Lab 4

Functional Programming (ITI0212)

2021.02.16

1. Write two functions

   ```
   beep : (Pair a b -> c) -> (a -> b -> c)
   ```

   ```
   boop : (a -> b -> c) -> (Pair a b -> c)
   ```

   such that for every function `f : Pair a b -> c` the equation

   ```
   boop (beep f) = f
   ```

   holds, and for every function `g : a -> b -> c` the equation

   ```
   beep (boop g) = g
   ```

   holds.

2. (a) Write functions

   ```
   conjunction : Bool -> Bool -> Bool
   ```

   ```
   disjunction : Bool -> Bool -> Bool
   ```

   that compute the logical conjunction and disjunction, respectively, of their inputs.

   (b) Using `foldList`, Write a function `conj : List Bool -> Bool` that returns the logical conjunction of the entire input list.

   (c) Using `foldList`, write a function `disj : List Bool -> Bool` that returns the logical disjunction of the entire input list.

   (d) Write the filter function for lists, `filterList : (a -> Bool) -> List a -> List a` in terms of `foldList`.

3. Recall the type of binary trees:

   ```
   data Tree : Type -> Type where
     Leaf : Tree a
     Node : Tree a -> a -> Tree a
   ```

(a) Write the fold function for binary trees.

(b) Use the fold function for binary trees to write the map function for binary trees, `mapTree : (a -> b) -> Tree a -> Tree b`.

(c) Use the fold function for binary trees to write a function `sumTree : Tree Nat -> Nat` that sums the data in the input tree.