Live on No Evil

`evilive.p`, `evilive.c`, `evilive.C`

One of the Czech ministers wants to run for the chair in the European Parliament. Now he is preparing his election campaign, which is based on the following simple program: that people should behave well, be nice to each other, and live on* good business only. For this purpose, he needs some mottos in English language. For example, the motto he likes most is "Live on no evil" because of its special property: it says the same thing when read backwards! Such strings are called *palindromes*.

The minister wants to find more palindromic sentences and mottos, such as "Rats live on no evil star", "Was it a car or a cat I saw?", etc. You are to write a computer program which is able to detect not only palindromes but also so called mirrored strings. With that program, he can produce more great mottos and then easily win the election.

A *regular palindrome* is a string of numbers or letters that is the same forward as backward. For example, the string "ABCDEDCBA" is a palindrome because it is the same when the string is read from left to right as when the string is read from right to left.

A *mirrored string* is a string for which when each of the elements of the string is changed to its reverse (if it has a reverse) and the string is read backwards the result is the same as the original string. For example, the string "3AIAE" is a mirrored string because "A" and "I" are their own reverses, and "3" and "E" are each others' reverses.

A *mirrored palindrome* is a string that meets the criteria of a regular palindrome and the criteria of a mirrored string. The string "ATOYOTA" is a mirrored palindrome because if the string is read backwards, the string is the same as the original and because if each of the characters is replaced by its reverse and the result is read backwards, the result is the same as the original string. Of course, "A", "T", "O", and "Y" are all their own reverses.

A list of all valid characters and their reverses is as follows. Note that "`0`" (zero) and "`O`" (the letter) are considered the same character and therefore *only* the letter "`O`" is a valid character.

Char	**Rev.**	**Char**	**Rev.**	**Char**	**Rev.**	**Char**	**Rev.**	**Char**	**Rev.**
A	A	H	H	O	O	V	V	3	E
B		I	I	P		W	W	4	
C		J	L	Q		X	X	5	Z
D		K		R		Y	Y	6	
E	3	L	J	S	2	Z	5	7	
F		M	M	T	T	1	1	8	8
G		N		U	U	2	S	9	

Input Specification

Input consists of strings (one per line) each of which will consist of one to twenty valid characters. There will be no invalid characters in any of the strings. Your program should read to the end of file.

*live on — žít z (čeho)

Output Specification

For each input string, you should print the string starting in column 1 immediately followed by exactly one of the following strings:

String	**Criteria:** if the string ...
“ `-- is not a palindrome.`”	... is not a palindrome and is not a mirrored string
“ `-- is a regular palindrome.`”	... is a palindrome and is not a mirrored string
“ `-- is a mirrored string.`”	... is not a palindrome and is a mirrored string
“ `-- is a mirrored palindrome.`”	... is a palindrome and is a mirrored string

Note that the output line is to include the dashes and spacing exactly as shown in the table above and demonstrated in the example below.

Sample Input

```
NOTAPALINDROME
ISAPALINILAPASI
A3MEA
ATOYOTA
```

Output for Sample Input

```
NOTAPALINDROME -- is not a palindrome.
ISAPALINILAPASI -- is a regular palindrome.
A3MEA -- is a mirrored string.
ATOYOTA -- is a mirrored palindrome.
```

P.S.: You do not need to make your own program palindromic, although such programs definitely have a better aesthetic value. See below for a small example.

```
                                 char rahc
                                    [ ]
                                     =
                                   "\n/"
                                     ,
                                 redivider
                                    [ ]
                                     =
                        "RATS LIVE ON NO EVIL STAR"
                                     ,
                                     *
                              deliver,reviled
                                     =
                                    1+1
                                     ,
                                niam ; main
                                    ( )
                                   {/*\}
                                    \*/
                                  int tni
                                     =
                                    0x0
                                     ,
                              rahctup,putchar
                                    ( )
                            ,LACEDx0 = 0xDECAL,
                                 rof ; for
                              (;(int) (tni);)
                                (int) (tni)
                           = reviled ; deliver =
                                 redivider
                                     ;
for ((int)(tni)++,++reviled;reviled* *deliver;deliver++,++(int)(tni)) rof
                                     =
                              (int) -1- (tni)
                           ;reviled--;--deliver;
                              (tni)  =  (int)
                           - 0xDECAL + LACEDx0 -
                                 rof ; for
        (reviled--,(int)--(tni);(int) (tni);(int)--(tni),--deliver)
                             rahctup = putchar
                            (reviled* *deliver)
                                     ;
                             rahctup * putchar
                             ((char) * (rahc))
                                     ;
                                    /*\
                                   {\*/}
```