Praktische Informatik II SS 2007

Übungsblatt 7

Abgabe: 18.06.2007

Aufgabe 1 Bäume schön zurechtgestutzt

In der Vorlesung habt ihr AVL-Bäume als Möglichkeit kennengelernt, degenerierte Bäume zu vermeiden und so effiziente Suchbäume aufzubauen.

Aufgabe 1.1 Bäume pflanzen - extended version (80%)

Implementiert eine Klasse

public class AVLTree <Type extends Comparable <Type >>

in Java, welche intern AVL-Bäume zur sortierten Speicherung von generischen Datentypen verwendet.

Stellt in eurer Klasse mindestens die folgenden drei Methoden zur Verfügung:

void insert(Type t): Fügt Eintrag `t` vom Typ `Type` sortiert in den AVL-Baum ein und rebalanciert ihn dabei wieder.

void remove(Type t): Löscht `t` aus dem AVL-Baum und rebalanciert ihn dabei wieder.

boolean contains(Type t): Gibt genau dann *true* zurück, wenn `t` in dem sortierten Baum vorkommt.

Für eine halbwegs vernünftige Darstellung des Baums soll ausserdem die Methode *toString()* überschrieben werden.

String toString(): Gibt die Struktur des AVL-Baumes in geklammerter Form als String zurück, d.h. zum Beispiel ein Baum mit Inhalten „`Hinz`“ und „`Kunz`“ vom Typ String führt zu der Ausgabe `((-, Hinz, -), Kunz, -)`, falls „`Kunz`“ in der Wurzel gespeichert ist („`-`“ steht für den leeren Baum).

Die interne Datenstruktur eurer AVL-Bäume soll dabei nach außen nicht sichtbar, d.h. vollständig gekapselt sein. Legt für die Knoten eine Klasse *Node* an, die ebenfalls vollständig gekapselt ist.

Aufgabe 1.2 Pfade im Wald (20%)

Gebt eine *kürzeste* Folge von Einträgen an, mit der alle Fallunterscheidungen der intern verwendeten Funktionen *rebal*, *shiftr*, *shiftl*, *rotr* und *rotl* abgedeckt werden können.