A propositional signature is a non-empty set of symbols called atoms. (In examples, we will assume that p, q, r are atoms.) The symbols

\[ \neg \land \lor \rightarrow \]

are called propositional connectives. Among them, \( \neg \) (negation) is a unary connective, and the symbols \( \land \) (conjunction), \( \lor \) (disjunction), and \( \rightarrow \) (implication) are binary.

Take a propositional signature \( \sigma \) that contains neither propositional connectives nor parentheses (, ). The alphabet of propositional logic consists of the atoms from \( \sigma \), the propositional connectives, and the parentheses. By a string we understand a finite string of symbols in this alphabet. We define when a string is a (propositional) formula recursively, as follows:

- every atom is a formula,
- if \( F \) is a formula then \( \neg F \) is a formula,
- for any binary connective \( \odot \), if \( F \) and \( G \) are formulas then \( (F \odot G) \) is a formula.

Properties of formulas can be often proved by structural induction. In such a proof, we check that all atoms have the property \( P \) that we would like to establish, and that this property is preserved when a new formula is formed using a unary or binary connective. More precisely, we show that

- every atom has property \( P \),
- if a formula \( F \) has property \( P \) then so does \( \neg F \),
- for any binary connective \( \odot \), if formulas \( F \) and \( G \) have property \( P \) then so does \( (F \odot G) \).

Then we can conclude that property \( P \) holds for all formulas.

**Problem 1.1** In any prefix of a formula, the number of left parentheses is greater than or equal to the number of right parentheses. (A prefix of a string \( a_1 \cdots a_n \) is any string of the form \( a_1 \cdots a_m \) where \( 0 \leq m \leq n \).)
Problem 1.2 Every prefix of a formula $F$

- is a string of negations (possibly empty), or
- has more left than right parentheses, or
- equals $F$.

Problem 1.3 No formula can be represented in the form $(F \odot G)$, where $F$ and $G$ are formulas and $\odot$ is a binary connective, in more than one way.

We will abbreviate formulas of the form $(F \odot G)$ by dropping the outermost parentheses in them. For any formulas $F_1, F_2, \ldots, F_n \ (n > 2)$,

$$F_1 \land F_2 \land \cdots \land F_n$$

will stand for

$$(\cdots(F_1 \land F_2) \land \cdots \land F_n).$$

The abbreviation $F_1 \lor F_2 \lor \cdots \lor F_n$ will be understood in a similar way. The expression $F \leftrightarrow G$ will be used as shorthand for

$$(F \to G) \land (G \to F).$$