(Gödel/Turing incompleteness) Prove that we cannot consistently and completely define a total, deterministic interpreter. An interpreter is a predicate $I$ that applies to texts; when applied to a text representing a binary expression, its result is equal to the represented expression. For example,

$$I \langle \forall s: \ast \text{char} \cdot #s \geq 0 \rangle = \forall s: \ast \text{char} \cdot #s \geq 0$$

§

Let $Q = \langle \neg I Q \rangle$. Now

$$I Q \quad \text{replace } Q \text{ with its equal}$$

$$= I \langle \neg I Q \rangle \quad \text{If } I \text{ is a complete interpreter as described in the question, then}$$

$$= \neg I Q \quad \text{If } I \text{ is a complete incomplete, we have an inconsistency. To save ourselves we can leave}$$

the interpreter incomplete. In particular,

$$I \langle \neg I Q \rangle = \neg I Q$$

must not be a theorem. If it is an antitheorem, then $I$ is not an interpreter. So leave it unclassified. Alternatively, we could let $I$ be partial so that $I Q = \text{null}$, or nondeterministic so that $I Q = \text{bin}$. Then $I Q = \neg I Q$ is a theorem, but we cannot use the Completion Rule to prove it is an antitheorem because $I Q$ is not elementary. So we do not have an inconsistency, but we also do not have a total, deterministic interpreter. As any programmer can see, applying $I$ to $\langle \neg I Q \rangle$ will cause an infinite execution, and produce no answer.

Although the question does not ask for this, here is how you define an interpreter. Start with

$$I \langle \top \rangle = \top$$
$$I \langle \bot \rangle = \bot$$

Now, for texts that represent negations, we want to say something like

$$I \langle (\neg; s) \rangle = \neg I s$$

It says: to apply $I$ to a text that starts with $\langle \neg \rangle$, just apply $I$ to the text after the $\neg$, and then negate the result. For texts that represent conjunctions, we want to say something like

$$I (s; \langle \&; t \rangle) = I s \land I t$$

And so on for all operators of the theory we are interpreting. The trouble is precedence. For example, the expression

$$\neg \top \land \bot$$

starts with $\neg$, but it’s not negating $\top \land \bot$. One solution is to insist that all expressions be fully parenthesized. Another solution is to use Polish prefix notation (see Subsection 3.2.2 on page 31.)