`adcb`

`c 4`
`abc`
`bca`
`cab`
`abc`
	`1`

`f 2`
`dea`
`cfb`
	`0`

`b 3`
`a`
`bb`
`b`
	`1`

Note: the last case is inconsistent because there is no alphabet for which **`bb`** comes before **`b`**.