7. Übung Algorithmen und Datenstrukturen WS 2016/17

Klaus Kriegel **Abgabe: 12.12.2016, 12:00 Uhr**

Aufgabe 1: Höhen-Balance-Eigenschaft (HBE) (3 + 2 + 1 Punkte)

Die Folge der Fibonacci-Zahlen ist rekursiv definiert durch $f_0 = 0$, $f_1 = 1$ und $f_n = f_{n-2} + f_{n-1}$ für alle $n \geq 2$.

a) Zeigen Sie (mit Induktion nach h), dass jeder echte Binärbaum der der Höhe $h \geq 1$ mit der HBE mindestens f_{h+1} innere Knoten hat. Achtung, es stimmt nicht für $h = 0$. Beachten Sie, dass der Induktionsanfang für zwei Werte gezeigt werden muss.

b) Verschärfen Sie die Aussage von a), indem Sie zeigen, dass jeder echte Binärbaum der der Höhe $h \geq 1$ mit der HBE mindestens $f_{h+2} - 1$ innere Knoten hat.

c) Zeigen Sie, dass es für jede Höhe $h \geq 1$ einen echten Binärbaum T_h der Höhe $h \geq 1$ mit $f_{h+2} - 1$ inneren Knoten gibt, der die HBE hat.

Aufgabe 2: AVL-Bäume I (3 + 3 Punkte)

a) Fügen Sie die Zahlen $7, 10, 4, 6, 5, 2, 8$ schrittweise in einen AVL-Baum ein, der am Anfang leer ist. Skizzieren Sie dazu den Zustand des Baums jeweils nach dem Einfügen eines neuen Elements und (falls notwendig) nach der anschließenden Rotation. Verwenden Sie dabei immer die in der Vorlesung besprochenen Methoden!

b) Löschen Sie aus dem obigen Baum erst den Eintrag 4, und dann den Eintrag 7 und stellen Sie wieder den Zustand nach jeder Löschoperation und (falls notwendig) nach den anschließenden Rotationen dar.

Aufgabe 3: AVL-Bäume II (6 Punkte)

Implementieren Sie Methoden `boolean checkSearchTree()` und `boolean checkHBE()`, die für einen echten Binärbaum mit `int`-Einträgen in den inneren Knoten (Blätter ohne Einträge) überprüfen, ob der Baum ein binärer Suchbaum ist, bzw. ob er die HBE hat. Sie können die Implementierung wahlweise in Java oder Python ausführen. Wenn Sie sich für Java entscheiden, können Sie einen minimalistischen Klassenrahmen aus der gegebenen BinTree-Klasse abschreiben und anpassen (`int` an Stelle von `E`).

Aufgabe 4: AVL-Bäume III (3 Punkte)

Nehmen Sie an, dass Sie eine Implementierung für AVL-Bäume gegeben haben, in der man zusätzlich zu den üblichen Binärbaum-Methoden zwei Methoden `insertKey(K k)` und `deleteKey(K k)` gegeben hat, welche das Einfügen und Löschen inklusive der erforderlichen Rotationen ausführen (zur Vereinfachung bestehen die Einträge nur aus Schlüsseln vom Typ `K`).
Sie wollen diese Datenstruktur zu Realisierung einer Prioritätswarteschlange einsetzen und müssen dazu Methoden `insertQ(K k)` und `K extractMin( )` implementieren. Skizieren Sie eine Idee, wie man beide Operationen mit logarithmischer Laufzeit realisieren kann und beschreiben Sie eine Implementierung in Pseudocode.