Boolean Logic

Propositions are logical formulas consisting of proposition symbols and connecting operators. They are recursively defined by the following rules:

1. All proposition symbols (in this problem, lower-case alphabetic characters, e.g., a and z) are propositions.
2. If P is a proposition, (!P) is a proposition, and P is a direct subformula of it.
3. If P and Q are propositions, (P&Q), (P|Q), (P-->Q), and (P<->Q) are propositions, and P and Q are direct subformulas of them.
4. Nothing else is a proposition.

The operations !, &, |, -->, and <-> denote logical negation, conjunction, disjunction, implication, and equivalence, respectively. A proposition P is a subformula of a proposition R if P=R or P is a direct subformula of a proposition Q and Q is a subformula of R.

Let P be a proposition and assign boolean values (i.e., 0 or 1) to all proposition symbols that occur in P. This induces a boolean value to all subformulas of P according to the standard semantics of the logical operators:

negation	conjunction	disjunction	implication	equivalence
!0=1	0&0=0	0|0=0	0-->0=1	0<->0=1
!1=0	0&1=0	0|1=1	0-->1=1	0<->1=0
	1&0=0	1|0=1	1-->0=0	1<->0=0
	1&1=1	1|1=1	1-->1=1	1<->1=1

This way, a value for P can be calculated. This value depends on the choice of the assignment of boolean values to the proposition symbols. If P contains n different proposition symbols, there are 2^n different assignments. To evaluate all possible assignments we may use truth tables.

A truth table contains one line per assignment (i.e., 2^n lines in total). Every line contains the values of all subformulas under the chosen assignment. The value of a subformula is aligned with the proposition symbol, if the subformula is a proposition symbol, and with the center of the operator otherwise.

Input Specification

The input contains several test cases, each on a separate line. Every test case denotes a proposition and may contain arbitrary amounts of spaces in between. The input file terminates immediately after the newline symbol following the last test case.

Output Specification

For each test case generate a truth table for the denoted proposition. Start the truth table by repeating the input line. Evaluate the proposition (and its subformulas) for all assignments to its variables, and output one line for each assignment. The line must have the same length as the corresponding input line and must consist only of spaces and the characters 0 and 1. Output an

empty line after each test case.

Let $s_1,...,s_n$ be the proposition symbols in the denoted proposition sorted in alphabetic order. Then, all assignments of 0 to s_1 must precede the assignments of 1 to s_1. Within each of these blocks of assignments, all assignments of 0 to s_2 must precede the assignments of 1 to s_2, and so on.

Sample Input

```
((b --> a) <-> ((! a) --> (! b)))
  ((y &  a)   -  ->(c |c))
```

Sample Output

```
((b --> a) <-> ((! a) --> (! b)))
  0  1  0   1    1 0   1   1 0
  1  0  0   1    1 0   0   0 1
  0  1  1   1    0 1   1   1 0
  1  1  1   1    0 1   1   0 1

  ((y &  a)   -  ->(c |c))
    0 0  0       1  0 00
    1 0  0       1  0 00
    0 0  0       1  1 11
    1 0  0       1  1 11
    0 0  1       1  0 00
    1 1  1       0  0 00
    0 0  1       1  1 11
    1 1  1       1  1 11
```