

## Problem 4 – Encrypt 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**. Your task is to write a program, which **encrypts all messages** in a specific communication, **prints them at the console as well as the total number of messages** that have been sent.

**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 encrypted messages’ content and total count**. Any **non-empty string** between the **“start”** and **“end”** keywords is considered a message. If **no messages have been sent** between the **“start”** and the **“end”** keywords, you should print on the console: **“No messages sent.”**

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: ‘ ’ (space) is converted to a **plus sign** (‘+’), ‘;’ is converted to ‘%’, ‘:’ is converted to ‘&’, ‘?’ is converted to ‘#’ and ‘!’ is converted to ‘\$’. **Digits** (0-9) are **not converted** and stay the same.

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

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

### 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 unencrypted 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 Hello!!! END	We start conversion from the 1 <sup>st</sup> character: ! → \$, ! → \$, ! → \$, o → b, l → y, l → y, e → r, H → U and reverse the newly received string "Uryyb\$\$\$" to the encrypted message "\$\$\$byyrU"	Total number of messages: 1 \$\$\$byyrU

Input	Output
START abcdefg meSSage1 end	Total number of messages: 2 tsrqpon 1rtnFFrz

Input	Output
start  END	No messages sent.

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