Hotline

Every customer sometimes needs help with new and unusual products. Therefore, hotline service is very important for every company. We need a single phone number where the customer can always find a friendly voice ready to help with anything. On the other hand, many people are needed to serve as hotline operators, and human resources are always very expensive. Moreover, it is not easy to pretend "friendly voice" at 4am and explain to a drunken man that you are really unable to give him the number to House of Parliament. It was also found that some of the questions repeat very often and it is very annoying to answer them again and again.

ACM is a modern company, wanting to solve its hotline problem. They want to decrease the number of human operators by creating a complex software system that would be able to answer most common questions. The customer's voice is analysed by a special Voice Recognition Module (VRM) and converted to a plain text. The text is then taken by an Artificial Automatic Adaptive Answering Algorithm (AAAAA). The most common questions are recognised and answered automatically. The replies are then converted to a sound by Text-to-Speech Module (TTS).

You are to write the AAAAA module. Because your algorithm should be adaptive, it has no explicit knowledge base. But it must be able to listen to sentences in English and remember the mentioned facts. Whenever the question is asked about such a fact, the system has to answer it properly. The VRM and TTS modules are already implemented, so the input and output of AAAAA will be in the text form.

Input

There is a single positive integer *T* on the first line of input. It stands for the number of dialogues to follow. Each dialogue consists of zero or more lines, each of them containing one sentence: either statement or question. The statement ends with a dot character (.), the question ends with a question mark (?). No statement will appear more than once, however the questions can be repeated. There is one extra line after each dialogue. That line ends with an exclamation mark (!).

Sentences can contain words, spaces and punctuation characters (such as commas, colons, semicolons etc.). All words contain only letters of English alphabet and are case-sensitive. That means the same word is always written the same way, usually in lowercase. Acronyms, names and some other words can begin with capital letters. For simplicity, all sentences begin with a lowercase letter. Only if the first word should be written with a capital, the sentence begins with a capital letter. There are no unneeded spaces between words. No line will have more than 100 characters. There will be at most 100 statements per each test case.

Statements

Each statement has one of the following two forms (_ denotes a space):
subject _*predicate*[s] [_*object*] .
subject _don't|doesn't _*predicate* [_*object*] .

The square brackets mark an optional part, the vertical line two possible variants. Subject is

a single word, noun or pronoun in singular. Predicate is a verb (single word) denoting some activity. Object can be any text. Object does not contain any dots. Any pair "verb + object" determines unique activity. The same verb with different objects makes different independent activities, i.e. the different and independent meaning of the sentence. Sentence without any object can be considered as sentence with an empty object. The verb without an object has different and independent meaning than the same verb with any non-empty object.

The first variant of sentence denotes a positive statement. The word "*predicate*[s]" means verb that matches the subject of the sentence. If the subject is "I" or "you", the verb has the same form as the infinitive. With any other subject, the letter "s" is appended on the end of the verb. Assume there are no irregular verbs.

The second variant is a negative statement. Verb "don't" or "doesn't" must also match the subject. The form "don't" is used with either "I" or "you", "doesn't" is used in any other case.

A special generic subject "everybody" can be used. It means the activity holds for any subject. Other special subject is "nobody". Such sentence also holds for any subject, but its meaning is negative. Both of these generic subjects can be used with the first variant only (without "doesn't"). The sentence "nobody likes something" is exactly equal to "everybody doesn't like something", except the latter form will never occur in the input.

Questions

Each question has one of the following three forms:

1. do|does _*subject* _*predicate* [_object] ?
2. who _*predicate*s [_*object*] ?
3. what _do|does _*subject* do ?

The word "do|does" always matches the subject ("do I?", "do you?", "does any other subject?"). In the second type of question, predicate always matches the word "who", i.e. the "s" is always appended. Generic subjects cannot be used in questions.

Output

For each dialogue, your program must output the line Dialogue #*D*:, where *D* is the sequence number of dialogue, starting with 1. Then print exactly three lines for every question: the first line repeats the question, the second line contains the answer, and the third line is empty. Print nothing for statements. After each dialogue, print the same line with an exclamation mark that was in the input. Then print one extra empty line. Empty line contains a new-line character only.

The answer must be properly formated to be accepted by a TTS module. Only the statements appearing in the input before the answer are used for the corresponding reply. If there is any contradiction among statements, the reply is always I am abroad.. If the question and statements consider the special subject "you", it must be replaced with "I" in the answer. If the question considers special subject "I", it must be replaced with "you" in the answer. The verb must always match the subject of the sentence. The exact form of the correct answer depends on the type of

question.

1. does subject predicate [object] ?

If there is any positive statement about the mentioned subject (or generic subject "everybody"), predicate and object, the answer is:
yes, *_subject _predicate*[s] [*_object*] .

If there is any negative statement about the mentioned subject (or generic subject "nobody"), predicate and object, the answer is:
no, *_subject* _don't|doesn't *_predicate* [*_object*] .

Otherwise, the answer is: maybe.

Subject in the answer is always the same subject as the subject of the question.

2. who predicates [object] ?

If there is a positive statement considering any subject, the specified predicate and object, the answer is:
subject _predicate[s] [*_object*] .

If two or more subjects match the activity, replace the subject in the answer with enumeration of all such subjects, in the same order as the corresponding statements have appeared in the input. Subjects are separated with comma and space, last two subjects are separated with the word "and". If "everybody" belongs to the group of enumerated subjects, do not enumerate subjects, and print "everybody" only. If the enumeration contains at least two subjects, the predicate matches the plural subject (i.e. verb is without trailing "s"), otherwise it matches the only subject.
subject1 , *_subject2* _and *_subject3 predicate* [*_object*] .

If there is a negative statement considering the generic subject "nobody", the specified predicate and object, the answer is:
*nobody _predicate*s [*_object*] .

Otherwise, the answer is: I don't know.

3. what does subject do ?

If there are one or more sentences (both positive and negative) considering the specified subject (or a generic subject "everybody" or "nobody"), all verbs and objects from such sentences must be included in a reply in the same order as the corresponding sentences have appeared in the input. No verb-object pair can be included more than once (the eventual second appearance must be skipped). The verb-object pairs are separated by a comma followed by a space, the last verb is separated by a comma and the word "and". Please note the comma is printed here although there was no comma when separating the subjects in the previous type of answer (see above). The negative answers have the same form as the statements, that means the verb "don't" or "doesn't" is used:
subject [_don't|doesn't] *_predicate1*[s] [*_object1*] ,
[_don't|doesn't] *_predicate2*[s] [*_object2*] ,
_and [_don't|doesn't] *_predicate3*[s] [*_object3*] .

subject [_don't|doesn't] _*predicate1*[s] [_*object1*] ,
_and [_don't|doesn't] _*predicate2*[s] [_*object2*] .
subject [_don't|doesn't] _*predicate*[s] [_*objectf*] .

Otherwise, the answer is: I don't know.

Example

Sample Input:

```
1
I like hotdogs.
nobody likes to work.
everybody smiles.
what do I do?
who smiles?
what do you do?
does Joe smile?
do I like to work?
everybody hurts sometimes.
who walks there?
Michal walks there.
who walks there?
what does Michal do?
do you understand?
nobody walks there.
do you understand now?
bye!
```

Sample Output:

```
Dialogue #1:
what do I do?
you like hotdogs, don't like to work, and smile.

who smiles?
everybody smiles.

what do you do?
I don't like to work, and smile.

does Joe smile?
yes, Joe smiles.

do I like to work?
no, you don't like to work.

who walks there?
I don't know.

who walks there?
Michal walks there.

what does Michal do?
Michal doesn't like to work, smiles, hurts sometimes, and walks there.

do you understand?
maybe.
```

do you understand now?
I am abroad.

bye!