Dickinson 2016: Camel Case 123

Betty has been programming for a long time. For most of her career, she's worked in C and C++, and given her variables names like file_name -- all lowercase, with underscores separating words.

However, when Betty gave Java a try, she became fond of the naming convention she discovered there -- camelcase. So, she now gives her variables names like fileName, starting lowercase, and using uppercase letters to identify the start of subsequent words.

Betty wants to write a program that will convert all of the variable names in her old source code from the C convention to the Java convention. Actually, she'd rather you wrote that program for her.

Write a program that reads text from standard input, replacing any word that contains underscores with an equivalent, "camelcased" string, while leaving the rest of the text unchanged. Here are the rules:

- For simplicity, the input will contain only alphabetic characters, spaces, newlines, and the underscore. The input will be a series of lines of such text. Each line will contain 0 or more words, with a single space separating successive words, and no spaces at the beginning or end of a line. Each line (including the last) ends with a newline character.
- Do not change spacing or line endings. Do not alter any word that doesn't contain an underscore.
- If a word contains 1 or more underscores, remove them, and modify the case of the letters in the string according to the rules of camelcase --- any letter that immediately followed the underscore should be capitalized. Every other letter, including the first letter in the word, should be lowercase. For example, `fancy_var_name` should become `fancyVarName`; `Badly_spelled_vaR` should become `badlySpelledVar`.
- You can presume that words never contain multiple underscores in a row; Betty would never write `weird___var`.
- However, she did occasionally end a variable name with an underscore, and sometimes (egads!) she even started with an underscore. In both of these special cases, you should remove the underscore, convert all the characters to lowercase, and then apply the rules above as needed. So, `terrible_` should become `terrible`, `_Horrible` should become `horrible`, and `_no_good_very_bad_` should become `noGoodVeryBad`.
- Betty can't recall if an underscore might appear by itself (i.e., with no alphabetic characters alongside it). If so, just leave it there unchanged.

Input Format

The input is some unknown number of lines of text, terminated by end-of-file, that follow the rules enumerated above.

Output Format

The output will be identical to the input, altered as described above.

Sample Input

```
This is some_text that
_SHOULD BE_ transformed into
cAMEl_cASE right away
```

Sample Output

```
This is someText that
should be transformed into
camelCase right away
```