

Name	Übungsgruppe	1	2	3	4	5
Theodor Bajusz	5	x	x	x	x	x
Valerij Dobler	13	x	x	x	x	x
Matz Radloff	6	x	x	x	x	x
Robin Wannags	5	x	x	x	x	x

Übung 1.

Gegeben ist eine Hashtabelle T der Länge $m = 11$ mit Hashfunktion $h(k; i) = (k + i^2) \bmod m$. Tragen Sie die Schlüssel 56, 46, 31, 45, 42, 65, 29, 44, 23 in der angegebenen Reihenfolge in die Hashtabelle ein. Verwalten Sie Kollisionen:

- (a) durch Verkettung ($i = 0$), (2 Punkte)
- (b) durch offene Adressierung. (2 Punkte)

Index	Verkettete Liste
0	<div>44</div>
1	<div>23</div> → <div>45</div> → <div>56</div>
2	<div>46</div>
3	
4	
5	
6	
7	<div>29</div>
8	
9	<div>42</div> → <div>31</div>
10	<div>65</div>

(a) durch Verkettung

Index	Elemente
0	<div>65</div>
1	<div>56</div>
2	<div>46</div>
3	
4	<div>44</div>
5	<div>45</div>
6	<div>23</div>
7	<div>29</div>
8	
9	<div>31</div>
10	<div>42</div>

(b) offene Adressierung



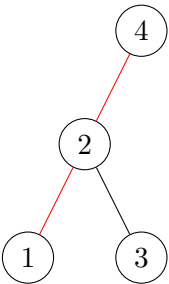
Übung 2.

- (a) Nehmen Sie an, die Suche nach einem Schlüssel k in einem binären Suchbaum endet in einem Blatt. Dann gibt es drei Mengen: A , die Schlüssel links des Suchpfads; B , die Schlüssel auf dem Suchpfad; sowie C , die Schlüssel rechts des Suchpfads. Beweisen oder widerlegen Sie die folgende Behauptung: Für beliebige drei Schlüssel $a \in A, b \in B$ und $c \in C$ gilt $a \leq b \leq c$. (2 Punkte)

Behauptung: Seien B und C wie oben definiert, dann gilt

$$\forall b \in B \forall c \in C : b \leq c$$

Beweis. Sei der Suchbaum wie folgt



und 1 der gesuchte Wert. Dann sehen die Mengen so aus:

$$A = \emptyset, \quad B = \{4, 2, 1\}, \quad C = \{3\}$$

Hierbei ist $4 \in B \wedge 3 \in C$ aber $4 > 3$ ist, was einen Widerspruch zur Behauptung liefert. □

- (b) Aus der Vorlesung ist Ihnen bekannt, dass zum Suchen in einem vollständigen Binärbaum $O(\log_2 n)$ Zeit benötigt wird. Wie ist die asymptotische Laufzeit, wenn stattdessen jeder innere Knoten bis zu 3 Kinder haben darf? Wie, bei bis zu $\log_2 n$ Kindern? (2 Punkte)

Ein ternärer Suchbaum ist ein Suchbaum bei dem jeder innere Knoten bis zu 3 Kindern haben kann. Die Suche in so einem vollständigen Baum benötigt $O(\log_2 n)$ Zeit.

Beweis. In einem ternären Baum muss man pro Knoten bis zu 2 Vergleiche machen um entscheiden zu können, in welchem Kind man weiter suchen muss. Ideal balanciert, würde dessen Höhe $\lceil \log_3 n \rceil$ betragen. In einem vollständigen ternären Suchbaum würde die Suche folglich

$$O(2 \cdot \log_3 n) = O\left(2 \cdot \frac{\log_2 n}{\log_2 3}\right) = O(\log_2 n)$$

Zeit benötigen. \square

Teil zwei. Die Höhe eines solchen Suchbaumes wächst in $O(\log_2(n))$ und die Anzahl der nötigen Vergleiche pro Knoten wächst auch in Abhängigkeit von $O(\log_2(n))$. Für die Laufzeit der Suche ergibt sich dann:

$$O(\log_{\log_2(n)}(n) \cdot \log_2(n))$$

Übung 3.

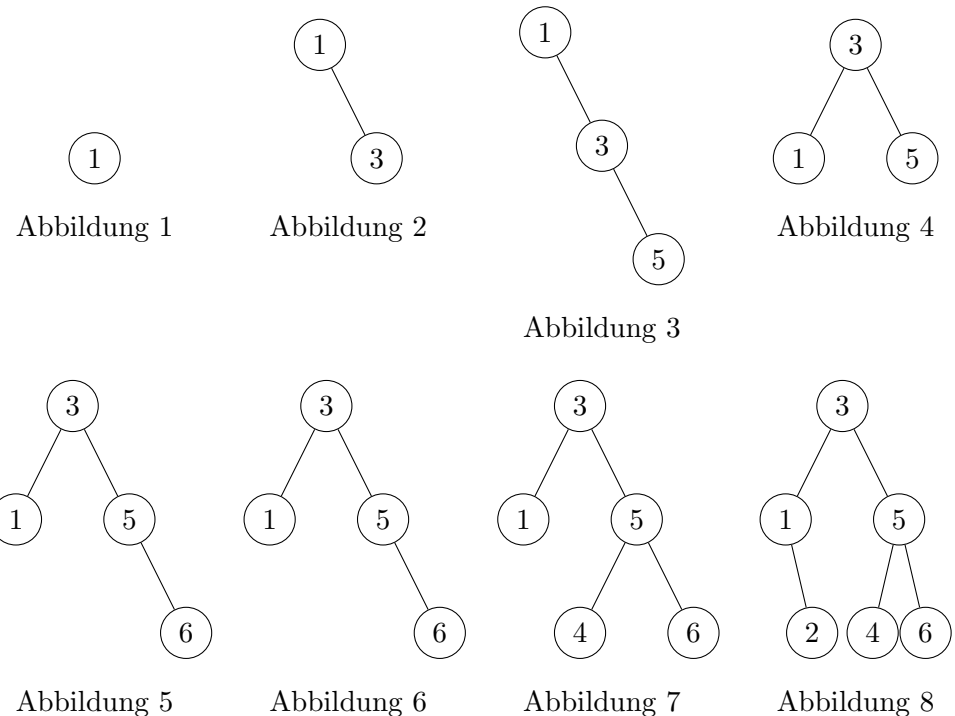
Sei n die Anzahl der Knoten im Baum, und h die Höhe des Baumes.

- (a) Fügen Sie die Schlüssel 1, 3, 5, 6, 6, 4 und 2 in dieser Reihenfolge in einen initial leeren AVL Baum ein. Geben Sie den Baum direkt nach dem Einfügen sowie nach jeder Rotation an. (2 Punkte)

Zuerst fügen wir in den leeren Baum unseren ersten Knoten 1 ein (siehe Abbildung 1).

Als zweiten Schritt fügen wir die 3 ein. Da $3 > 1$ ist, fügen wir diese als rechtes Kind ein (siehe Abb. 2).

Schritt Nummer drei ist das Einfügen von der 5. Da sowohl $5 > 1$ und $5 > 3$ ist, wird die 5 das rechte Kind der 3 (Abb. 3).



Nun wird die Bedingung des AVL Baumes verletzt, da 3 die Höhe 1 hat, das theoretische linke Kind von 1 jedoch die Höhe -1 hat. Wir rotieren links um die 3 um die Eigenschaft wiederherzustellen (siehe Abb. 4).

Als fünften Schritt fügen wir eine 6 in den Baum ein. $6 > 3$ und $6 > 5 \implies 6$ wird das rechte Kind von 5 (siehe Abb. 5).

Als sechsten Schritt versuchen wir erneut eine 6 in den Baum einzufügen, da unsere **AVL-Insert**-Prozedur keine Duplikate zulässt, bleibt der Baum unverändert (siehe Abb. 6).

Schritt 7 ist es eine 4 in den AVL Baum einzufügen. Es gilt $4 > 3$, aber $4 < 5$, weswegen die 4 als linkes Kind der 5 platziert wird (siehe Abb. 7).

Als letzten Schritt fügen wir eine 2 hinzu. $2 < 3$, aber $2 > 1$, deshalb wird die 2 zum rechten Kind der 1. Somit wurden alle Schlüssel hinzugefügt und der AVL Baum ist komplett (siehe Abb. 8). \square

- (b) Zeigen Sie, für einen beliebigen AVL-Baum gilt: $n \leq 2^{h+1} - 1$ (2 Punkte)

Beweis. Für vollständige Binärbäume der Höhe h wissen wir dass diese $n = 2^{h+1} - 1$ Knoten haben. Jeder andere AVL-Baum der Höhe h muss folglich weniger Elemente besitzen, sodass $n \leq 2^{h+1} - 1$ gilt. \square

- (c) Zeigen Sie, für einen beliebigen AVL-Baum gilt: $(\frac{3}{2})^h \leq n$ (2 Punkte) Für einen AVL-Baum der Höhe h gilt $\frac{3}{2}^h < \phi^h \leq n$

Beweis. Sei $N(h)$ die minimale Knotenanzahl eines AVL-Baumes der Höhe h . Wir stellen eine Rekurrenzgleichung mit $N(0) = 1$ und $N(1) = 2$ auf: Für $n \geq 2$ sei h_L und h_R bezeichnen entsprechend die Höhe des linken- und rechten Teilbaums. Weil der Baum die Höhe h hat, muss einer der Teilbäume die Höhe $h - 1$ haben, angenommen es sei h_L . Um die Anzahl der Knoten zu minimieren, machen wir den anderen Teilbaum, h_R , so klein wie möglich. Wegen der AVL Eigenschaft folgt daraus $h_R = h - 2$. Zählt man nun den Wurzelknoten, die Anzahl der Knoten im rechten- und linken Teilbaum bekommt man folgende Rekurrenzgleichung für die totale Anzahl an Knoten:

$$\begin{aligned} N(0) &= 1 \\ N(1) &= 2 \\ N(h) &= 1 + N(h_L) + N(h_R) \\ &= 1 + N(h - 1) + N(h - 2). \end{aligned}$$

Wir stellen fest, dass die Anzahl der Blätter der Fibonacci-Reihe folgt. Da wir eine untere Schranke suchen, ignorieren wir den Term $+1$ der Wurzel. Also gilt $N(h) \geq \phi^h \approx 1.618^h$ 

Folglich muss $N(h) < n$ sein, sodass gilt:

$$\left(\frac{3}{2}\right)^h < \phi^h \leq n.$$

\square



Übung 4.

Beweisen oder widerlegen Sie folgende Aussagen zu ungerichteten Graphen:

- (a) Es gilt für $G = (V, E)$:

$$\sum_{v \in V} \deg(v) = 2|E|$$

(2 Punkte)

Beweis. Sei G ein kantenloser Graph mit mindestens 2 Kanten. Für G gilt $\deg(V_i) = 2 \cdot 0 = 0$.

Für jede Kante $e = (v_1, v_2)$, die wir hinzufügen, erhöhen wir den Grad für v_1 und v_2 um jeweils 1. Da eine neue Kante immer 2 Knoten verbindet, muss die Summe aller Grade immer doppelt so groß sein, wie die Anzahl der Kanten. Die Behauptung stimmt. \square

- (b) Seien v, w die einzigen beiden Knoten in $G = (V, E)$ mit ungeradem Grad, so sind v und w über einen Pfad in G verbunden. (2 Punkte)

Hinweis: Nutzen Sie die Aussage aus Aufgabe a).

Beweis durch Widerspruch: Seien v und w Knoten in zwei jeweils nicht zusammenhängenden Teilgraphen. Aus Aufgabe 4a wissen wir, dass v nicht der einzige Knoten mit ungeradem Grad in seinem Teilgraph sein kann, daher muss es einen zweiten Knoten mit ungeradem Grad in diesem Teilgraph geben. Da v und w die einzigen Knoten mit ungeradem Grad sind, müssen beide in einem zusammenhängenden Teilgraph liegen. Folglich gibt es einen Pfad zwischen v und w . \square

(c) $G = (V, E)$ ist zusammenhängend, wenn für alle Knoten $v \in V$ gilt:

$$\deg(v) \geq \left\lceil \frac{|V| - 1}{2} \right\rceil$$

(2 Punkte)

In der Vorlesung haben wir ungerichtete Graphen als schleifenfrei eingeführt (Foliensatz 5, Seite 3).

Angenommen G ist nicht zusammenhängend, dann existiert o.B.d.A. eine kleinste Zusammenhangskomponente G_Z :

$$\begin{aligned} G_Z &= (V_Z, E_Z) \\ V_Z &\subsetneq V \\ E_Z &= \{\{e_i, e_j\} \mid e_i, e_j \in V_Z\} \cap E \\ |V_Z| &\leq \left\lfloor \frac{|V|}{2} \right\rfloor. \end{aligned}$$

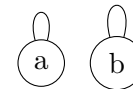
Sei $\deg(v)$ für alle $v \in V_Z$ maximal, dann wäre die Zusammenhangskomponente G_Z ein vollständiger Graph $K_{|V_Z|}$ dann ist $\forall v \in V_Z$:

$$\begin{aligned} \deg(v) &= |V_Z - 1| \\ &= \left\lfloor \frac{|V|}{2} \right\rfloor - 1 = \left\lfloor \frac{|V|}{2} \right\rfloor - 1 \\ &< \left\lfloor \frac{|V|}{2} \right\rfloor + 1 = \left\lceil \frac{|V|}{2} \right\rceil. \end{aligned}$$

Um die Formel aus der Behauptung erfüllen zu können, brauchen alle Knoten in der Zusammenhangskomponente G_Z jeweils eine Kante mehr, da aber alle Knoten in G_Z zu allen anderen Knoten in G_Z eine Kante haben, müssen die neuen Kanten zu anderen Zusammenhangskomponenten gehen. Da das für alle Zusammenhangskomponenten gilt, muss G zusammenhängend sein. \square

Angenommen, man lässt Schleifen zu. Wir betrachten folgenden Graphen G' :

$$\begin{aligned} G' &= (V', E') \\ V' &= \{a, b\} \\ E' &= \{\{a, a\}, \{b, b\}\} \end{aligned}$$



Da Schleifen den Grad eines Knotens um 2 erhöht, gilt offensichtlich sowohl für a als auch für b , dass

$$\deg(a) = \deg(b) = 2 > \left\lceil \frac{|V'| - 1}{2} \right\rceil = \left\lceil \frac{1}{2} \right\rceil = 1$$

, jedoch ist G' nicht zusammenhängend. \nexists



Übung 5.

Bonusaufgabe

Im Kapitel 13-1 im Cormen wird der Rot-Schwarz-Baum, ein spezieller Binärbaum, definiert. Lies das Kapitel durch, und beantworte folgende Fragen zu dem Rot-Schwarz-Baum.

- (a) Geben Sie einen validen Rot-Schwarz-Baum an der aus folgenden Elementen besteht: $\langle 1, 5, 9, 10, 13, 16, 20 \rangle$ (1 Punkt)

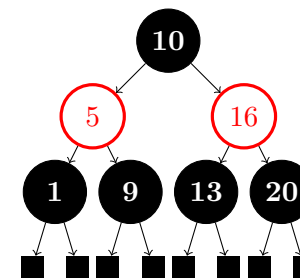


Abbildung 9: Lösung für Aufgabe 5a)

- (b) Zeigen Sie, dass in einem Rot-Schwarz Baum der längste einfache Pfad von einem beliebigen Knoten x zu einem untergeordneten Blatt höchstens doppelt so lang sein kann wie zu einem anderen untergeordneten Blatt. (2 Punkte)

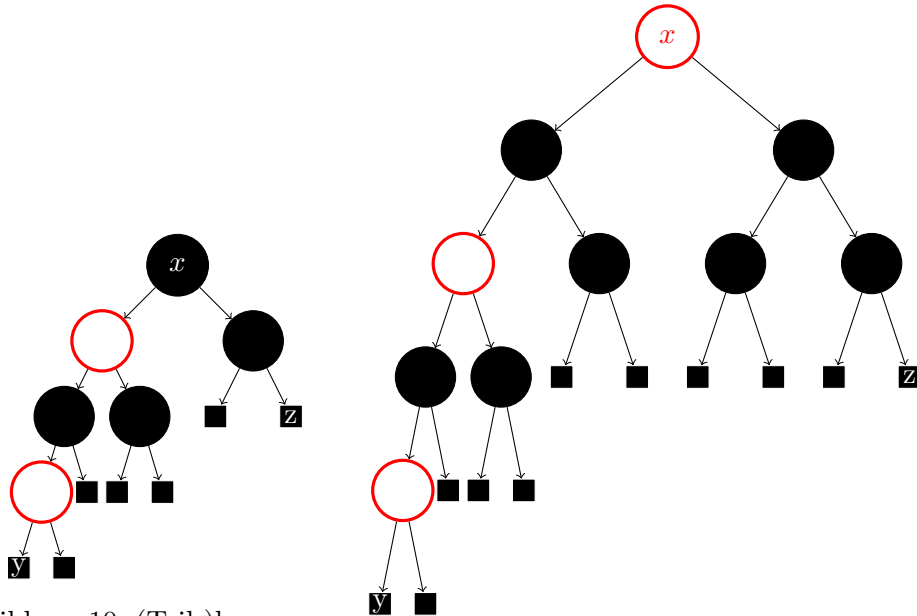


Abbildung 10: (Teil-)baum
von schwarzem x

Abbildung 11: Teilbaum von rotem x

Wir erinnern uns an die Eigenschaften eines Rot-Schwarz Baums (RBT)

1. Jeder Knoten ist entweder *rot* oder *schwarz*.
2. Die Wurzel ist *schwarz*.
3. Die Blätter (NIL-Knoten) sind *schwarz*.
4. Wenn ein Knoten *rot* ist, dann sind seine beiden Kinder *schwarz*.
5. Jeder Knoten x hat die selbe Schwarz-Höhe, d.h. das auf jedem Pfad von x zu dessen Blätter gleich viele *schwarze* Knoten sind.

Behauptung: Seien p_s und p_r Pfade von x zu dessen Blätter. Dabei ist p_s ein möglichst kurzer und p_r ein möglichst langer Pfad. Ferner sei h_s die Höhe des Pfades von p_s , dann gilt für p_r , dass dessen Höhe h_r durch die Ungleichung $h_r \leq 2 \cdot h_s$ beschränkt ist.

Beweis. Da alle Plade von x zu dessen Blätter die selbe Schwarz-Höhe haben müssen (Eigenschaft 5), ist die Anzahl der schwarzen Knoten auf den Pfaden p_r und p_s gleich. Hierbei sieht man leicht, dass p_s lediglich aus schwarzen Knoten besteht (Eigenschaft 1-3). Damit p_r größtmöglich ist, kann nur an jeder zweiten Stelle des Pfades ein roter Knoten sein (Eigenschaft 4).

Falls x selbst schwarz ist, dann kann man nur höchstens h_s rote Knoten in p_r haben, womit p_r dann $h_s + h_s = 2 \cdot h_s$ Knoten hat. Wenn x rot ist, kann der nächste Knoten auf p_r nicht auch rot sein (Eigenschaft 4). Wenn jeder andere zweite Knoten in p_r rot ist, ergibt das p_r dann $h_s + h_s - 1 \leq 2 \cdot h_s$. Falls ein Pfad von x zu einem seiner Blätter weniger rote Knoten als p_r hat gilt die Ungleichung erst recht. \square

- (c) Was ist die größte Anzahl an internen Knoten, aus die ein Rot-Schwarz-Baum bestehen kann wenn die Schwarz-Höhe des Baumes k ist? Was ist die geringste Anzahl? (2 Punkte)

Bei einem RBT der Schwarz-Höhe k ist die größte Anzahl der internen Knoten $2^{2k} - 1$, wobei der Baum dann vollständig mit der Höhe $2k$ ist (insg. $2^{2k+1} - 1$ Knoten) und 2^k NIL-Knoten hat. Jede zweite Ebene ist dabei rot.

$$\underbrace{2^{2k+1} - 1}_{\#gesamt} - \underbrace{2^{2k}}_{\#NIL-Knoten} = \underbrace{2^{2k} - 1}_{\#innere-Knoten} \quad (\mathbf{max})$$

Sei der RBT mit der Schwarz-Höhe k vollständig und alle internen Knoten *schwarz*. Dann ist dessen Höhe auch k und für vollständige Binärbäume der Höhe k gilt:

$$\underbrace{2^{k+1} - 1}_{\#gesamt} - \underbrace{2^k}_{\#NIL-Knoten} = \underbrace{2^k - 1}_{\#innere-Knoten} \quad (\text{min})$$