

## Übungsblatt 14 - Abgabe 07.02.06

### Aufgabe 1 (Ausgeglichene Suchbäume)

a) Konstruieren Sie zu gegebener Folge einen AVL-Baum, indem Sie die Schlüssel in der angegebenen Reihenfolge in den leeren Baum einfügen.

9, 4, 13, 14, 17, 15, 16, 6, 5, 7

Benutzen Sie dabei das in der Vorlesung angegebene Verfahren. Zeichnen Sie den Baum nach jeder Einfügeoperation, geben Sie die notwendige Rotation an und veranschaulichen Sie die Rotation in dem Baum durch Kennzeichnung der relevanten Bestandteile. Annotieren Sie in jedem Knoten den Balancefaktor.

b) Folgende konkrete Repräsentation einer Menge verwendet Rot-Schwarz-Bäume zur Implementierung. Gesucht sind Abstraktionsfunktion und konkrete Invariante in Pseudo-Haskell.

```
class TreeSet<T extends Comparable<T>> implements Set<T> {
    private class RedBlackNode {
        boolean isBlack;
        T root;
        RedBlackNode left, right, parent;
    }
    RedBlackNode root;
    ...
}
```

### Aufgabe 2 (Iteratoren und Bäume)

Häufig interessiert man sich für diejenigen Elemente  $x$  einer linear geordneten Menge  $M$ , die zwischen zwei vorgegebenen Werten  $a$  und  $b$  liegen, z.B. alle Namen in einem Namensverzeichnis, die mit dem Buchstaben  $Y$  beginnen:

$$\{ x \mid x \in M, "Y" \leq x < "Z" \}$$

Diese Elemente kann man zwar mit einer filternden Traversierungsoperation finden, eine gezielte Suche, die einen speziellen externen Iterator zurückliefert, ist aber in der Regel effizienter. Eine Schnittstelle mit Bereichssuche sehe wie folgt aus ( $from$  und  $to$  seien im Bereich enthalten):

```
interface Set2<T extends Comparable<T>> {
    Iterator<T> list(T from, T to);
    void add(T e);
    boolean contains(T e);
}
```

Implementieren Sie eine Klasse `ThreadedTreeSet<T extends Comparable<T>> implements Set2<T>` für binäre Suchbäume in gefädelter Darstellung und testen Sie Ihre Implementierung. Die Methode `remove` des Iterators brauchen sie nicht zu implementieren. Wie reagiert Ihr Iterator, wenn die Menge geändert wird, während eine Traversierung erfolgt?