

*Lexicographical order* is often known as alphabetical order when dealing with strings. A string is *greater* than another string if it comes later in a lexicographically sorted list.

Given a word, create a new word by swapping some or all of its characters. This new word must meet two criteria:

- It must be greater than the original word
- It must be the smallest word that meets the first condition

## Example

$w = \text{abcd}$

The next largest word is **abdc**.

Complete the function *biggerIsGreater* below to create and return the new string meeting the criteria. If it is not possible, return `no answer`.

## Function Description

Complete the *biggerIsGreater* function in the editor below.

*biggerIsGreater* has the following parameter(s):

- *string w*: a word

## Returns

- *string*: the smallest lexicographically higher string possible or `no answer`

## Input Format

The first line of input contains  $T$ , the number of test cases.  
Each of the next  $T$  lines contains  $w$ .

## Constraints

- $1 \leq T \leq 10^5$
- $1 \leq \text{length of } w \leq 100$
- $w$  will contain only letters in the range `ascii[a..z]`.

## Sample Input 0

```
5
ab
bb
hefg
dhck
dkhc
```

## Sample Output 0

```
ba
no answer
hegf
dhkc
hcdk
```

## Explanation 0

- *Test case 1:*  
`ba` is the only string which can be made by rearranging `ab`. It is greater.
- *Test case 2:*  
It is not possible to rearrange `bb` and get a greater string.
- *Test case 3:*  
`hegf` is the next string greater than `hefg`.
- *Test case 4:*  
`dhkc` is the next string greater than `dhck`.
- *Test case 5:*  
`hcdk` is the next string greater than `dkhc`.

## Sample Input 1

```
6
lmno
dcba
dcbb
abdc
abcd
fedcbabcd
```

## Sample Output 1

```
lmon
no answer
no answer
acbd
abdc
fedcbabdc
```