Problem H. HaUI UUCP

Input file:	`standard input`
Output file:	`standard output`
Time limit:	10 seconds
Memory limit:	128 megabytes

Forty some odd years ago, email (as we know it today) was sent using a protocol called UUCP (UNIX-toUNIX Copy Program). This required the sender of a mail message to know every machine name along the path to the recipient, and to specify those machine names (or hops) in the recipients UUCP mail address. This type of ancient email address was known as the bang path. The bang path was a list of machine names separated by exclamation points, ending with the recipient's account name on the last machine. An example of this would be:

$$texasam!rice!baylor!csdept!bresearch!bpoucher$$

This address told the sender's system to send the mail to the "texasam" machine. The "texasam" machine would then send it to the "rice" machine. "rice" would send it to the "baylor" machine. "baylor" would send it to "csdept" who would send it to "bresearch". When it reached "bresearch" (the last machine), it would deliver it to user "bpoucher" on that machine. Any error along the way (unknown machine, machine down, etc) was supposed to send an email back to the sender indicating what the issue was (good luck with that). Bang paths of eight or ten machines were not uncommon back in the day. The onus of the email routing was on the sender since they had to know how to get from their machine to the recipient's machine using as many hops as necessary to get there. Often, machines communicated with one another using dial-up modems once or twice a day (there was no internet yet), so it could take days before an email was received by the recipient.

While it was common to have several machines on the UUCP network with the same name, it was not permitted to have two machines with the same name talk to a common machine. That is, using the example above, the "baylor" machine could not directly communicate with two machines named "csdept" for obvious reasons. Sometimes, inexperienced users would have "loops"in their bang path:

$$texasam!rice!baylor!csdept!baylor!rice!dev!bresearch!bpoucher$$

While this would work, it is inefficient (and delays the email) since the "rice"machine would send the email to "baylor" who would send it to "csdept" who would send it back to "baylor" who would send it to back to "rice" who would then send it to "dev". The steps of sending to "csdept!baylor!rice" could be left out with the same net effect, and the email would get to the recipient faster. In addition, a machine may forward an email to itself:

$$texasam!Rice!rice!rice!rice!rice!bpoucher$$

in this case, all but one of the "rice" hops can be left out: $texasam!Rice!bpoucher$

For this problem, you will read a bang path, remove any unnecessary hops and output the new, possibly shorter, bang path.

Input

The single line of input contains a string which is the valid bang path to process. The string will be at least one character long and no longer than 256 characters. Each component of the bang path (including the user name at the end) will be between 1 and 10 alpha-numeric characters. Components are separated by a single exclamation point (!). Note that machine names and user names are case-insensitive, but case-preserving. If the bang path does not contain any exclamation points, then the single component specifies a user name on the sender's machine

Output

Output consists of a single line which is a string that represents the possibly shorter bang path of the input string. As shown in sample 2, the case of a machine (baYlor) must be preserved, but it does not matter which one you chose if one or more of the same machine are eliminated.

Examples

standard input	standard output
texasam!rice!baylor!csdept!baylor!rice!dev!bresearch!bpoucher	texasam!rice!dev!bresearch!bpoucher
texasam!Rice!baYlor!csdept!BayloR!dev!Rice!bresearch!bpoucher	texasam!Rice!baYlor!dev!Rice!bresearch!bpoucher
bresearch!bpoucher	bresearch!bpoucher