Code Jam 2009 - Round 3

Alphabetomials

Problem

As we all know, there is a big difference between polynomials of degree 4 and those of degree 5. The question of the non-existence of a closed formula for the roots of general degree 5 polynomials produced the famous Galois theory, which, as far as the author sees, bears no relation to our problem here.

We consider only the multi-variable polynomials of degree up to 4, over 26 variables, represented by the set of 26 lowercase English letters. Here is one such polynomial:

```
aber+aab+c
```

Given a string *s*, we evaluate the polynomial on it. The evaluation gives *p*(*S*) as follows: Each variable is substituted with the number of appearances of that letter in *S*.
For example, take the polynomial above, and let *S* = "abracadabra edgar". There are six a's, two b's, one c, one e, and three r's. So

```
p(S) = 6 * 2 * 1 * 3 + 6 * 6 * 2 + 1 = 109.
```

Given a dictionary of distinct words that consist of only lower case letters, we call a string *S* a *d-phrase* if

$$S = "S_1\ S_2\ S_3\ \ldots\ S_d",$$

where S_i is any word in the dictionary, for $1 \le i \le d$. i.e., *S* is in the form of *d* dictionary words separated with spaces. Given a number **K** ≤ 10, your task is, for each 1≤ *d* ≤ **K**, to compute the sum of *p*(*S*) over all the *d*-phrases. Since the answers might be big, you are asked to compute the remainder when the answer is divided by 10009.

Input

The first line contains the number of cases **T**. **T** test cases follow. The format of each test case is:
A line containing an expression *p* for the multi-variable polynomial, as described below in this section, then a space, then follows an integer **K**.
A line with an integer **n**, the number of words in the dictionary.
Then **n** lines, each with a word, consists of only lower case letters. No word will be repeated in the same test case.

We always write a polynomial in the form of a sum of terms; each term is a product of variables. We write a^t simply as *t a*'s concatenated together. For example, a^2b is written as *aab*. Variables in each term are always lexicographically non-decreasing.

Output

For each test case, output a single line in the form

$$\texttt{Case \#X: } sum_1\ sum_2\ \ldots\ sum_K$$

where X is the case number starting from 1, and sum_i is the sum of $p(S)$, where S ranges over all i-phrases, modulo 10009.

Limits

Memory limit: 1 GB.
1 ≤ **T** ≤ 100.
The string p consists of one or more terms joined by '+'. It will not start nor end with a '+'. There will be at most 5 terms for each p. Each term consists at least 1 and at most 4 lower case letters, sorted in non-decreasing order. No two terms in the same polynomial will be the same. Each word is non-empty, consists only of lower case English letters, and will not be longer than 50 characters. No word will be repeated in the same dictionary.

Small dataset

Time limit: 30 seconds.
1 ≤ **n** ≤ 20
1 ≤ **K** ≤ 5

Large dataset

Time limit: 60 seconds.
1 ≤ **n** ≤ 100
1 ≤ **K** ≤ 10

Sample

Sample Input

```
2
ehw+hwww 5
6
where
when
what
whether
who
whose
a+e+i+o+u 3
4
apple
orange
watermelon
banana
```

Sample Output

```
Case #1: 15 1032 7522 6864 253
Case #2: 12 96 576
```