UVa Online Judge

13258 Romeo and Juliet Secrets

The Capulets and the Montagues hate each other to death and will never accept the love of Romeo and Juliet. This is why the two young lovers must hide their affair by making phone chats unintelligible to their parents. They do this by altering the messages in their chat using the following three-step process:

1. First, they remove spaces and punctuation marks, and make every letter lowercase.
2. Then, they choose a word W (e.g., their names) they want to hide.
3. Finally, they replace in every occurrence of W a substring of length K with that many random letters.

For example, assume Juliet wants to hide the word *Romeo* in the message

O Romeo, Romeo! Wherefore art thou Romeo?

After the first step, the encrypted message will be

oromeoromeowhereforeartthouromeo

Note that the word *Romeo* appears at positions 2, 7, and 28. If $K = 2$, then in each one of these occurrences a substring of length two must be replaced with exactly two random letters. One possible encrypted version of the original message under this scheme could be

oxwmeorozkowhereforeartthouromeo

Unfortunately, Lord Capulet is aware of Romeo and Juliet secret, and he has intercepted one of Juliet's messages. He needs your help in writing a program to find a given word in the intercepted message, even if that word was altered using the secret scheme. Specifically, Lord Capulet wants to know how many times the given word could appear in the original message.

Input

The first line of input contains the number of test cases. Each test case consists of three lines:

- A line containing a string T: the message intercepted by Lord Capulet ($1 \leq |T| \leq 10^5$).
- A line containing a string W: the word Lord Capulet wants to find in T ($1 \leq |W| \leq |T|$).
- A line containing integer K ($1 \leq K \leq |W|$).

Both T and W will exclusively consist of lowercase letters of the English alphabet ('`a`', ..., '`z`').

Output

For each test case, print a single line indicating the number of occurrences of word W in message T, including all possible encrypted versions of W. Occurrences that overlap must be counted separately.

Sample Input

```
3
oxwmeorozkowhereforeartthouromeo
romeo
2
abcabcabcabcabc
aba
1
abcabcabcabcabc
acb
2
```

Sample Output

```
3
5
9
```