

Petra Mutzel

Sommersemester 2009

Nicola Beume, Christian Bockermann, Christian Horoba,
Morteza Monemizadeh, Ingo Schulz, Dirk Sudholt, Christine Zarges

DAP2 – 2. Übungstest

Datum: 16. Juni 2009 — Gruppe: Gelb

Matrikelnummer:	
Vorname:	
Nachname:	
Tutor/in:	

Der Test dauert **30 Minuten**. Es sind keine Unterlagen oder sonstige Hilfsmittel erlaubt. Mit dem Beginn des Ausfüllens dieses Tests gilt die Prüfungsfähigkeit als bestätigt. Täuschungsversuche führen zu einer Bewertung mit 0 Punkten. Bitte **leserlich** schreiben. Unlesbares wird als falsch gewertet.

Aufgabe 1 (4 Punkte)

Kreuze für jede der unten aufgeführten Beziehungen *alle* Zeichen aus ω , Ω , Θ , o und O an, durch die man \star ersetzen kann um eine wahre Aussage zu erhalten. Hier brauchst du die Aussagen nicht zu beweisen, allerdings sollen für jeden Fall *alle* zulässigen Ersetzungen für \star angegeben werden, also *nicht* nur eine zulässige oder die genaueste Charakterisierung der linken Seite.

(a) $n^2\sqrt{n} + n^2 = \star(n^2 \log n + n^3)$ o O Θ Ω ω

(b) $(n - 1)(n + 1) = \star(n\sqrt{n})$ o O Θ Ω ω

(c) $n + \log n = \star(n)$ o O Θ Ω ω

(d) $(n^3 + 3n^2 + 3n + 1) = \star(n^4)$ o O Θ Ω ω

Aufgabe 2 (4 Punkte)

Wir betrachten *Priority-Queues*, die durch *Min-Heaps* implementiert werden. Gegeben sei die *Priority-Queue* P durch die Array-Darstellung

$$\langle 3, 5, 4, 9, 11, 14, 12, 13 \rangle.$$

Anmerkung: Wir speichern hier in der *Priority-Queue* nur die Prioritäten und keine Werte.

(a) Gib den dadurch repräsentierten *Min-Heap* in Baumdarstellung an.

(b) Führe auf P die Operation $Insert(6)$ aus und gib den resultierenden *Min-Heap* in Baumdarstellung an.

- (c) Betrachte wieder die angegebene Priority-Queue P ohne den in (b) eingefügten Schlüssel. Führe danach auf P die Operation $ExtractMin()$ aus und gib den resultierenden *Min-Heap* in Baumdarstellung an.

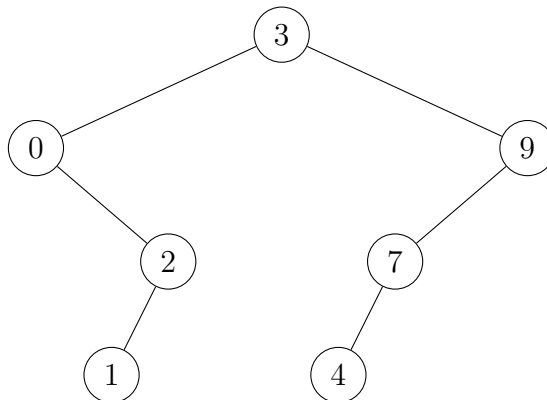
Aufgabe 3 (4 Punkte)

- (a) Gegeben sind die folgenden Traversierungsreihenfolgen eines binären Baums. Rekonstruiere den Baum aus den Traversierungen.

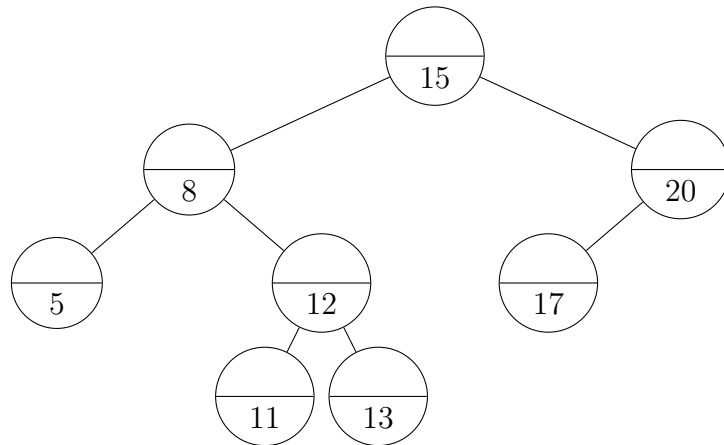
Inorder: 1, 7, 2, 9, 5, 3, 10, 8, 6

Postorder: 7, 1, 9, 3, 10, 6, 8, 5, 2

- (b) Lösche den Schlüssel 3 in folgendem binären Suchbaum und zeichne den resultierenden binären Suchbaum. Verwende dabei die Implementierung von DELETE aus dem Skript, die auf der Funktion PREDECESSOR basiert.



Aufgabe 4 (4 Punkte)

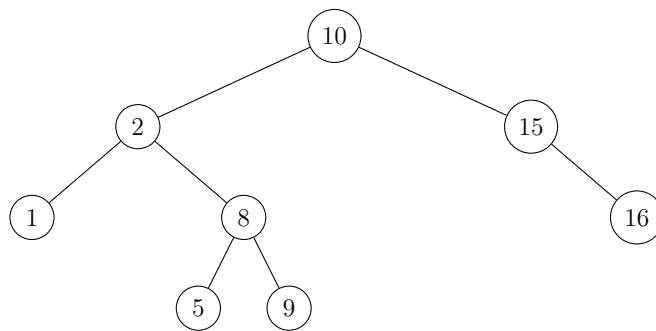


- (a) Trage in die obere Hälfte jedes Knotens im oben angegebenen AVL-Baum die entsprechenden (Höhen-)Balancewerte ein.
- (b) Zeichne den AVL-Baum, der sich durch Einfügung eines Knotens mit dem Schlüssel 14 in den oben angegebenen AVL-Baum ergibt. **Balancewerte müssen nicht angegeben werden.**

Aufgabe 5 (4 Punkte)

In dieser Aufgabe wird jede richtige Antwort mit einem Punkt bewertet, **für jede falsche Antwort wird ein Punkt abgezogen**. Nicht beantwortete Fragen gehen nicht in die Bewertung ein. Durch Minuspunkte kannst du für diese Aufgabe insgesamt **nicht weniger als 0 Punkte** bekommen.

- (a) Der kleinste Schlüssel in einem binären Suchbaum ist immer in einem Knoten maximaler Tiefe abgelegt. ja nein
- (b) Bei jeder Einfügung eines Knotens in einen AVL-Baum findet höchstens eine konstante Anzahl von (Doppel-)Rotationen statt. ja nein
- (c) Für jede natürliche Zahl n gibt es einen AVL-Baum mit n Knoten und Höhe $n - 1$. ja nein
- (d) Gegeben ist der folgende AVL-Baum:



Kreuze den AVL-Baum an, der sich durch Löschen des Schlüssels 16 aus dem oben angegebenen AVL-Baum ergibt. Verwende dabei die Implementierung von DELETE aus dem Skript, die auf der Funktion PREDECESSOR basiert.

Lösung 1

```

            graph TD
              8((8)) --- 2((2))
              8 --- 10((10))
              2 --- 1((1))
              2 --- 5((5))
              10 --- 9((9))
              10 --- 15((15))
          
```

Lösung 2

```

            graph TD
              9((9)) --- 8((8))
              9 --- 15((15))
              8 --- 2((2))
              8 --- 5((5))
              5 --- 1((1))
              15 --- 10((10))
          
```

Lösung 3

```

            graph TD
              8((8)) --- 2((2))
              8 --- 9((9))
              2 --- 1((1))
              2 --- 5((5))
              9 --- 10((10))
              10 --- 15((15))
          
```

Platz für Nebenrechnungen