Lista 13

Zdefiniowany jest następujący typ dla drzew:

```
data Tree a = Leaf a | Branch (Tree a) (Tree a)
            deriving (Eq, Ord, Show, Read)
```

1. Zdefiniuj funkcję label :: Tree a -> Tree (a, Int), etykietującą liście drzewa kolejnymi liczbami naturalnymi począwszy od zera. Np. wartością wyrażenia

```
test = let t = Branch (Leaf  'a') (Leaf 'b')
       in label (Branch t t)
```

   jest drzewo:

```
Branch (Branch (Leaf ('a',0)) (Leaf ('b',1)))
       (Branch (Leaf ('a',2)) (Leaf ('b',3)))
```

2. Zdefiniuj funkcję mlabel :: Tree a -> Tree (a,Int) z taką samą funkcjonalnością jak label, ale wykorzystującą monadę stanu.

   Wskazówka. Zmieniającym się stanem jest wartość etykiety. Wykorzystaj monadę State z biblioteki standardowej (import Control.Monad.State).
   Kod funkcji mlabel nie będzie niestety krótszy niż kod funkcji label. Ponieważ stan jest „przeciągany" tylko dwukrotnie, praktycznie nie ma żadnej korzyści z jego ukrycia w monadzie stanu, ale jest to tylko proste zadanie dydaktyczne.