

Problem 4 – Decrypt the Messages

You are working for a company which is very concerned about its information and communication. For this reason, they have invented an internal approach to communication between different departments – they are communicating to each other via **messages, which are reversed (written backwards) and then encrypted**. In order to be read and understood, each message has to be decrypted. Your task is to write a program, which **receives all encrypted messages** in a specific communication, **decrypts them** and **prints all decrypted messages at the console as well as the total number of messages** that have been received.

At the beginning of a communication, you will receive either the keyword **“START”** (upper case) or **“start”** (lower case), which indicates that you will **start receiving reversed and encrypted messages**. At the end of the communication, you will receive either the keyword **“END”** (upper case) or **“end”** (lower case), which indicates that the communication is over and you need to **show the decrypted messages’ content and total count**. Any **non-empty string** between the **“start”** and **“end”** keywords is considered a message. If **no messages have been received** between the **“start”** and the **“end”** keywords, you should print on the console: **“No message received.”**

All messages are case-sensitive and consist of **letters, digits, as well as some special characters** – ‘+’, ‘%’, ‘&’, ‘#’ and ‘\$’. Letters **from A to M** are **converted** to letters **from N to Z** (A → N; B → O; ... M → Z) and letters **from N to Z** are **converted** to letters **from A to M** (N → A; O → B; ... Z → M). The **converted** letter should keep the **case** of the **original** letter. The **special characters** are converted in the following way: ‘+’ is converted to a **single space** (‘ ’), ‘%’ is converted to a **comma** (‘,’), ‘&’ is converted to a **dot** (‘.’), ‘#’ is converted to a **question mark** (‘?’) and ‘\$’ is converted to an **exclamation mark** (‘!’). The **digits** (0-9) are **not converted** and stay the same.

For example, you receive the following message – **“\$1+rtnffrz+greprF”** and you start decrypting it. Convert the 1st character ‘\$’ to ‘!’, then the 2nd character – ‘1’ stays the same, then convert the 3rd character – ‘+’ to single space ‘ ’, ‘r’ → ‘e’, ‘t’ → ‘g’, ‘n’ → ‘a’, ‘f’ → ‘s’, ‘f’ → ‘s’, ‘r’ → ‘e’, ‘z’ → ‘m’, ‘+’ → ‘ ’, ‘g’ → ‘t’, ‘r’ → ‘e’, ‘e’ → ‘r’, ‘p’ → ‘c’, ‘r’ → ‘e’, ‘F’ → ‘S’. After decrypting all letters, the message is: **“!1 egassem terceS”** and when you reverse it, you get the original message: **“Secret message 1!”**

Input

The input data should be read from the console. The input will contain a random number of lines. The line that holds the **keyword “START” or “start”** will always be before the line that holds the **keyword “END” or “end”**. The input data will always be valid and in the format described. There is no need to check it explicitly.

Output

The output data should be printed on the console.

- On the **first line** print the total number of messages that have been received in format: **“Total number of messages: N”** – where N is the number of received and decrypted messages.
- On the next N lines print the decrypted messages.
- If **no messages have been received** between the **“start”** and the **“end”** keywords, you should **print on the console** only one line holding: **“No message received.”**

Constraints

- The **number of messages** between the **“start”** and the **“end”** keywords will be between 0 and 100.
- The **length of each message** will be between 1 and 1000 symbols.
- Each encrypted message will contain only Latin letters, digits and the special symbols described above.
- Allowed working time: 0.1 seconds. Allowed memory: 16 MB.

Examples

Input	Comments	Output
START \$\$\$byrU END	We start conversion from the 1 st character: \$ → !, \$ → !, \$ → !, b → o, y → l, y → l, r → e, U → H and reverse the newly received string “!!!olleH” to the original message “Hello!!!”	Total number of messages: 1 Hello!!!

Input	Comments	Output
start tsrqpon 1rtnFFrz end	We start conversion from the 1 st character: t → g, s → f, r → e, q → d, p → c, o → b, n → a and reverse the newly received string “gfedcba” to the original message “abcdefg”. Then we do the same for the second message.	Total number of messages: 2 abcdefg meSSage1

Input	Comments	Output
start END	There is no message received.	No message received.

Input	Output
Normal communication message. START \$rtnffrz+tavjbybs+rug+gclepar+bg+leg+%rfnryC end	Total number of messages: 1 Please, try to encrypt the following message!