2005 South Central USA Regional Programming Contest

Defining Moment

Introduction:

As a homework assignment, you have been tasked with creating a program that provides the meanings for many different words. As you dislike the idea of writing a program that just prints definitions of words, you decide to write a program that can print definitions of many variations of just a handful of different root words. You do this by recognizing common prefixes and suffixes. Since your program is smart enough to recognize up to one prefix and one suffix per word, it can process many forms of each word, significantly reducing the number of rote definitions required.

For this problem, you'll be writing the prefix/suffix processing portion of the program.

Valid prefixes and their meanings are:

anti<word>	against <word>
post<word>	after <word>
pre<word>	before <word>
re<word>	<word> again
un<word>	not <word>

Valid suffixes and their meanings are:

<word>er	one who <word>s
<word>ing	to actively <word>
<word>ize	change into <word>
<word>s	multiple instances of <word>
<word>tion	the process of <word>ing

Note that suffixes are tied more tightly to their root word and should therefore be expanded last. For example, the word "vaporize" would be expanded through the following steps:

```
unvaporize
not vaporize
not change into vapor
```

Of course, the definitions are not exactly right, but how much polish does the professor expect for a single homework grade?

Input:

Input to this problem will begin with a line containing a single integer *n* indicating the number of words to define. Each of the following *n* lines will contain a single word. You need to expand at most one prefix and one suffix, and each word is guaranteed to have a non-empty root (i.e., if the prefix and/or suffix are removed, a non-empty string will remain). Each word will be composed of no more than 100 printable characters.

Output:

For each word in the input, output the expanded form of the word by replacing the prefix and/or suffix with its meaning.

Sample Input:

```
6
vaporize
prewar
recooking
root
repopularize
uninforming
```

Sample Output:

```
change into vapor
before war
to actively cook again
root
change into popular again
not to actively inform
```

The statements and opinions included in these pages are those of *Hosts of the South Central USA Regional Programming Contest* only. Any statements and opinions included in these pages are not those of *Louisiana State University* or the *LSU* Board of Supervisors.