Question 1: Squares

Input File: squares.in
Output File: squares.out

A few days earlier a certain mathematician had been fired from his job so he has made up his mind to take revenge on his former employers and has changed all the numbers in their databases to their corresponding forms in different numerical systems using different bases. At the beginning it seemed to everyone to be just a stupid joke and hopefully they would soon find the correct data hidden somewhere. They were wrong, because even the backup database copies have been changed. The only hint, they were given was that all the data had been transformed to systems with such a base that it is the smallest base in which input numbers are squares.

Your task is to find these bases. You need to hurry up, because the whole firm's activity depends on your database fix. You may however assume that:

- for each number, there exists a sought base and it is less than 100
- all the digits in input numbers are characters 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- each number written in decimal system is smaller that 1000000000

Input

Data set consists of lines containing single numbers. Occurrence of 0 means the end of data set (0 is not treated as valid data).

Output

For each number you should find a smallest base of a numerical system in which this number is a square of some other number. Each number should be outputted in separate line.

Sample Input

```
61
1100
509
510
1013
0
```

Sample Output

```
8
3
12
16
6
```

Question 2: Guessing Game

Input File: guess.in
Output File: guess.out

Stan and Ollie are playing a guessing game. Stan thinks of a number between 1 and 10 and Ollie guesses what the number might be. After each guess, Stan indicates whether Ollie's guess is too high, too low, or right on.

After playing several rounds, Ollie has become suspicious that Stan cheats; that is, that he changes the number between Ollie's guesses. To prepare his case against Stan, Ollie has recorded a transcript of several games. You are to determine whether or not each transcript proves that Stan is cheating.

Standard input consists of several transcripts. Each transcript consists of a number of paired guesses and responses. A guess is a line containing single integer between 1 and 10, and a response is a line containing "too high", "too low", or "right on". Each game ends with "right on". A line containing 0 follows the last transcript.

For each game, output a line "Stan is dishonest" if Stan's responses are inconsistent with the final guess and response. Otherwise, print "Stan may be honest".

Sample Input

```
10
too high
3
too low
4
too high
2
right on
5
too low
7
too high
6
right on
0
```

Output for Sample Input

```
Stan is dishonest
Stan may be honest
```

Question 3: Passwords

Input File: password.in
Output File: password.out

Having several accounts on several servers one has to remember many passwords. You can imagine a situation when someone forgets one of them. He/she remembers only that it consisted of words x, y and z as well as two digits: one at the very beginning and the other one at the end of the password.

Your task is to write a program which will generate all possible password on the basis of given dictionary and set of rules. For the example given above the dictionary contains three words: x, y, z, and the rule is given as 0#0 what stands for <digit><word_from_the_dictionary><digit>.

Input

First line contains a number of words in the dictionary (n). The words themselves are given in n consecutive lines. The next line contains number of rules (m). Similarly consecutive m lines contain rules. Each rule consists of characters `#' and `0' given in arbitrary order. The character `#' stands for word from the dictionary while the character "0" stands for a digit. Input data may contain many sets of dictionaries with rules attached two them.

Output

For each set `dictionary + rules' you should output two hyphens followed by a linebreak and all matching passwords given in consecutive lines. Passwords should be sorted by rules what means that first all passwords matching the first rule and all words must be given, followed by passwords matching the second rule and all words, etc. Within set of passwords matching a word and a rule an ascending digit order must be preserved.

Assumptions: A number of words in the dictionary is greater than 0 and smaller or equal to 100 ($0<n\leq100$). Length of the word is greater than 0 and smaller than 256. A word may contain characters `A'..`Z',`a'..`z',`0'..`9'.A number of rules is smaller that 1000, and a rule is shorter that 256 characters. A character `0' may occur in the rule no more than 9 times, but it has to occur at least once. The character `#' is not mandatory meaning there can be so such characters in the rule.

Sample Input

```
2
root
2super
1
#0
1
admin
1
#0#
```

Sample Output

```
--
root0
root1
root2
root3
root4
root5
root6
root7
root8
root9
2super0
2super1
2super2
2super3
2super4
2super5
2super6
2super7
2super8
2super9
--
admin0admin
admin1admin
admin2admin
admin3admin
admin4admin
admin5admin
admin6admin
admin7admin
admin8admin
admin9admin
```

Question 4: Just The Facts

Input File: facts.in
Output File: facts.out

The expression N!, read as ``N factorial,'' denotes the product of the first N positive integers, where N is nonnegative. So, for example,

N	N!
0	1
1	1
2	2
3	6
4	24
5	120
10	3628800

For this problem, you are to write a program that can compute the last non-zero digit of any factorial for ($0 \leq N \leq 10000$). For example, if your program is asked to compute the last nonzero digit of 5!, your program should produce ``2'' because 5! = 120, and 2 is the last nonzero digit of 120.

Input

Input to the program is a series of nonnegative integers not exceeding 10000, each on its own line with no other letters, digits or spaces. For each integer N, you should read the value and compute the last nonzero digit of N!.

Output

For each integer input, the program should print exactly one line of output. Each line of output should contain the value N, right-justified in columns 1 through 5 with leading blanks, not leading zeroes. Columns 6 - 9 must contain “->” (space hyphen greater space). Column 10 must contain the single last non-zero digit of N!.

Sample Input

```
1
2
26
125
3125
9999
```

Sample Output

```
    1 -> 1
    2 -> 2
   26 -> 4
  125 -> 8
 3125 -> 2
 9999 -> 8
```

Question 5: Parenthesis Balance

Input File: balance.in
Output File: balance.out

You are given a string consisting of parentheses () and []. A string of this type is said to be correct:

a) if it is the empty string
b) if A and B are correct, AB is correct,
c) if A is correct, (A) and [A] is correct.

Write a program that takes a sequence of strings of this type and check their correctness. Your program can assume that the maximum string length is 128.

Input

The file contains a positive integer n and a sequence of n strings of parentheses () and [], one string a line.

Output

A sequence of Yes or No on the output file.

Sample Input

```
3
([])
(([()]))
([()[]()])()
```

Sample Output

```
Yes
No
Yes
```

Question 6: The Knights Of The Round Table

Input File: knights.in
Output File: knights.out

The Problem

King Arthur is planning to build the round table in a new room, but this time he wants a room that have sunlight entering it, so he planned to build a glass roof. He also wishes his round table to shine during the day, specially at noon, so he wants it to be covered totally by the sunlight. But Lancelot wants the glass part of the room roof to be triangular (and nobody knows the reason why, maybe he made a vow or something like that). So, there will be a triangular area in the room which will be all covered by the sunlight at noon and the round table must be build in this area.

Now, King Arthur wants to build the biggest table that he cans such that it fits in the triangular sunlighted area. As he is not very good in geometry, he asked Galahad to help him (Lancelot is very good in geometry, but King Arthur didn't asked Lancelot to help him because he feared that he would come up with another strange suggestion).

Can you help Galahad (since he's not too good with computers) and write a program which gives the radius of the biggest round table that fits in the sunlighted area? You can assume that the round table is a perfect circle.

The Input

There'll be an arbitrary number of rooms. Each room is represented by three real numbers (a, b and c), which stand for the sizes of the triangular sunlighted area. No triangle size will be greater than 1000000 and you may assume that max(a,b,c) <= (a + b + c) / 2. You must read until you reach the end of the file.

The Output

For each room configuration read, you must print the following line:
The radius of the round table is: r
Where r is the radius of the biggest round table that fits in the sunlighted area, rounded to 3 decimal digits.

Sample Input

12.0 12.0 8.0

Sample Output

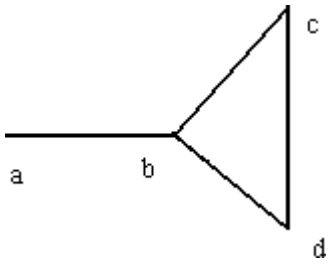
The radius of the round table is: 2.828

Question 7: The Forrest for the Trees

Input File: trees.in
Output File: trees.out

A graph G is a set of point V(G), together with a set of edges E(G), where each element of E(G) is an unordered pair of distinct points of V(G).

Example 1: Let G be a graph where V(G) = {a,b,c,d} and E(G) = {(a,b),(b,c),(c,d), (d,b)}. The figure gives a depiction of G.

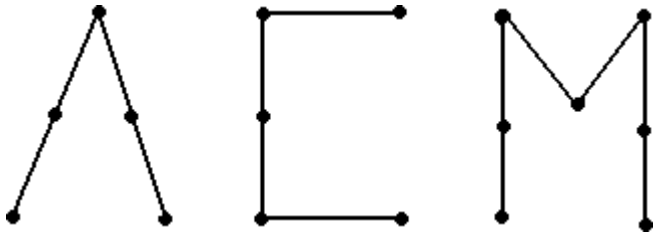


Notice that G contains the ``cycle" , {(b,c),(c,d),(d,b)}. A graph devoid of cycles is called a tree. A path in a graph G is an alternating sequence of points and edges, (beginning and ending with a point) such that all the points of the path are distinct. In the graph of example 1, {a, (a,b), b, (b,c), c, (c,d), d} is a path.

Fact: Every two points of a tree are joined by a unique path.

A graph is called connected if every pair of points are joined by a path. The graph of example 1 is connected. If a graph is not connected then it is made up of ``subgraphs" which are. Each one of these subgraphs is called a connected component of the graph G.

A graph for which each connected component is a tree is called a forest, see figure below.



One extreme case worth mentioning is the case when one of the component trees has one point but no edges joined to it. This tree likes like an isolated dot. We will call this an acorn. We are ready to define the problem.

Problem: Given a forest you are to write a program that counts the number of trees and acorns.

Input

The first line of the input file contains the number of test cases your program has to process. Each test case is a forest description consisting of two parts:

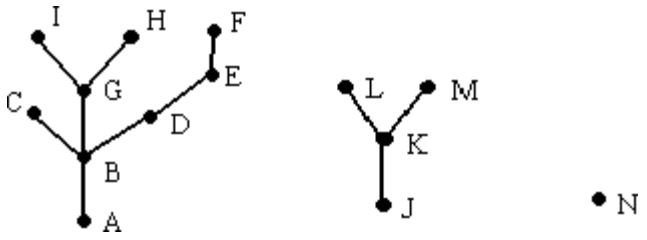
1. A list of edges of the tree (one per line, given as an unordered pair of capital letters delimited by a row of asterisks).
2. A list of points of the tree (these will be given on one line with a maximum of 26 corresponding to the capital letters, A - Z).

Output

For each test case your program should print the number of trees and the number of acorns, in a sentence, for example:

``There are x tree(s) and y acorn(s).", where x and y are the numbers of trees and acorns, respectively.

Example 2: Let G be a graph whose edges and points are given by the first test case in the sample input. A depiction of this graph is given in figure following.



Notes: A forest may have no trees and all acorns, all trees and no acorns, or anything inbetween, so keep your eyes open and don't miss the forest for the trees!

Sample Input

```
2
(A,B)
(B,C)
(B,D)
(D,E)
(E,F)
(B,G)
(G,H)
(G,I)
(J,K)
(K,L)
(K,M)
****
A,B,C,D,E,F,G,H,I,J,K,L,M,N
(A,B)
(A,C)
(C,F)
**
A,B,C,D,F
```

Sample Output

```
There are 2 tree(s) and 1 acorn(s).
There are 1 tree(s) and 1 acorn(s).
```