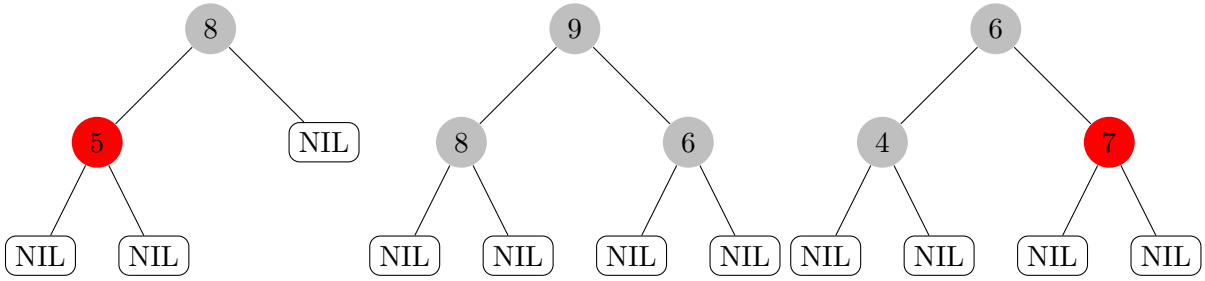
Albert-Ludwigs-Universität
Institut für Informatik
Prof. Dr. F. Kuhn
P. Bamberger
P. Schneider

Algorithmen und Datenstrukturen
Sommersemester 2020
Musterlösung Übungsblatt 7

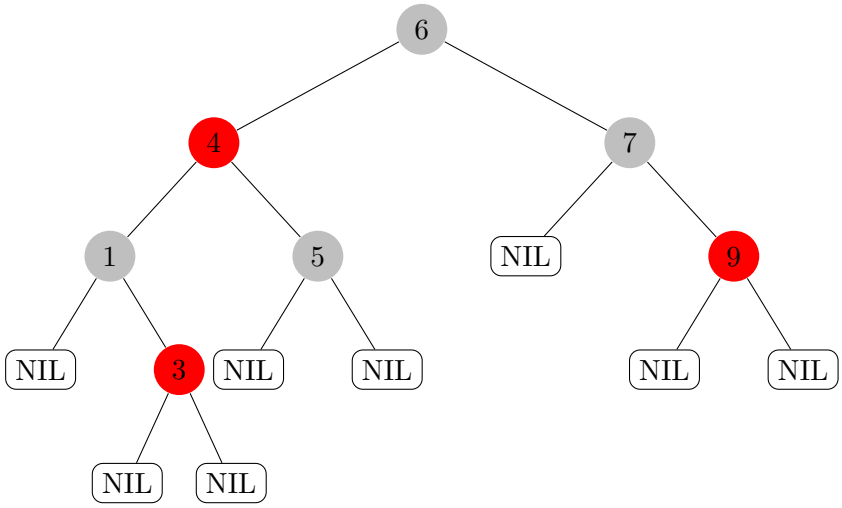
Abgabe: Mittwoch, 01.07.2020, 16:00 Uhr.

Aufgabe 1: Rot-Schwarz Bäume *(10 Punkte)*

(a) Entscheiden Sie für jeden der folgenden Bäume, ob es sich um einen Rot-Schwarz Baum handelt und falls nicht, welche Eigenschaft verletzt ist: *(3 Punkte)*



(b) Führen Sie auf folgendem Rot-Schwarz Baum zuerst die Operation `insert`(8) und danach `delete`(5) aus. Zeichnen Sie den resultierenden Baum. Dokumentieren Sie Ihre Zwischenschritte. *(7 Punkte)*



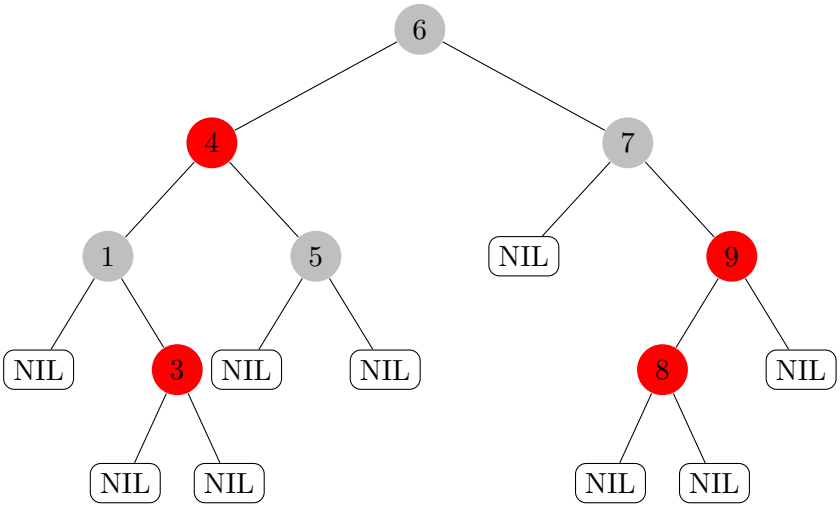
Musterlösung

(a) Von links nach rechts:

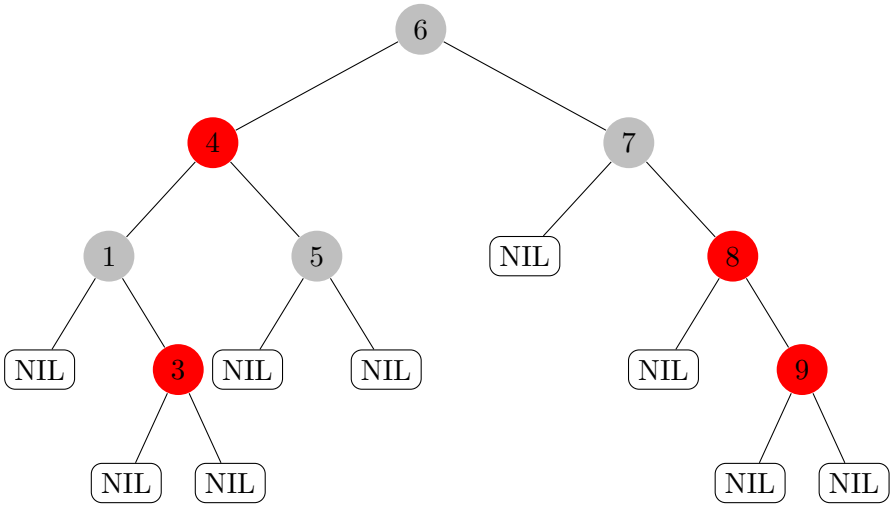
1) Rot-Schwarz-Baum

2) Kein Rot-Schwarz-Baum, da kein binärer Suchbaum (die Wurzel hat ein rechtes Kind mit kleinerem Schlüssel).
3) Kein Rot-Schwarz-Baum: Die Anzahl schwarzer Knoten auf einem Pfad von der Wurzel zu einem Blatt ist größer, wenn man durch den linken Teilbaum geht.

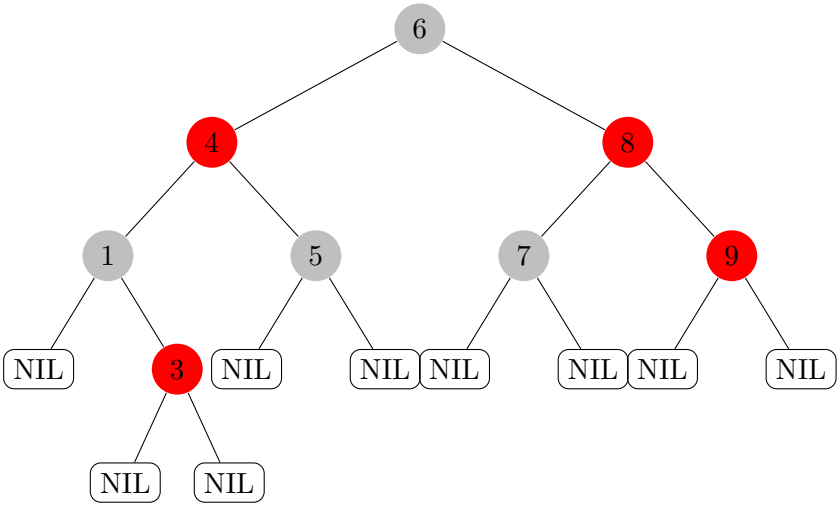
(b) Wir fügen zuerst einen roten Knoten mit Schlüssel 8 ein, so wie man allgemein Schlüssel in binäre Suchbäume einfügt.



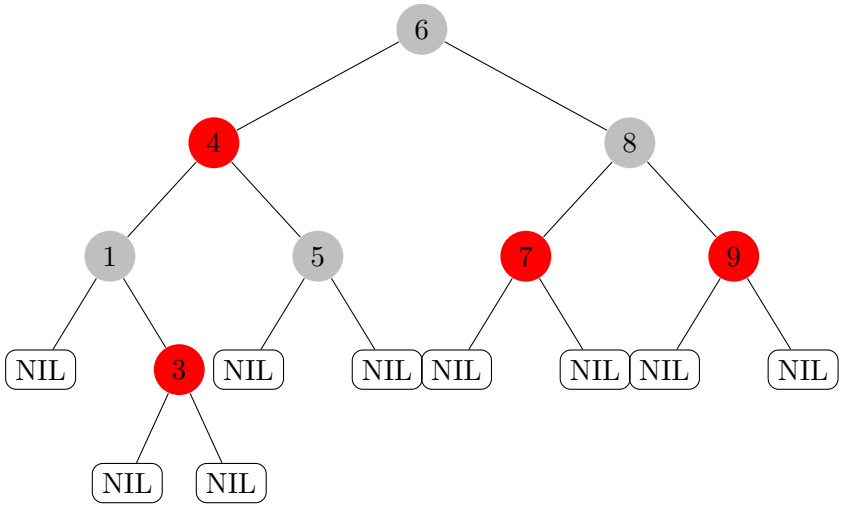
Wir befinden uns im Fall 1b aus der Vorlesung. Wir machen also ein right-rotate(9,8),



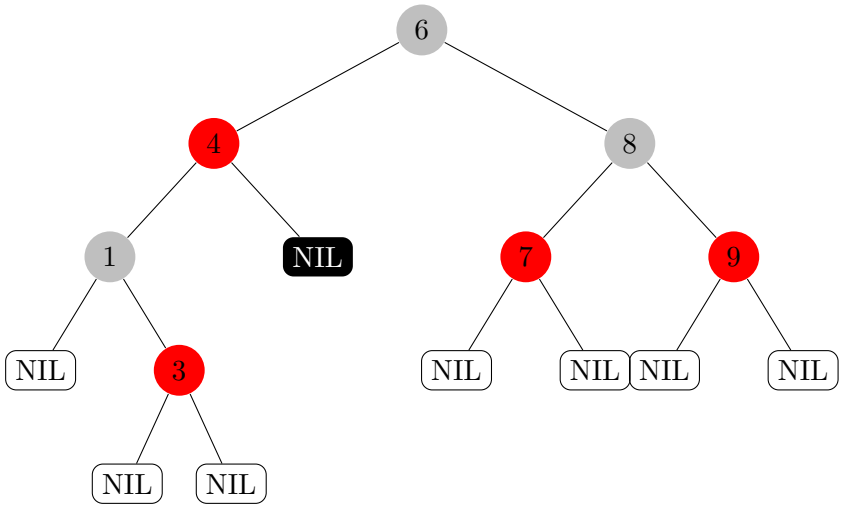
danach ein left-rotate(8,7)



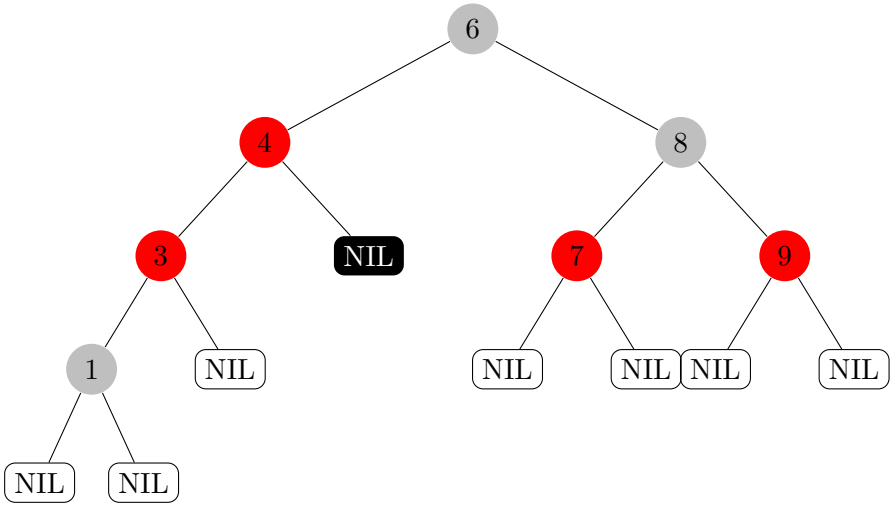
und färben schließlich Knoten 8 schwarz und Knoten 7 rot.



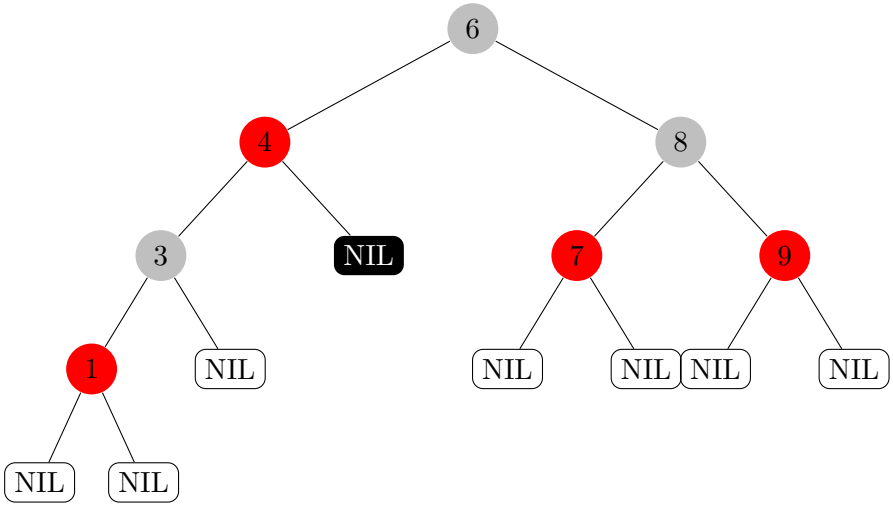
Nun führen wir `delete`(5) aus. Wir befinden uns im Fall 2b aus der Vorlesung (schwarzer Knoten mit zwei NIL-Kindern). Zunächst entfernen wir Knoten 5 aus dem Baum und färben –um die Schwarztiefe zu korrigieren– das rechte NIL-Kind von 4 doppelt schwarz.



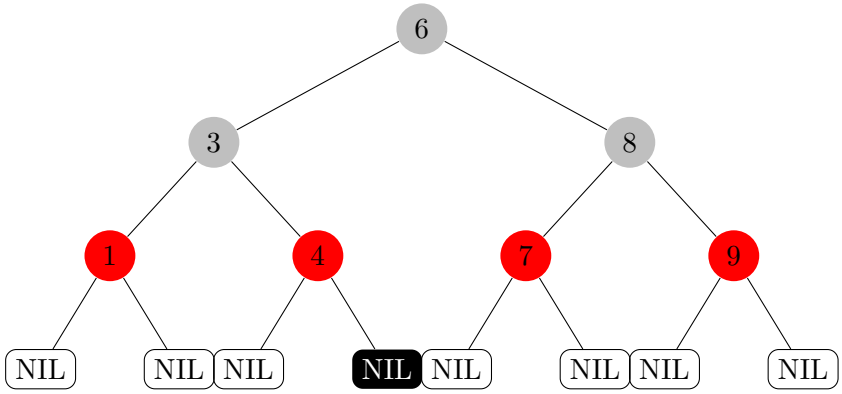
Wir befinden uns im Fall A.2 aus der Vorlesung. Wir machen ein left-rotate(3,1)



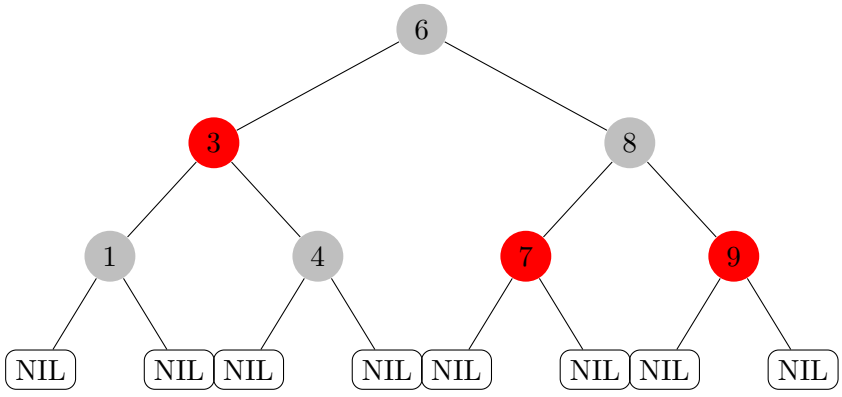
und färben Knoten 1 und 3 um.



Nun befinden wir uns im Fall A.1. Wir machen also ein right-rotate(3,4)



und färben um. Am Ende sieht der Baum so aus



Aufgabe 2: AVL-Bäume *(10 Punkte)*

Ein AVL-Baum ist ein binärer Suchbaum, für den folgende Zusatzeigenschaft gilt. Für jeden Knoten v im Baum ist die Differenz der Höhen des linken und des rechten Teilbaumes von v höchstens 1.

(a) Beweisen Sie *induktiv*, dass ein AVL-Baum der Tiefe d bis zur Tiefe $\lfloor \frac{d}{2} \rfloor$ voll besetzt ist. *(3 Punkte)*

Bemerkung: Ein Baum ist zur Tiefe d' voll besetzt wenn er für alle $x \leq d'$ genau 2^x Knoten mit Tiefe x hat (d.h., Schicht x des Baumes hat die maximale Anzahl Knoten).

(b) Geben Sie die minimale Anzahl von Knoten eines AVL-Baumes in Abhängigkeit von d als Rekursionsgleichung an und begründen Sie. *(3 Punkte)*

Hinweis: Drücken Sie die minimale Anzahl der Knoten eines AVL-Baumes der Tiefe d rekursiv mittels der min. Anzahl Knoten niedrigerer AVL-Bäume aus (mit Basisfällen für Tiefe 0 bzw. 1).

(c) Zeigen Sie dass ein AVL-Baum mit n Knoten Tiefe $\mathcal{O}(\log n)$ hat. *(4 Punkte)*

Hinweis: Benutzen Sie dazu entweder Aussage a) oder Aussage b) mit dem Hinweis dass für die Folge definiert durch $F_0 = 1, F_1 = 2, F_i = F_{i-1} + F_{i-2}, i \geq 2$ gilt dass $F_{i+2} \geq 2F_i$.

Musterlösung

(a) **Induktionsanfang:** Jeder (nicht triviale) Baum hat eine Wurzel, ist also bis zur Tiefe[1] 0 voll besetzt. Die Behauptung gilt also für $d = 0$ und $d = 1$.

Induktionsschluss: Angenommen die Behauptung stimmt für alle AVL-Bäume mit Tiefe höchstens d. Wir zeigen dass dies auch für einen AVL-Baum der Tiefe $d+1$ gilt.

Sei r die Wurzel eines solchen AVL-Baumes T und T_ℓ und T_r der linke bzw. rechte Teilbaum an r. Einer der beiden Teilbäume T_ℓ, T_r muss nun Tiefe d haben, da T Tiefe $d+1$ hat. Der andere muss dann Tiefe mindestens d–1 haben, aufgrund der AVL-Eigenschaft. Der niedrigere von beiden Teilbäumen ist aufgrund der Induktionsvoraussetzung bis Tiefe $\lfloor \frac{d-1}{2} \rfloor = \lfloor \frac{d+1}{2} - 1 \rfloor = \lfloor \frac{d+1}{2} \rfloor - 1$ voll besetzt. Dies gilt natürlich auch für den anderen (tieferen) Teilbaum. Damit haben wir in T eine voll besetzte Schicht mehr (die Wurzel kommt hinzu) weshalb T bis zur Tiefe $\lfloor \frac{d+1}{2} \rfloor - 1 + 1 = \lfloor \frac{d+1}{2} \rfloor$ voll besetzt ist.

(b) Sei N_d die Mindestanzahl von Knoten eines AVL-Baumes der Tiefe d. Ein (nicht trivialer) AVL-Baum der Tiefe 0 hat genau einen Knoten (die Wurzel), wir setzen daher $N_0 = 1$. Ein Baum der Tiefe 1 hat entweder 2 oder 3 Knoten. Wir setzen $N_1 = 2$.

Sei nun $d \geq 2$. Wir definieren N_d rekursiv. Ein AVL-Baum T dieser Tiefe besteht aus einer Wurzel r und einem rechten und linken Teilbaum T_ℓ respektive T_r. Einer der beiden Teilbäume (o.B.d.A. sei das T_r) muss offensichtlich Tiefe $d-1$ haben (da T Tiefe d hat) besteht also aus mindestens N_{d-1} Knoten. Dann muss T_ℓ aufgrund der AVL-Eigenschaft Tiefe mindestens $d-2$ haben besteht also N_{d-2} Knoten. Damit ist die Mindestanzahl der Knoten von T rekursiv gegeben durch

$$N_d = N_{d-1} + N_{d-2} + 1.$$

(c) **Mittels (a)**: Ein AVL-Baum der Tiefe d ist nach (a) bis zur $\lfloor \frac{d}{2} \rfloor$-ten Schicht voll besetzt. D.h., dieser Baum hat mindestens $n \geq 2^{\lfloor d/2 \rfloor}$ Knoten (wir brauchen nur eine sehr grobe Abschätzung für die Asymptotik). Damit ist

$$\begin{aligned}
& 2^{\lfloor \frac{d}{2} \rfloor} \leq n \\
\iff \quad & \lfloor \frac{d}{2} \rfloor \leq \log(n) \\
\implies \quad & \frac{d}{2} - \frac{1}{2} \leq \lfloor \frac{d}{2} \rfloor \leq \log(n) \\
\implies \quad & d \leq 2 \log n + 1 \\
\implies \quad & d \in \mathcal{O}(\log(n)).
\end{aligned}$$

Mittels (b): Fast analog zur Fibonacci-Folge aus dem Hinweis haben wir $N_d = N_{d-1} + N_{d-2} + 1 = 2N_{d-2} + N_{d-3} + 2 \geq 2N_{d-2}$. Anschaulich heißt das, dass sich die Mindestanzahl der Knoten in einem AVL-Baum mindestens alle zwei "Tiefenschritte" verdoppelt. Konkret heißt das $N_d \geq 2N_{d-2} \geq 2^2 N_{d-4} \geq \cdots \geq 2^{\lfloor d/2 \rfloor} N_{d-2\lfloor d/2 \rfloor} \geq 2^{\lfloor d/2 \rfloor} N_0 = 2^{\lfloor d/2 \rfloor}$. Der Rest geht wie oben.

[1] Für Bäume gilt Tiefe = Höhe!