Prof. Dr. Leif Kobbelt
Stefan Dollase, Ira Fesefeldt, Alexandra Heuschling, Gregor Kobsik

# Übung 8

## Tutoraufgabe 1 (AVL-Baum):

Führen Sie die folgenden Operationen beginnend mit einem anfangs leeren *AVL-Baum* aus und geben Sie die entstehenden Bäume nach jeder *Einfüge-* und *Löschoperation* sowie jeder *Rotation* an. Markieren Sie außerdem zu jeder Rotation, welcher Knoten in welche Richtung rotiert wird:

1. 1 einfügen
2. 7 einfügen
3. 6 einfügen
4. 5 einfügen
5. 4 einfügen
6. 3 einfügen
7. 2 einfügen
8. 8 einfügen
9. 10 einfügen
10. 9 einfügen
11. 11 einfügen
12. 1 löschen
13. 3 löschen
14. 2 löschen

## Tutoraufgabe 2 (B-Baum):

Führen Sie folgenden Operationen beginnend mit einem anfangs leeren *B-Baum* mit Grad $t = 3$ aus und geben Sie die dabei jeweils entstehenden Bäume an:

1. 1 einfügen
2. 7 einfügen
3. 6 einfügen
4. 5 einfügen
5. 4 einfügen
6. 3 einfügen
7. 2 einfügen
8. 8 einfügen
9. 10 einfügen
10. 9 einfügen
11. 11 einfügen
12. 7 löschen
13. 8 löschen

## Tutoraufgabe 3 (Rot-Schwarz Bäume – Theorie):

Die Anzahl der schwarzen Knoten auf einem Pfad von der Wurzel eines Rot-Schwarz Baums bis zu einem Blatt (exclusive der Wurzel), ist für einen gegebenen Rot-Schwarz Baum immer dieselbe, unabhängig vom gewählten Pfad. Die Schwarzhöhe eines Rot-Schwarz Baums ist definiert als eben diese Anzahl.

Beweisen Sie (per Induktion) die folgende Aussage:
Ein Rot-Schwarz-Baum der Schwarzhöhe $h$ hat maximal $rot(h) = \frac{2}{3} \cdot 4^h - \frac{2}{3}$ rote Knoten.

## Aufgabe 4 (B-Baum): 10 Punkte

Führen Sie folgenden Operationen beginnend mit einem anfangs leeren *B-Baum* mit Grad $t = 3$ aus und geben Sie die dabei jeweils entstehenden Bäume an:

1. 9 einfügen
2. 6 einfügen
3. 8 einfügen
4. 7 einfügen
5. 2 einfügen
6. 3 einfügen
7. 4 einfügen
8. 5 einfügen
9. 1 einfügen
10. 12 einfügen
11. 8 löschen
12. 5 löschen
13. 7 löschen
14. 3 löschen
15. 9 löschen

## Aufgabe 5 (Max 3 Skip-Liste): 3 + 5 + 2 = 10 Punkte

Sei eine Skip-Liste mit unterschiedlichen Elementen $k_1 \dots k_n$ gegeben. Die Anzahl der Vorwärtskanten des Elements $k_i$ ist gegeben durch $level(k_i)$. Wir definieren nun eine Max 3 Skip-Liste als eine solche Skip-Liste, in der maximal 3 Elemente mit einer Höhe ohne Unterbrechung eines größeren Level vorkommen darf und dass kein level unnötig ist. Mit unnötig meinen wir, dass es zwischen zwei Elementen mit einer Vorwärtskante (oder dem Anfang, bzw das Ende) es mindestens ein Element für jedes kleinere Level gibt. Formal ist eine Skip-Liste genau dann eine Max 3 Skip-Liste

1. wenn für alle $i < i' < i''$ mit $level(k_i) = level(k_{i'}) = level(k_{i''})$ gilt,
   a) dass ein $j$ mit $i < j < i''$ existiert sodass $level(k_j) > level(k_i), level(k_{i'}), level(k_{i''})$, oder
   b) dass kein $j$ mit $i < j < i''$ und $j \neq i'$ existiert sodass $level(k_j) = level(k_i), level(k_{i'}), level(k_{i''})$; und
2. wenn für jedes $i$ alles vier gilt:
   a) dass es für jedes $l < level(k_i)$ ein Element $k_j$ mit $level(k_j) = l$ und $j < i$ existiert,
   b) dass es für jedes $l < level(k_i)$ ein Element $k_j$ mit $level(k_j) = l$ und $j > i$ existiert,
   c) dass für jedes $i'$ mit $i < i'$ und $level(k_i) \leq level(k_{i'})$ und jedes $l < level(k_i)$ ein Element $k_j$ mit $level(k_j) = l$ und $i < j < i'$ existiert, und
   d) dass für jedes $i'$ mit $i > i'$ und $level(k_i) \leq level(k_{i'})$ und jedes $l < level(k_i)$ ein Element $k_j$ mit $level(k_j) = l$ und $i > j > i'$ existiert.

**a)** Geben Sie ein Beispiel einer Max 3 Skip-Liste mit 11 Elementen an, in der genau ein Element Level 3 hat.

**b)** Geben Sie an, wie eine Max 3 Skip-Liste mit maximalem Level $l$ in ein B-Baum mit minimalen Grad 2 und Höhe $l - 1$ transformiert werden kann. Begründen Sie auch die Korrektheit ihrer Transformation, das heißt begründen Sie,
   *i.* dass die Elemente der Skip-Liste die Elemente des B-Baums sind,
   *ii.* dass der B-Baum minimalen Grad 2 hat, und
   *iii.* dass der B-Baum Höhe $l - 1$ hat.

**c)** Erläutern Sie, wie Sie aus ihrer Transformation von Max 3 Skip-Listen auf B-Bäume Methoden zum deterministischen Einfügen und Löschen in einer Max 3 Skip-Liste herleiten können. Es ist nicht nötig, die Methoden im Detail zu erläutern.

## Aufgabe 6 (Programmierung in Python - AVL-Bäume): 2 + 6 + 6 + 6 = 20 Punkte

Bearbeiten Sie die Python Programmieraufgaben. In dieser praktischen Aufgabe werden Sie sich mit AVL-Bäumen auseindersetzen. Diese Aufgabe dient dazu einige Konzepte der Vorlesung zu wiederholen und zu vertiefen.
Zum Bearbeiten der Programmieraufgabe können Sie einfach den Anweisungen des Notebooks *blatt08-python.ipynb* folgen. Das Notebook steht in der .zip-Datei zum Übungsblatt im Lernraum zur Vergügung.
Ihre Implementierung soll immer nach dem # *YOUR CODE HERE* Statement kommen. Ändern Sie keine weiteren Zellen.
Laden Sie spätestens bis zur Deadline dieses Übungsblatts auch Ihre Lösung der Programmieraufgabe im Lernraum hoch. Die Lösung des Übungsblatts und die Lösung der Programmieraufgabe muss im Lernraum an derselben Stelle hochgeladen werden. Die Lösung des Übungsblatts muss dazu als .pdf-Datei hochgeladen werden. Die Lösung der Programmieraufgabe muss als .ipynb-Datei hochgeladen werden.

**Übersicht der zu bearbeitenden Aufgaben:**

**a)** AVL-Bäume
- get_balance()
- rotate_left()
- rotate_right()
- balance()