Philipps Universität Marburg

# Übungen zur „Praktischen Informatik III“, WS 2008/09

Prof. Dr. R. Loogen · Fachbereich Mathematik und Informatik · Hans-Meerwein-Straße, D-35032 Marburg

**Nr. 4, Abgabe: 12. November 2008 vor der Vorlesung**

8. Suchbäume

Das auf der Vorlesungsseite zur Verfügung gestellte Modul `SearchTree.hs` exportiert die folgenden Operationen. Nur diese sollen in dieser Aufgabe verwendet werden. Pattern Matching ist für Suchbäume damit nicht möglich. Importieren Sie das Modul durch die Angabe von `import SearchTree` am Anfang Ihres Programms. 6 Punkte

```
module SearchTree
  (STree,
   nil, node,          -- :: STree a,  :: a -> STree a -> STree a -> STree a
   isNil,   isNode,    -- :: STree a -> Bool
   leftSub, rightSub, -- :: STree a -> STree a
   rootVal,            -- :: STree a -> a
   insTree, delTree    -- :: Ord a => a -> STree a -> STree a
   minTree,            -- :: Ord a => STree a -> a
   printTree           -- :: STree a -> IO ()          )    where ...
```

Ein binärer Baum heißt *balanciert*, wenn sich die Anzahl der Knoten im linken und rechten Teilbaum um höchstens eins unterscheiden und wenn beide Teilbäume balanciert sind. Der leere Baum ist per definitionem balanciert.

(a) Definieren Sie eine Funktion `size :: STree a -> Int` zur Bestimmung der Anzahl der Knoten in einem Suchbaum. / 1

(b) Schreiben Sie eine Funktion `isBalanced :: STree a -> Bool`, die testet, ob ein gegebenener Baum balanciert ist. / 2

(c) Entwickeln Sie eine Funktion `balance :: Ord a => STree a -> STree a`, die einen beliebigen Suchbaum in einen balancierten Suchbaum mit denselben Einträgen umwandelt. Testen Sie die Korrektheit Ihrer Funktionsdefinition mit QuickCheck. / 3

9. Listeninduktion, QuickCheck 6 Punkte

(a) Schreiben Sie rekursive Funktionen `delete :: Eq a => a -> [a] -> [a]` und `count :: Eq a => a -> [a] -> Int`. / 1

`delete` eliminiert alle Vorkommen eines Elements in einer Liste und `count` bestimmt, wie oft ein Element in einer Liste vorkommt.

Beispiel: `delete 'a' "Praktische Informatik"` $\Rightarrow^*$ `Prktische Informtik"` und `count 'a' "Praktische Informatik"` $\Rightarrow^*$ 2

(b) Testen Sie mit QuickCheck, ob die von Ihnen definierten Funktionen die folgende Gleichung erfüllen: `length (delete c cs) = length cs - count c cs`. / 2
Lassen Sie dabei bestimmen, in wievielen Testfällen die leere Liste, eine einelementige Liste bzw. eine Liste der Länge $\geq 5$ gewählt wurde. Als Typ der Testfunktion sollten Sie `Int -> [Int] -> Property` festlegen.

(c) Beweisen Sie mittels Listeninduktion die Gültigkeit der Gleichung aus (b) für beliebige endliche Listen `cs::[a]` und für beliebige Listenelemente `c::a`. / 3