SWE2004: Principles in Programming (Spring 2013)

Programming Lab #9
Due-date : May 16th, 11:59 PM

Everybody now sends SMS ("text") messages from phone to phone. On many phones, the procedure for sending a message is to press the numeric key associated with the desired letter as many times as necessary in order to select that letter. For example, to create the letter "N", one would press the "6" twice – the first selects "M" and the second press changes the "M" into an "N". A space between words is created by pressing the "#" once.

This procedure is sufficient for a word like "BET" where one can first press "2" twice, then "3" twice, then "8". A complicating factor is when the adjacent letters use the same number, as in "CAB". It is necessary to press "2" three times for the "C", then wait a certain time, then press "2" once for the "A", wait a certain time, then press "2" twice for the "B". Let's refer to this waiting as a "pause" in the buttons, and the notation "222P-2P-22" stands for the word "CAB". There is a dash between the keys necessary for each letter, and there is a "P" to indicate the necessary pause.

HELLO	44-33-555P-555-666
THIS IS A MESSAGE	8-44P-444-7777-#-444-7777-#-2-#-6-33-7777P-7777-2-4-33

1	2 ABC	3 DEF
4 GHI	5 JKL	6 MNO
7 PQRS	8 TUV	9 WXYZ
*	0	# space

The problem

Write a text message encoder/decoder. When presented with a plain text message, the program should print the text message equivalent. When presented with a text message, the program should
print the plain text version. The program can determine whether to convert to or from text messages based on the initial character of the message; a number or "#" should convert from

text format to plain format, and a letter or space should convert from plain text to text format. In the latter case, the input is guaranteed to be upper case letters and will contain no digits.

Input

There will be multiple lines in the input. Each line represents one message, and each message is to be read, converted as necessary, and displayed. The last line in the input will start with a "*" and indicates that the program should terminate.

You may assume all of the input is valid. There will be no invalid characters or tab characters in the input, and there will be no trailing spaces at the end of a line. Plain text lines will contain only uppercase alphabetic characters, space characters, and the end of line.

Output

For each input line, create the appropriate output line.

Sample Input

```
THIS IS A MESSAGE
8-33-99-8
TEXT
*
```

Sample Output

```
8-44P-444-7777-#-444-7777-#-2-#-6-33-7777P-7777-2-4-33
TEXT
8-33-99-8
```