Vigenère Cipher Analysis

`analyse.c`, `analyse.C`, `analyse.java`

In this problem set, there is another problem (`vigenere`) asking you to implement the *Vigenère Cipher* encryption algorithm. This time, we will demonstrate one of the caveats of that cipher.

A secret organization Amateur Codebreakers Movement has a strong suspicion that bank robbers are planning another strike soon. Unfortunately, we do not know neither the name of the bank nor the exact day and time. ACM is able to eavesdrop the communication between robbers and their driver but the communication is encrypted using the Vigenère Cipher.

Your task is to try to break the cipher. You are given two words that are likely to appear in the original plaintext — so-called *cribs* (such words played an important role, for example, in breaking the famous Enigma code).

Input Specification

(For the specification of the Vigenère cipher, please refer to the `vigenere` problem.)

The input contains several instances. Each instance consists of four lines — the first line is an integer number K, $1 \leq K \leq 100$, the maximum length of the encryption key to be considered. The second and third lines contain the cribs W_1 and W_2, $1 \leq K \leq length(W_i) \leq 100$. The fourth line is the ciphertext C, $1 \leq length(C) \leq 100\,000$. Both the cribs W_1, W_2 and the ciphertext C consist only of uppercase letters of the standard English alphabet $\{A, B, C, ..., Z\}$. The input is terminated by a line containing one zero.

Output Specification

Your program must determine how many different plaintexts there exist that contain *both* of the given cribs *simultaneously* inside the same message and that will result into the given ciphertext using the Vigenère Cipher with some key Q, $1 \leq length(Q) \leq K$.

Print one line for each input instance:

- If there is exactly one plaintext satisfying all conditions, output that plaintext with no additional spaces.
- If two or more such plaintexts exist, print the word “`ambiguous`”.
- If there is no such plaintext, print “`impossible`”.

Sample Input

```
4
BANK
MONEY
FTAGUAVMKILCKPRIJCHRJZIYUAXFNBSLNNXMVDVPXLERWDSL
5
SECOND
PARSEC
SUKCTZHYYES
3
ACM
IBM
JDNCOFBEN
4
ABCD
EFGH
OPQRHKLMN
0
```

Output for Sample Input

```
WEWILLROBTHEBANKANDTAKEALLTHEMONEYTOMORROWATNOON
impossible
ambiguous
EFGHXABCD
```