

## 7. Übungsblatt

**Ausgabe:** 01. Dezember 2010    **Abgabe:** 08. Dezember 2010, 10 Uhr

Die Bearbeitung in Zweiergruppen ist ausdrücklich erwünscht.

### Aufgabe 1: AVL-Bäume

10 Punkte

- (a) Geben Sie eine Folge von Zahlen  $(a_1, \dots, a_k)$  an, so dass das Ausführen der Operationen  $\text{insert}(a_1), \dots, \text{insert}(a_k)$  in einen AVL-Baum zu vier verschiedenen Fällen von Unbalanciertheit führt; in der Notation des Skriptes sollen diese Fälle den Kombinationen

- $v$  ist {linkes|rechtes} Kind von  $u$
- ×
- $w$  ist {linkes|rechtes} Kind von  $v$

entsprechen. Skizzieren Sie den Anfangszustand des Baumes, sowie die Zustände nach jedem Ausführen einer Rotation.

Die Anzahl der Einfügeoperationen  $k$  und den Anfangszustand des Baumes können Sie frei wählen; im eigenen und unserem Interesse aber natürlich möglichst klein.

Wenn Sie möchten, können Sie verschiedene Anfangszustände verwenden, so dass jeweils eine **insert** Operation zum gewünschten Verhalten führt.

- (b) Geben Sie einen AVL-Baum sowie eine Folge von **remove** Operationen an die zu einer einfachen, einer doppelten und einmal zu mindestens zwei Rotationen (einfach oder doppelt) führen. Skizzieren Sie den Anfangszustand des Baumes, sowie die Zustände nach jedem Ausführen einer Rotation.

Wenn Sie möchten, können Sie verschiedene Anfangszustände verwenden, so dass jeweils eine **remove** Operation zum gewünschten Verhalten führt.

[Bitte wenden]

## Aufgabe 2: Suchbäume

10 Punkte

- (a) Realisieren Sie einen (unbalancierten) binären Suchbaum. Leiten Sie dazu eine Klasse `u07.name.BinarySearchTree` von der abstrakten Klasse `material.u07.AbstractBinaryTree` ab. (5 Punkte)
- (b) Die Höhe  $h(v)$  eines Knotens  $v$  ist definiert durch

$$h(v) = \begin{cases} 0 & \text{wenn } v \text{ ein Blatt ist} \\ 1 + \max(h(v.left), h(v.right)) & \text{sonst} \end{cases}$$

Die Höhe des Baumes entspricht dann der Höhe des Wurzelknotens.

Implementieren Sie die Methode `update(BinaryTreeNode node)` und verwenden Sie diese in `insert` und `delete`, so dass nach diesen Operationen die Höheninformation an den notwendigen Knoten effizient aktualisiert wird. (3 Punkte)

Achten Sie auf eine lesbare Implementierung, insbesondere auf Einrückungen, sinnvolle Namensgebung (Variablennamen aus dem Skript können beibehalten werden) und stichhaltige Kommentare! (2 Punkte)

Hinweise:

- Beachten Sie die Kommentare in den zur Verfügung gestellten Klassen.
- Sie können annehmen, dass jeder Schlüssel im Suchbaum nur einmal vorkommen darf.
- Achten Sie darauf, die parent-Zeiger der Knoten zu ändern, wenn nötig. (→ aktualisiertes Skript)