code jam
print "hello, world!"

Practice Mode

Contest scoreboard | Sign in

EuroPython 2011

A. Centauri Prime

B. Music Collection

C. Irregular Expressions

D. Twibet

Questions asked

Submissions

Centauri Prime

7pt	Not attempted **41/42 users** correct (98%)
8pt	Not attempted **10/38 users** correct (26%)

Music Collection

8pt	Not attempted **16/27 users** correct (59%)
12pt	Not attempted **15/16 users** correct (94%)

Irregular Expressions

10pt	Not attempted **14/16 users** correct (88%)
15pt	Not attempted **8/14 users** correct (57%)

Twibet

15pt	Not attempted **16/16 users** correct (100%)
25pt	Not attempted **13/16 users** correct (81%)

Top Scores

charango	100
JPerla	100
valentin	92
RadomirDopieralski	77
vad	77
fox91	77
fon	75
mstepniowski	75
alexamici	72
davider	67

Problem A. Centauri Prime

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the Quick-Start Guide to get started.

Small input 1 7 points	Solve A-small-1
Small input 2 8 points	Solve A-small-2

Problem

Back in the old days before the creation of the mighty Centauri Republic, the planet Centauri Prime was split into several independent kingdoms. The kingdom of Mollaristan was ruled by king Loatold, while the kingdom of Auritania was under the rule of queen Elana. In fact, it just so happened that every kingdom whose name ended in a consonant was ruled by a king, while every kingdom whose name ended in a vowel was ruled by a queen. Also because of an amazing coincidence, all kingdoms whose named ended in the letter 'y' were constantly in a state of turmoil and were not ruled by anyone. Can you write a program that will determine the current rulers of several countries, given the countries' names?

Input

The first line of the input gives the number of test cases, **T**. **T** lines follow, each one containing the name of one country. Country names will consist of only lower case English letters, starting with a capital letter. There will be no other characters on any line, and no empty lines.

Output

For each test case, output one line containing "Case #**x**: **C** is ruled by **Y.**", where **x** is the case number (starting from 1), **C** is the country name, and **Y** is either "a king", "a queen" or "nobody".

Be careful with capitalization and the terminating period. Your output must be in exactly this format. See the examples below.

Limits

1 ≤ **T** ≤ 300.

Small dataset

Each country name will have between 3 and 20 letters.

Large dataset

Each country name will have at most 100 letters.

Sample

```
Input          Output

3              Case #1: Mollaristan is ruled by a king.
Mollaristan    Case #2: Auritania is ruled by a queen.
Auritania      Case #3: Zizily is ruled by nobody.
Zizily
```

EuroPython 2011

A. Centauri Prime

B. Music Collection

C. Irregular Expressions

D. Twibet

Questions asked

Submissions

Centauri Prime

7pt Not attempted
41/42 users correct (98%)

8pt Not attempted
10/38 users correct (26%)

Music Collection

8pt Not attempted
16/27 users correct (59%)

12pt Not attempted
15/16 users correct (94%)

Irregular Expressions

10pt Not attempted
14/16 users correct (88%)

15pt Not attempted
8/14 users correct (57%)

Twibet

15pt Not attempted
16/16 users correct (100%)

25pt Not attempted
13/16 users correct (81%)

Top Scores

charango	100
JPerla	100
valentin	92
RadomirDopieralski	77
vad	77
fox91	77
fon	75
mstepniowski	75
alexamici	72
davider	67

Problem B. **Music Collection**

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the Quick-Start Guide to get started.

Small input 1
8 points
Solve B-small-1

Small input 2
12 points
Solve B-small-2

Problem

Audio Phil has a huge music collection, and he is very particular about the songs he listens to. Each song has a name that is a string of characters. His music player has a search feature that lets Phil type a substring into the search box, and the player then lists all songs whose names contain the substring. If there is exactly one song that matches the search, then Phil can hit the Enter key to play that song.

Phil hates using the mouse, and he doesn't like typing too much, so he insists on always typing the shortest possible substring that will match exactly the one song that he wants to play at this moment. Could you help him find his optimal search query?

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. Each one starts with a line containing a single number **N**. The next **N** lines each contain one song name -- these are all of the songs in Phil's collection.

Each song name will consist of only letters, spaces and the hyphen character (-). All songs in Phil's collection will be unique and at most 100 characters in length. Song names are case insensitive, so "dZihan" is the same is "Dzihan". The search algorithm is also case insensitive.

Output

For each test case, output one line containing "Case #**x**:", where **x** is the case number (starting from 1). After that, print **N** lines, one for each song in Phil's collection, in the order that the songs were given in the input. For each song, print the shortest string of characters that will uniquely find that song. If there are several correct answers, print the lexicographically smallest one. Put double quotes around each string. If there is no correct answer, print ":(" without the double quotes.

Note that upper case letters come lexicographically before lower case letters, hyphen comes before all letters, and space comes before hyphen.

Limits

1 ≤ **T** ≤ 100.

Small dataset

1 ≤ **N** ≤ 10.

Large dataset

1 ≤ **N** ≤ 100.

Sample

```
Input                                 Output

5
6                                     Case #1:
A Perfect Circle - Gravity            "V"
Aimee Mann - You Do                   " D"
Aqualung - Cinderella                 "Q"
Arcade Fire - Haiti                   " F"
Art of Noise - Pleure                 "S"
ATB - Marrakech                       "B"
2                                     Case #2:
Hybrid - Altitude                     "A"
Kings of Convenience - The Build-up   "C"
3                                     Case #3:
aaaaaaaabb                            "AAAAAAAA"
aaaaaaabbb                            "BBB"
ababababab                            "BA"
3                                     Case #4:
butter                                :(
fly                                   :(
butterfly                             "RF"
1                                     Case #5:
Unknown Artist - Track One            ""
```

EuroPython 2011

A. Centauri Prime
B. Music Collection
C. Irregular Expressions
D. Twibet

Questions asked

Submissions

Centauri Prime

7pt	Not attempted **41/42 users** correct (98%)
8pt	Not attempted **10/38 users** correct (26%)

Music Collection

8pt	Not attempted **16/27 users** correct (59%)
12pt	Not attempted **15/16 users** correct (94%)

Irregular Expressions

10pt	Not attempted **14/16 users** correct (88%)
15pt	Not attempted **8/14 users** correct (57%)

Twibet

15pt	Not attempted **16/16 users** correct (100%)
25pt	Not attempted **13/16 users** correct (81%)

Top Scores

charango	100
JPerla	100
valentin	92
RadomirDopieralski	77
vad	77
fox91	77
fon	75
mstepniowski	75
alexamici	72
davider	67

Problem C. **Irregular Expressions**

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the Quick-Start Guide to get started.

Small input 10 points	Solve C-small
Large input 15 points	Solve C-large

Problem

You are competing at the 2011 World Witch and Warlock Tournament. In this round of the competition, the Head Witch challenges all contestants to test their spell deflection skills. Contestants stand in a circle, and the Head Witch stands in the center and starts casting spells. The contestants then have to cast counter-spells as quickly as possible. Any contestant who is too slow risks being turned into a toad or petrified.

Each spell that the Head Witch casts consists of three words -- the start, the middle and the end. Each word consists of one or more syllables. The start word is always the same as the end word and consists of at least two syllables. A syllable consists of any number of letters, including exactly one vowel. There are 5 vowels: 'a', 'e', 'i', 'o' and 'u'. All other letters are considered to be consonants, including the letter 'y'.

Examples of valid syllables are "ab", "ra", "cad", "o" and "shabbr". Strings like "ero" and "grrgh" are not valid syllables.

To make things more difficult, the Head Witch speaks very fast, so you cannot easily figure out where one word ends and another one begins. What's worse, she sometimes says some useless gibberish before and after the spell in order to confuse the contestants, or she may not even cast any spell at all and say something completely unrelated instead.

For example, she may say "abracadabra", which is a valid spell because it consists of the words "abra", "cad" and "abra", with the start word and the end word being the same. The word "abra" consists of two syllables -- "ab" and "ra". The word "cad" consists of one syllable -- "cad". (Alternatively, "abra" could also be interpreted as "a-bra" or "abr-a".)

The Head Witch might also say "kajabbamajabbajab", which contains the spell "jabba ma jabba". Or she might say "frufrumfuffle", which is gibberish and does not contain any spells.

For each expression that the Head Witch says, you want to determine quickly whether the expression contains a spell or not. Your molecular integrity depends on it! Fortunately, you have managed to conjure up a computer. Now all you need to do is determine which of the Witch's expressions contain spells.

Input

The first line of the input gives the number of test cases, **T**. **T** lines follow. Each one contains an expression, consisting of one or more lower case English letters and no spaces.

Output

For each test case, output one line containing "Case #x: y", where x is the case number (starting from 1) and y is either "Spell!" or "Nothing." (be careful with spelling and punctuation).

Limits

1 ≤ **T** ≤ 100.

Small dataset

At most 20 characters in each expression.

Large dataset

At most 100 characters in each expression.

Sample

```
Input                  Output

4                      Case #1: Spell!
abracadabra            Case #2: Spell!

kajabbamajabbajab      Case #3: Nothing.
frufrumfuffle          Case #4: Nothing.
```

```
schprexityschprex
```

EuroPython 2011

Questions asked

Submissions

Centauri Prime

7pt	Not attempted **41/42 users** correct (98%)
8pt	Not attempted **10/38 users** correct (26%)

Music Collection

8pt	Not attempted **16/27 users** correct (59%)
12pt	Not attempted **15/16 users** correct (94%)

Irregular Expressions

10pt	Not attempted **14/16 users** correct (88%)
15pt	Not attempted **8/14 users** correct (57%)

Twibet

15pt	Not attempted **16/16 users** correct (100%)
25pt	Not attempted **13/16 users** correct (81%)

Top Scores

charango	100
JPerla	100
valentin	92
RadomirDopieralski	77
vad	77
fox91	77
fon	75
mstepniowski	75
alexamici	72
davider	67

Problem D. Twibet

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the Quick-Start Guide to get started.

Small input 15 points	Solve D-small
Large input 25 points	Solve D-large

Problem

The holy country of Twibet has **N** monks. Each monk has a unique number, from 1 to **N**. They do not use names for religious reasons. The monks are constantly on the move, slowly walking around Twibet. Each monk follows exactly one other monk.

Most of the time, every monk is silent, but on day K, monk number K stops, turns around and whispers the 140 Words of Wisdom. The whisper is quiet, so only the monk's immediate followers can hear it. At that point, each of his followers stops, turns around and whispers the same words to each of his own followers. This chain continues -- each follower who has just heard the Words, but has not yet whispered the Words today, stops and whispers to his followers.

After all of the monks who could have heard the words have whispered them, they all turn back around and continue walking as usual... until the next day, when this all starts again, but this time starting with a different monk.

How many monks will whisper the 140 Words of Wisdom on day K, for each K between 1 and **N**?

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. Each one starts with a line containing a single integer **N**. The next line contains **N** space-separated integers $\mathbf{F_1}$, $\mathbf{F_2}$, ..., $\mathbf{F_N}$. Monk 1 follows monk $\mathbf{F_1}$. Monk 2 follows monk $\mathbf{F_2}$, etc.

Output

For each test case, output one line containing "Case #x:", where x is the case number (starting from 1). Then output **N** lines, one for each day. The first line should contain the number of monks who will whisper the Words on day 1. The next line -- on day 2, etc.

Limits

1 ≤ **T** ≤ 100.
No monk will immediately follow himself ($\mathbf{F_k}$ is never equal to **k**).

Small dataset

2 ≤ **N** ≤ 10.

Large dataset

2 ≤ **N** ≤ 1000.

Sample

```
Input        Output

2            Case #1:
3            3
2 3 1        3
4            3
2 3 2 1      Case #2:
             2
             4
             4
             1
```

Explanation

In Case #1, all 3 monks are walking around in a circle. Whenever one of them whispers the Words, his follower whispers next, and the the remaining monk whispers after that. On each of the 3 days, all 3 monks will eventually whisper the Words.

In Case #2, 1 follows 2, 2 follows 3, 3 follows 2, and 4 follows 1. On day 1, when monk 1 whispers first, monk 4 hears and whispers next; monks 2 and 3 will not hear the words that day. On day 2, monk 2 whispers first; monks 1 and 3 hear and whisper next; finally, monk 4 hears monk 1 and whispers last. On day 3, monks whisper in the order 3, 2, 1, 4. On day 4, monk 4 whispers the Words and no one hears him.