1 Practicing Functional Programming in SML

1.1 Record Types (3 Points)

Given the following type definition for a date:

\[
\text{type date} = \{ \text{day : int, month : int, year : int} \}
\]

write a function

\[
\text{fun age(birthday : date, today : date) : date * date -> int}
\]

which calculates the age of a person in years given the birthday and the current date.

1.2 Lists (3 Points)

Implement a function:

\[
\text{fun avg_length : string list -> real}
\]

which, for a given list of strings, calculates the average string length based on the map and foldl functions presented in class. The length of a single string can be determined using String.size.
1.3 Sum Types and Higher-order Functions (4 Points)

Given the following type definition for a ternary tree:

```plaintext
datatype 'a tree =
    Leaf of 'a
  | Node of 'a tree * 'a tree * 'a tree
```

implement functions

```plaintext
fun tree_map (f : 'a -> 'b) (t : 'a tree) : 'b tree

and

fun tree_foldl (f : 'a * 'a -> 'a) (n: 'a) (t : 'a tree) : 'a
```

similar to the map and foldl functions for lists that were presented in class.