Let’s first define some terms:

- A string is palindromic if it reads the same forward and backward. Examples of palindromes are *madam* and *toot*.

- A string is a dromicpalin if we can rearrange its letters to make it a palindrome. An example of a dromicpalin string is *mmaad* because we can rearrange the letters to make it *madam*, which is a palindrome.

- A substring is any contiguous sequence of characters of a string. Some substrings of ‘acmicpc’ are ‘a’, ‘c’, ‘i’, ‘icp’, ‘acmicpc’ but ‘acpc’ is not a substring. For this problem, we are not considering the empty substring, so that means there are \( n(n + 1) \) over 2 substrings of a string of length \( n \).

Given a string, you have to figure out how many of its substrings are dromicpalin.

**Input**

The first line of input is an integer \( T \) \((T < 100)\) indicating the number of test cases. Each case is a line containing a string. The strings will contain only lowercase letters [a - z]. The length of each string will be positive and not greater than 1000.

**Output**

For each case, first output the case number followed by the number of substrings that are dromicpalin. Follow the samples for exact format. There is no new-line between cases.

**Sample Input**

```
4
acmicpc
aaaaa
isyoursolutionfastenough
ababababababa
```

**Sample Output**

```
Case 1: 8
Case 2: 15
Case 3: 24
Case 4: 67
```