Problem I
An I for an Eye

Ken has been having trouble lately staying under the word limit in Twitter, so he's decided to write a little front-end program which will take in text and shorten it using a fixed set of abbreviations for commonly used letter sequences. Those abbreviations are shown in the table below:

The character ...	... substitutes for the letter sequence
@	at
&	and
1	one, won
2	to, too, two
4	for, four
b	bea, be, bee
c	sea, see
i	eye
o	oh, owe
r	are
u	you
y	why

Ken is about to start writing this program when he realizes that the extent of his computer knowledge is ... well ... using Twitter. He's looking for someone to help him – `r u th@ some1`?

Input

Input starts with a single integer n indicating the number of lines of text to process. Following this are n lines of text. Each line will contain only alphanumeric characters and spaces, and each line will have at least one non-space character. Each line has at most 200 characters.

Output

Display each line with the appropriate substitutions made. Substitutions should also be made inside words, e.g., the word `that` should be changed to `th@`. If two letter sequences overlap (like `at` and `to` in the word `baton`) just replace the first one (in this case resulting in `b@on`). If two letter sequences start at the same location (like `be` and `bee` in `been`) replace the longer one (in this case resulting in `bn`). If the letter sequence starts with an upper-case letter, then the abbreviation should also be in upper-case (if appropriate). Finally, no substituted letter should later be part of another substitution. For example, if the input is `oweh`, you would first replace the `owe` with an `o` to get `oh`. At this point you do NOT replace the `oh` with an `o` since the `oh` contains a substituted letter.

Sample Input 1

```
3
Oh say can you see
I do not understand why you are so cranky just because Karel won
Formation
```

Sample Output 1

```
O say can u c
I do not underst& y u r so cranky just bcause Krl 1
4m@ion
```