Problem D: Is the Name of This Problem

Source file: `quine.{c, cpp, java}`
Input file: `quine.in`

The philosopher Willard Van Orman Quine (1908–2000) described a novel method of constructing a sentence in order to illustrate the contradictions that can arise from self-reference. This operation takes as input a single phrase and produces a sentence from that phrase. (The author Douglas R. Hofstadter refers to this process as *to Quine a phrase*.) We can define the Quine operation like so:

```
Quine(A) = "A" A
```

In other words, if A is a phrase, then Quine(A) is A enclosed in quotes ("), followed by a space, followed by A. For example:

```
Quine(HELLO WORLD) = "HELLO WORLD" HELLO WORLD
```

Below are some other examples of sentences that can be created by the Quine operation. Note that Quining allows sentences to be indirectly self-referential, such as the last sentence below.

```
"IS A SENTENCE FRAGMENT" IS A SENTENCE FRAGMENT
"IS THE NAME OF THIS PROBLEM" IS THE NAME OF THIS PROBLEM
"YIELDS FALSEHOOD WHEN QUINED" YIELDS FALSEHOOD WHEN QUINED
```

Your goal for this problem is to take a sentence and decide whether the sentence is the result of a Quine operation.

Input: The input will consist of a sequence of sentences, one sentence per line, ending with a line that has the single word, **`END`**. Each sentence will contain only uppercase letters, spaces, and quotation marks. Each sentence will contain between 1 and 80 characters and will not have any leading, trailing, or consecutive spaces.

You must decide whether each sentence is the result of a Quine operation. To be a Quine, a sentence must match the following pattern *exactly*:

1. A quotation mark
2. Any nonempty sequence of letters and spaces (call this phrase *A*)
3. A quotation mark
4. A space
5. Phrase *A*—exactly as it appeared in (2)

If it matches this pattern, the sentence is a Quine of the phrase *A*. Note that phrase A must contain the exact same sequence of characters both times it appears.

Output: There will be one line of output for each sentence in the data set. If the sentence is the result of a Quine operation, your output should be of the form, `Quine(A)`, where *A* is the phrase to Quine to create the sentence.

If the sentence is not the result of a Quine operation, your output should be the phrase, **`not a quine`**.

Example input:	**Example output:**

```
"HELLO WORLD" HELLO WORLD
"IS A SENTENCE FRAGMENT" IS A SENTENCE FRAGMENT
"IS THE NAME OF THIS PROBLEM" IS THE NAME OF THIS PROBLEM
"YIELDS FALSEHOOD WHEN QUINED" YIELDS FALSEHOOD WHEN QUINED
"HELLO" I SAID
WHAT ABOUT "WHAT ABOUT"
" NO EXTRA SPACES " NO EXTRA SPACES
"NO"QUOTES" NO"QUOTES
""
END
```

```
Quine(HELLO WORLD)
Quine(IS A SENTENCE FRAGMENT)
Quine(IS THE NAME OF THIS PROBLEM)
Quine(YIELDS FALSEHOOD WHEN QUINED)
not a quine
not a quine
not a quine
not a quine
not a quine
```

A review of quotation marks in strings: As a reminder, the quotation mark character is a regular character, and can be referred to in C, C++, and Java using the standard single-quote notation, like so:

```
'"'
```

However, to place a quotation mark inside a double-quoted string in C, C++, and Java, you must place a backslash (\) in front of it. If you do not it will be interpreted as the end of the string, causing syntax errors. For example:

```
"This quotation mark \" is inside the string"
"\""
"\"SAID SHE\" SAID SHE"
```