Aufgabe 8.1.  [Tree Regular Expressions: Practice] (10 points)
Specify tree regular expressions for the following languages:

1. Odd natural numbers, represented by 0/0 and Suc/1.
2. Trees over f/1, g/1, a/0, that contain an odd number of f-symbols
3. Trees over f/2, g/2, a/0, where each path contains an odd number of f-symbols.
4. The mutually recursive datatype t, which represents unranked trees where nodes are labeled by Booleans, including the empty tree:
   \[
   b ::= True \mid False \\
   t ::= Empty \mid Node of b \ast f \\
   f ::= Nil \mid Cons of t \ast f
   \]
   Note, the alphabet is True/0, False/0, Empty/0, Node/2, Nil/0, Cons/2
5. The datatype f from above.

Aufgabe 8.2.  [Tree Regular Expressions: Theory] (10 points)

1. Warmup: Describe an algorithm contains-empty(e), which checks for a word-regular expression e, whether \( \varepsilon \in \left[ e \right] \). No proof required.
2. Given a ranked alphabet \( \mathcal{F} \). Describe and prove correct an algorithm contains(c, e) that checks whether, for a tree-regular expression e and a constant \( c \in \mathcal{F}_0 \), we have \( c \in \left[ e \right] \). Hint: contains can be defined recursively over the structure of e.