4443 N Questions

(This problem uses an interactive server.)

The question and answer game "20 Questions" involves two people, one who chooses some thing, and the guesser who asks yes-or-no questions, trying to narrow down the possibilities and finally deduce what the thing is. If the guesser determines the thing, he wins; if not, he loses.

You are to write a program that plays the guesser's part of asking questions. The server replies '`Y`' (yes) or '`N`' (no). Rather than always requiring 20 questions, your program will be given the highest number of questions, N, that can be asked. It will always be possible to determine the thing within the allotted number of questions.

The server and your client program share the same set of knowledge. This knowledge includes a table of things, the questions that can be asked, and the yes-no answers to those questions. Your program may ask up to N questions to deduce the thing that the server has chosen.

Program Interaction

Your program must converse with a server, issuing queries to standard output and receiving responses through standard input. Your client starts the conversation by requesting the shared body of knowledge from the server. Next, your program issues a series of queries to determine the thing that the server has chosen.The first command issued to the server must be a '`C`' command followed by end-of-line. The server responds with multiple sections of line-oriented information:

- The first section is a list of things, one per line. This list of things is terminated by a line containing only a comma. The comma is not a thing.
- Following the list of things is a table of questions and answers, one question/answer combination per line. Each question starts at the beginning of the line, and is terminated by a question mark. One or more spaces follow the question mark. A series of plus and minus signs follow the spaces, corresponding to the things in the list. A '`+`' indicates a yes answer; a '`-`' indicates a no answer. For the sample transaction question '`Is it round?`', the answer is '`Yes`' for a soccer ball, an apple, and the sun. For a hammer, a car, the sky, and a bulldozer, the answer is '`No`'. The question and answer table is terminated by a line with a single period.
- The server responds with a final configuration line containing the decimal integer N, the maximum number of queries that the client can ask before losing the game.

Queries are lines that end with a question mark followed by end-of-line. The queries may either be questions asked verbatim from the table of questions and answers, or a direct inquiry regarding a thing formed by concatenating '`Is it`' with a thing and a question mark '`?`'. The server responds with a single character followed by newline. The character '`Y`' indicates a yes answer and '`N`' a no answer to the query.

It is very important that your program flush standard output after issuing a query to the server. e.g.

C `fputs("C\n",stdout); fflush(stdout);`

C++ `cout << "C\n" << flush;`

Java `System.out.println("C"; System.out.flush();`

There will never be more than 50 things. There will not be any duplicate things. No things will exceed 16 characters in length, excluding newline. There will never be more than 100 questions. There will never be any duplicate questions.

The question formed by '`Is it`' + thing + '`?`' will never duplicate a question from the table. No question will exceed 32 characters in length, excluding the question mark. The server will never respond with a line longer than 85 characters, excluding newline.

A sample transcript follows. Client commands are offset for clarity. The comments to the right of the transcript must not appear in your client commands and will not occur in the server's responses; they are for description only.

Sample Transcript of Game Play

```
                          C                    client request for the set of knowledge
a soccer ball
an apple
a hammer
a car
the sky
a bulldozer
the sun
,                                              end of list of things
Is it red?         -++----
Is it round?       ++----+
Is it a plant?     -+-----
Is it a tool?      --+--+-
Is it blue?        ----+--
Can you touch it? ++++-+-
.                                              end of question/answer lines
4                                              up to four questions may be asked
                          Can you touch it?    client query based upon a question
Y
                          Is it red?
Y
                          Is it an apple?      client query based upon a thing
N
                          Is it a hammer?
Y
```

Judges' Responses

An ill-formed command or query produces a PRESENTATION ERROR. In order for your program to be accepted by the judges, the final question must be '`Is it`' followed by the correct thing and a question mark. Failing to guess the chosen thing within the query limit results in a WRONG ANSWER judgment. Spurious dialog with the server after successfully guessing the chosen thing elicits PRESENTATION ERROR. Finally, if the server has to wait longer than one second (real time) for your client to issue a query, the judged response is TIME LIMIT EXCEEDED. Note that this TIME LIMIT EXCEEDED is the exact same message issued if your program accumulates more than the allotted CPU seconds.

Hints for Testing

You may construct a list of things and a table of questions and answers to play manually. Execute your client program with you, the programmer, acting as the server. Let standard input come from your command line environment. Supply correct responses to your program.

The server process used by the judges is available to the contestants. You may use the server in two ways. In the absence of a working client program, you, the programmer, can act as the client, typing input to the server. Use the following command:

`test4` *configurationFile transcriptFile*
To test your program and its conversation with the server, use the command:

`test4` *configurationFile transcriptFile sourceFile*
where

- *configurationFile* is described below.
- *transcriptFile* is a file that captures the server's inputs and outputs for later analysis. The server's inputs are your client's queries. Client queries are offset by two tab characters to distinguish the client vs. server dialog.
- *sourceFile* is the file name of the source code to your client. When specified, the test4 script calls the compile script.

The server configuration file is almost exactly the same format as its response to the `"C"` command. The only difference is that the configuration file contains one line at the end that the client never sees: the thing that is chosen. The configuration file for the sample transaction is contained reproduced here.

Sample Configuration File

```
a soccer ball
an apple
a hammer
a car
the sky
a bulldozer
the sun
,
Is it red?        -++----
Is it round?      ++----+
Is it a plant?    -+-----
Is it a tool?     --+--+-
Is it blue?       ----+--
Can you touch it? ++++-+-
.
4
a hammer
```