Haskell Exercises 9: Data Types

Antoni Diller

26 July 2011

(1) The type *Btree a* is defined as follows:

```
data Btree a = Tip a | Bin (Btree a) (Btree a)
               deriving (Eq,Ord,Show)
```

Define a function *mirror* which forms the mirror image of a tree. For example,

```
mirror (Bin (Tip 1) (Bin (Tip 6) (Tip 5)))
= Bin (Bin (Tip 5) (Tip 6)) (Tip 1)
```

(2) The type *Btree a* is defined as follows:

```
data Btree a = Tip a | Bin (Btree a) (Btree a)
               deriving (Eq,Ord,Show)
```

Define a function *leftmost* which takes an ordered tree, whose smallest element occurs at the extreme left, and returns a 2-tuple consisting of the leftmost element of the tree and a new version of the tree with the leftmost element removed.

(3) Using *leftmost* from (2), define the function *graft* which takes two ordered trees *lt* and *rt* in which every element of *lt* is smaller than every element of *rt* and joins them into a single tree so that the leftmost element of *rt* is replaced by the tree *lt*.

(4) Using *graft* from (3), define *remove* which takes a tree and an element and removes the leftmost occurrences of that element from the tree.