A seguinte linguagem descrita representa a Lógica Proposicional e trabalha com expressões proposicionais.

- Ela possui 4 comandos:

  1. comando de atribuição:
     ```
     variável := expressão
     ```

  2. comando de exibição:
     ```
     out expressão_1, expressão_2, ..., expressão_n
     ```

  3. comando de entrada de dados:
     ```
     in variável_1, variável_2, ..., variável_n
     ```

  4. comando condicional:
     ```
     if(expressão) {
        comandos_1
     }
     else {
        comandos_2
     }
     ```

- Os comandos de atribuição, exibição e de entrada de dados devem ser terminados por ponto-e-vírgula.

- Os nomes das variáveis são formadas por uma seqüência de letras minúsculas e/ou dígitos. Mas, sempre devem iniciar por letra.

- A linguagem é case-sensitive.

- As expressões são formadas por parênteses balanceados, valores lógicos, variáveis, operadores lógicos.
  - Os valores lógicos podem ser:
    - `1`
    - `0`
    - `"v"`
    - `"f"`
    - `"verdadeiro"`
    - `"falso"`
    - `"false"`
    - `"true"`

    (observe que em alguns casos exige-se aspas).
  - Os operadores podem ser: `∨, ∧, ', ->, <->`

Exemplos de programas aceitos:

```
in a;
out a';
```

```
in a, b,c;
out a ^ b, a v b, a -> c;
```

```
in a,b,c;
d := a ^ (b <-> c);
if(d) {
      out d ^ a;
}
else{
      out d ^ c;

}
```

```
a := "v";
b := "f";
c := a -> b;
out c,c ^ 1;
```

```
out "v" ^ "f" -> "true" <-> "false";
```

```
in a,b,c;
d := (b' v a) ^ ((b <-> c) v (c'' ^ a)')';
if(a ^ (b <-> c)) {
      e := 1;
      f := 0;
      out d ^ a;
}
else{
      if(d) {
            e := 0;
      }
      else{
            f := 1;
      }
      out d v a;
}
out e ^ f;
```