A seaport container terminal stores large containers that are eventually loaded on seagoing ships for transport abroad. Containers coming to the terminal by road and rail are stacked at the terminal as they arrive.

Seagoing ships carry large numbers of containers. The time to load a ship depends in part on the locations of its containers. The loading time increases when the containers are not on the top of the stacks, but can be fetched only after removing other containers that are on top of them.

The container terminal needs a plan for stacking containers in order to decrease loading time. The plan must allow each ship to be loaded by accessing only topmost containers on the stacks, and minimizing the total number of stacks needed.

For this problem, we know the order in which ships must be loaded and the order in which containers arrive. Each ship is represented by a capital letter between A and Z (inclusive), and the ships will be loaded in alphabetical order. Each container is labeled with a capital letter representing the ship onto which it needs to be loaded. There is no limit on the number of containers that can be placed in a single stack.

Input

The input file contains multiple test cases. Each test case consists of a single line containing from 1 to 1000 capital letters representing the order of arrival of a set of containers. For example, the line ABAC means consecutive containers arrive to be loaded onto ships A, B, A, and C, respectively. When all containers have arrived, the ships are loaded in strictly increasing order: first ship A, then ship B, and so on.

A line containing the word `end` follows the last test case.

Output

For each input case, print the case number (beginning with 1) and the minimum number of stacks needed to store the containers before loading starts. Your output format should be similar to the one shown here.

Sample Input

```
A
CBACBACBACBACBA
CCCCBBBBAAAA
ACMICPC
end
```

Sample Output

```
Case 1: 1
Case 2: 3
Case 3: 1
Case 4: 4
```