692 BUT We Need a Diagram

Consider a data structure called BUT (Binary and/or Unary Tree). A BUT is defined inductively as follows:

- Let l be a letter of the English alphabet, either lowercase or uppercase (in the sequel, we say simply "a letter"). Then, the object that consists only of l, designating l as its label, is a BUT. In this case, it is called a 0-ary BUT.
- Let l be a letter and C a BUT. Then, the object that consists of l and C, designating l as its label and C as its component, is a BUT. In this case, it is called a unary BUT.
- Let l be a letter, L and R BUTs. Then, the object that consists of l, L and R, designating l as its label, L as its left component, and R as its right component, is a BUT. In this case, it is called a binary BUT.

A BUT can be represented by an expression in the following way.

- When a BUT B is 0-ary, its representation is the letter of its label.
- When a BUT B is unary, its representation is the letter of its label followed by the parenthesized representation of its component.
- When a BUT B is binary, its representation is the letter of its label, a left parenthesis, the representation of its left component, a comma, the representation of its right component, and a right parenthesis, arranged in this order.

Here are examples:

```
a
A(b)
a(a,B)
a(B(c(D),E),f(g(H,i)))
```

Such an expression is concise, but a diagram is much more appealing to our eyes. We prefer a diagram:

```
D  H i
-  ---
c E g
--- -
 B  f
 ----
  a
```

to the expression `a(B(c(D),E),f(g(H,i)))`.

Your mission is to write a program that converts the expression representing a BUT into its diagram. We want to keep a diagram as compact as possible assuming that we display it on a conventional character terminal with a fixed pitch font such as Courier. Let's define the diagram D for a BUT B inductively along the structure of B as follows:

- When B is 0-ary, D consists only of a letter of its label. The letter is called the root of D, and also called the leaf of D.
- When B is unary, D consists of a letter l of its label, a minus symbol S, and the diagram C for its component, satisfying the following constraints:
  - l is just below S.
  - The root of C is just above S.

  l is called the root of D, and the leaves of C are called the leaves of D.
- When B is binary, D consists of a letter l of its label, a sequence of minus symbols S, the diagram L for its left component, and the diagram R for its right component, satisfying the following constraints:
  - S is contiguous, and is in a line.
  - l is just below the central minus symbol of S, where, if the center of S locates on a minus symbol s, s is the central, and if the center of S locates between adjacent minus symbols, the left one of them is the central.
  - The root of L is just above the leftmost minus symbol of S, and the root of R is just above the rightmost minus symbol of S.
  - In any line of D, L and R do not touch or overlap each other.
  - No minus symbols are just above the leaves of L and R.

  l is called the root of D, and the leaves of L and R are called the leaves of D.

Input

The input to the program is a sequence of expressions representing BUTs. Each expression except the last one is terminated by a semicolon. The last expression is terminated by a period. White spaces (tabs and blanks) should be ignored. An expression may extend over multiple lines. The number of letters, i.e., the number of characters except parentheses, commas, and white spaces, in an expression is at most 80.

You may assume that the input is syntactically correct and need not take care of error cases.

Output

Each expression is to be identified with a number starting with 1 in the order of occurrence in the input. Output should be produced in the order of the input.

For each expression, a line consisting of the identification number of the expression followed by a colon should be produced first, and then, the diagram for the BUT represented by the expression should be produced.

For a diagram, output should consist of the minimum number of lines, which contain only letters or minus symbols together with minimum number of blanks required to obey the rules shown above.

Sample Input

```
a(A,b(B,C));
x( y( y( z(z), v( s, t ) ) ), u ) ;

a( b( c,
      d(
         e(f),
         g
       )
    ),
   h( i(
         j(
            k(k,k),
            l(l)
          ),
         m(m)
       )
    )
 );

a(B(C),d(e(f(g(h(i(j,k),l),m),n),o),p))
.
```

Sample Output

```
1:
 B C
 ---
A b
---
 a
2:
z s t
- ---
z  v
----
 y
 -
 y u
 ---
  x
3:
   k k l
   --- -
 f  k  l m
 -  ---- -
 e g j   m
 --- -----
c d    i
---    -
 b     h
 -------
    a
4:
j k
---
 i l
 ---
  h m
  ---
   g n
   ---
    f o
    ---
   C e p
   - ---
   B  d
   ----
    a
```