Aufgabe 6.1. [Path Language] (10 points)
Consider an ranked alphabet $F$, and a set $\Sigma$ that contains the function symbols of $f$, without arity information. (E.g., if $F$ is $f/2, g/1, a/0$, then $\Sigma = \{f, g, a\}$).

We define the paths of a tree as the words of function symbols on paths from the root to a leaf, i.e., we define $\pi : T(F) \to \Sigma^*$ by:
\[
\pi(f(t_1, \ldots, t_n)) := \{f \cdot w \mid \exists i. w \in \pi(t_i)\}
\]
Here, $\cdot$ is concatenation of words.

We extend $\pi$ to sets of trees: $\pi(L) := \bigcup_{t \in L} \pi(t)$.

Now, consider a regular word language $W$ over the alphabet $\Sigma$.

1. Prove that the language $L := \{t \mid \pi(t) \subseteq W\}$ is a tree regular language. (Hint: Assume $W$ is described by a word automaton, and construct a tree automaton for $L$. Try to make clear the intuition behind your construction.)

2. Can $L$ always be described by a deterministic top-down tree automaton? (Hint: You may assume that $W$ is described by a deterministic word automaton.)
Aufgabe 6.2. [Towards Schema Languages] (10 points)
Consider the datatype (e.g. in Haskell or ML)

\[
\begin{align*}
\text{char} &= A | \ldots | Z | 0 | \ldots | 9 \\
('a', 'b') \text{ pair} &= \text{Pair of ('a * 'b)} \\
'a' \text{ list} &= \text{Nil | Cons of ('a * 'a list)} \\
\text{string} &= \text{char list} \\
\text{db} &= ((\text{string, string}) \text{ pair}) \text{ list}
\end{align*}
\]

Intuitively, the type \text{db} describes a database as a map from keys to values, encoded as a list of pairs of keys and values. Both, keys and values, are strings, i.e., lists of characters. For example, a database that maps student names to student ids:

\[
[(\text{MARTIN, 617523}), (\text{PHIL, 2175312}), (\text{MARIA, 1872235})]
\]

Instances of the type \text{db} can be interpreted as trees over the alphabet

\[
\mathcal{F} = A/0, \ldots, Z/0, 0/0, \ldots, 9/0, \text{Pair}/2, \text{Nil}/0, \text{Cons}/2
\]

Note that the types \text{string} and \text{db} are just aliases, and introduce no constructors themselves.

1. Specify a tree automaton (Hint: Top-Down might be easier) for a well-formed student-id database, that is, a database where keys contain only letters, and values contain only digits. Moreover, neither empty keys, nor empty values should be allowed.

2. Now consider those databases that contain no duplicate keys. Can they be described as a regular language? (Justify your answer.)