# PMT Additional Exercises (Week 4)

Nicholas Sim

February 1st, 2018

## 1 Tree Reversal

[2010 Q1b (C146)] Consider the following operations on trees and lists:

```haskell
data Tree a = Leaf a | Node (Tree a) (Tree a)

flatten :: Tree a -> [a]
flatten Leaf x = [x]
flatten (Node t1 t2) = (flatten t1) ++ (flatten t2)

rotate :: Tree a -> Tree a
rotate Leaf x = Leaf x
rotate (Node t1 t2) = Node (rotate t2) (rotate t1)

rev :: [a] -> [a]
rev [] = []
rev (x:xs) = (rev xs) ++ [x]
```

Prove, *using structural induction*, that:

$$\forall \texttt{t:Tree a.}\quad \texttt{flatten (rotate t) = rev (flatten t)}$$

In the proofs, state what is given, the induction hypothesis (if any), what is to be shown, and justify each step. You may use the property $(P)$, where:

```haskell
rev (xs ++ ys) = (rev ys) ++ (rev xs)
```

## 2 Solution to trees

A straightforward structural induction (to prove: as in question) on trees. Note that the type `a` does not matter in this question; consider it fixed.

**Base case** To show: `flatten (rotate (Leaf x)) = rev (flatten (Leaf x))`

| | |
|---|---|
| `flatten (rotate (Leaf x))` | |
| = `flatten (Leaf x)` | by def. rotate |
| = `[x]` | by def. flatten |
| = `rev [x]` | by def. rev |
| = `rev (flatten (Leaf x))` | by def. flatten |

**Inductive step** Take trees `t1`, `t2` arbitrarily. Inductive Hypothesis:

$$\forall \texttt{t1:Tree a.}\ \ \texttt{flatten (rotate t1) = rev (flatten t1)}$$
$$\wedge\forall \texttt{t2:Tree a.}\ \ \texttt{flatten (rotate t2) = rev (flatten t2)}$$

To show: `flatten (rotate (Node t1 t2)) = rev (flatten (Node t1 t2))`

| | |
|---|---|
| `flatten (rotate (Node t1 t2))` | |
| = `flatten (Node (rotate t2) (rotate t1))` | by def. rotate |
| = `(flatten (rotate t2)) ++ (flatten (rotate t1))` | by def. flatten |
| = `(rev (flatten t2)) ++ (rev (flatten t1))` | by I.H |
| = `rev (flatten t1) ++ (flatten t2)` | by (P) |
| = `rev (flatten (Node t1 t2))` | by def. flatten |