SoSe 2014

Prof. Dr. Margarita Esponda

ProInformatik II: Funktionale Programmierung

8. Übungsblatt (9. Tag)

1. Aufgabe

Nehmen wir an, wir müssen eine Reihe Berechnungen realisieren mit Zahlen, die folgende Form haben:

$a + b\sqrt{2}$, wobei *a* und *b* ganze Zahlen sind.

Würden wir zuerst die Wurzeln ausrechnen und dann die Berechnungen anstellen, hätten wir Rundungsfehler, die im Laufe der Berechnungen größer werden können. Die Rundungsfehler können minimiert werden, wenn wir zuerst nur die ganzzahligen Operationen realisieren und das Ausrechnen der Wurzeln ans Ende verschieben.

Beispiel:

$$\left(3 + 2\sqrt{2}\right) \cdot \left(2 + \sqrt{2}\right) = 10 + 7\sqrt{2}$$

Definieren Sie einen algebraischen Datentyp **SqRootNum**, der unsere Zahlen nur mit Hilfe der Koeffizienten **a** und **b** darstellt und definieren Sie die Additions-, Subtraktions- und Multiplikationsoperation für diesen Datentyp.

2. Aufgabe

Definieren Sie für folgende algebraische Datentyp-Definition (Baumstruktur ohne Inhalt aus der Vorlesung)

```
data SBTree = L | N SBTree SBTree

                deriving Show
```

folgende Funktionen:

```
insertLeaf :: SBTree -> SBTree                -- ein Blatt wird in den Baum eingefügt
insertLeafs :: SBTree -> Integer -> SBTree    -- eine eingegebene Anzahl von Blättern
                                                 wird in den Baum eingefügt

deleteLeaf :: SBTree -> SBTree                -- löscht ein Blatt aus dem Baum
deleteLeafs :: SBTree -> Integer -> SBTree    -- eine eingegebene Anzahl von Blättern
                                                 wird in den Baum gelöscht
```

Die **insertLeaf**- und **deleteLeaf**- Operationen sollen dafür sorgen, dass der Baum möglichst balanciert bleibt.

Definieren Sie eine Funktion **full**, die überprüft, ob eine **SBTree**-Baumstruktur ein vollständiger binärer Baum ist.

3. Aufgabe

a) Erweitern Sie die Funktionen für den algebraischen Datentyp **BSearchTree** (Binäre Suchbäume) um folgende Funktionen:

```
twoChildren :: (Ord a) => BSearchTree a -> Bool
    -- entscheidet, ob jeder Knoten des Baums genau zwei Kinder hat oder nicht
mapTree :: (Ord a, Ord b) => (a -> b) -> BSearchTree a -> BSearchTree b
foldTree :: (Ord a) => b -> (a -> b -> b -> b) -> BSearchTree a -> b
```

b) Redefinieren Sie mit Hilfe der **foldTree**-Funktion die **size**- und **height**-Funktionen, die für den **BSearchTree** Datentyp in der Vorlesung definiert worden sind.