Homework 10: Testing

CIS 194: Homework 10
Due Tuesday, November 8

The is just small, 10 points, finger exercise to reinforce the lecture material. Your main task is to work on the project.

Exercise 1

From your or the **example solution of week 7**, extract the `Tree` data type and the `labelTree` function. You can add `Eq` to the derived classes of `Tree`.

Declare an `Arbitrary` instance for trees:

```
instance Arbitrary a => Arbitrary (Tree a) where …
```

You do not have to implement a `shrink` function.

Use `sample` in GHCi to visually assess whether you generate useful looking trees.

Exercise 2

Implement these functions:

```
size :: Tree a -> Int
toList :: Tree a -> [a]
```

where `size` counts the number of leaves in the tree, and `toList` contains all the values in the leafs, from left to right.

Exercise 3

Create these QuickCheck properties:

- `prop_lengthToList :: Tree Integer -> Bool`

  The length of the list produced by `toList` is the size of the given tree.

- `prop_sizeLabelTree :: Tree Integer -> Bool`

  `labelTree` does not change the size of the tree.

- `prop_labelTree :: Tree Integer -> Bool`

  For every tree `t`, `toList (labelTree t)` is the expected list.

  Hint: `[0..n]` denotes the list of numbers from `0` to `n`, inclusively.

- `prop_labelTreeIdempotent :: Tree Integer -> Bool`

  Applying `labelTree` to a list twice does yield the same list as applying it once.