1 Practicing Functional Programming in SML

1.1 Record Types (3 Points)

Given the following type definition for a date:

```sml
type date = {day : int, month : int, year : int}
```

write a function

```sml
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:

```sml
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:

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

implement functions

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

and

```ml
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.