Problem ID: crackingthecode
CPU Time limit: 1 second
Memory limit: 1024 MB

Source: Benelux Algorithm Programming Contest (BAPC) 2013

Problem D
Cracking the Code

A terrorist organisation known as the New World Ensemble for Rebellious Coders (NWERC) is a menace to our society. Fortunately, we have found a way to intercept their communications without them knowing it. There is a problem however, since their messages are encrypted.

What we do know from intelligence is that their messages consist of lowercase letters only, and that the encryption method is the Basic Alphabet Permutation Code (BAPC). In this code, every same letter is replaced with another (lowercase) letter from the alphabet (or possibly the same letter). Obviously, two different letters in the original message are still different after encryption, so that there is a unique decryption. So for example, "`hello`" might be encrypted as "`ifmmp`" or "`holle`", but not as "`cnoiz`" or "`bgrrb`".

Still, this leaves a lot of possibilities, and our efforts to break the code have been in vain so far. Fortunately, we had a stroke of luck: through an informant, we were able to get hold of a decrypted message D. We do not have the corresponding encrypted message, but we are fairly certain that it must be somewhere in a list of intercepted encrypted messages we have.

Your task is to write a program which, given the original message D and a list of encrypted messages, works out which encrypted letter belongs to which decrypted letter (assuming that one of the encrypted messages corresponds to the original message), for as much as is possible.

This should then be used to decrypt a freshly intercepted encoded message X, as much as possible. Specifically, each letter of X for which it can be determined with absolute certainty what it represents – either directly or deductively – should be decrypted. All other letters should be replaced by a question mark.

Note that, even when multiple messages can correspond to D, it may still be possible to decrypt certain letters (see for example the third test case in the sample input).

Input

On the first line one positive number: the number of test cases, at most 100. After that per test case:

- one line with a single integer n ($1 \leq n \leq 100$): the number of encrypted messages.
- n lines with a single string M_i: the encrypted messages.
- one line with a single string D: the decrypted message, assumed to correspond to one of the encrypted messages.
- one line with a single string X: the message to be decoded.

All strings consist of at least 1 and at most 1 000 lowercase letters.

Output

Per test case:

- one line with a single string Y. For each letter in X, the corresponding letter in Y should be the decrypted counterpart, or a '`?`' if it is unknown. If there is no encrypted message M_i that could possibly be the encrypted version of D, then the string should be "`IMPOSSIBLE`" instead.

Sample Input 1

```
3
3
mtahovcjqxels
irajsbkticlur
gnubipwdgkryf
aboringsample
rbunyfka
2
ejotydins
xchmrwbcxg
decrypted
dmvenw
2
abccdeb
afccdgf
message
abcdefg
```

Sample Output 1

```
problem?
IMPOSSIBLE
m?sa???
```