Matrikelnummer:			
DHBW Duale Hochschule Baden-Württemberg Stuttgart **ÜBUNGSKLAUSUR**		Fakultät Studiengang: Jahrgang / Kurs : Studienhalbjahr:	**Technik** **Angewandte Informatik** **2017 B/C** **2. Semester**
Datum:	**11./12. Juli 2018**	Bearbeitungszeit:	**90 Minuten**
Modul: Unit:	**T2INF1003.1** **Algorithmen**	Dozent:	**Stephan Schulz**
Hilfsmittel:	**Vorlesungsskript, eigene Notizen**		

Aufgabe	erreichbar	erreicht
1	7	
2	6	
3	9	
4	12	
5	8	
6	14	
7	12	
Summe	68	

1. Sind Sie gesund und prüfungsfähig?
2. Sind Ihre Taschen und sämtliche Unterlagen, insbesondere alle nicht erlaubten Hilfsmittel, seitlich an der Wand zum Gang hin abgestellt und nicht in Reichweite des Arbeitsplatzes?
3. Haben Sie auch außerhalb des Klausurraumes im Gebäude keine unerlaubten Hilfsmittel oder ähnliche Unterlagen liegen lassen?
4. Haben Sie Ihr Handy ausgeschaltet und abgegeben?

(Falls Ziff. 2 oder 3 nicht erfüllt sind, liegt ein Täuschungsversuch vor, der die Note „nicht ausreichend“ zur Folge hat.)

Aufgabe 1 (5+2 Punkte)
Betrachten Sie die Folge

$$S = (0, 7, 22, 1, 20, 11, 5, 8, 19, 9)$$

a) Sortieren Sie die Folge S mit dem in der Vorlesung gezeigten Bubble-Sort-Verfahren. Geben Sie hierzu den Zustand von S nach jedem Durchlauf der äußersten Schleife an.

b) Wie viele *Vertauschungen* von Elementen der Folge S benötigen Sie?

Lösung

Original:	0	7	22	1	20	11	5	8	19	9	–
Iteration 1:	0	7	1	20	11	5	8	19	9	22	Swaps: 7
Iteration 2:	0	1	7	11	5	8	19	9	20	22	Swaps: 6
Iteration 3:	0	1	7	5	8	11	9	19	20	22	Swaps: 3
Iteration 4:	0	1	5	7	8	9	11	19	20	22	Swaps: 2
Iteration 5:	0	1	5	7	8	9	11	19	20	22	Swaps: 0

18 Vertauschungen insgesamt

Aufgabe 2 (3+3 Punkte)
Für die folgenden Funktionen sei $x \in \mathbb{N}$.

a) Zeigen oder widerlegen Sie: $5 \cdot \sqrt{x} + x \in \mathcal{O}(x)$

b) Zeigen oder widerlegen Sie: $2^x \in \mathcal{O}(2^{2x})$

Sie können die in der Vorlesung behandelten Resultate verwenden.

Lösung

a) Behauptung: $5 \cdot \sqrt{x} + x \in \mathcal{O}(x)$. Beweis mit Grenzwertkriterium:

$$\begin{aligned} & \lim_{x \to \infty} \frac{5 \cdot \sqrt{x} + x}{x} \\ = & \lim_{x \to \infty} 5 \cdot \frac{\sqrt{x}}{x} + 1 \\ = & \lim_{x \to \infty} 5 \cdot \frac{\sqrt{x}}{\sqrt{x} \cdot \sqrt{x}} + 1 \\ = & \lim_{x \to \infty} 5 \cdot \frac{1}{\sqrt{x}} + 1 \\ = & 1 \in \mathbb{R} \end{aligned}$$

- Behauptung: $2^x \in \mathcal{O}(2^{2x})$. Beweis mit Grenzwertkriterium:

$$\begin{aligned} & \lim_{x \to \infty} \frac{2^x}{2^{2x}} \\ = & \lim_{x \to \infty} \frac{2^x}{2^{x+x}} \\ = & \lim_{x \to \infty} \frac{2^x}{2^x \cdot 2^x} \\ = & \lim_{x \to \infty} \frac{1}{2^x} \\ = & 0 \in \mathbb{R} \end{aligned}$$

Aufgabe 3 (1+2+3+3 Punkte)
Betrachten Sie die folgende C-Funktion.

```
int tuwas(int n)
{
  int i,j,s = 0;

  for (i=n; i>0; i--)
  {
     s = s + i;
  }

  for(i=0; i<n; i++)
  {
     for(j=i; j<i+3; j++)
     {
        s = s + j;
     }
  }
  return s;
}
```

a) Bestimmmen Sie den Rückgabwert für die Eingaben $n = 1, n = 2, n = 3$.

b) Bestimmen Sie das kleinste $k \in \mathbb{N}$, so dass die *Laufzeitkomplexität* von `tuwas()` in $\mathcal{O}(n^k)$ ist. Begründen Sie Ihre Antwort.

Lösung

a) $1 \mapsto 4, 2 \mapsto 12, 3 \mapsto 24$

b) $k = 1$, d.h. die Laufzeitkomplexität von `tuwas()` ist in $\mathcal{O}(n^1)$, oder äquivalent in $\mathcal{O}(n)$.

Begründung:

- Die erste `i`-Schleife läuft von n bis 1, hat also Laufzeit von $c_1 \cdot n$ für eine geeignete Konstante c_1.
- Die zweite `i`-Schleife läuft von 0 bis $n-1$. Die geschachtelte `j`-Schleife läuft immer von i bis $i+2$, also eine *konstante* Zahl von Durchläufen. Damit sind die Kosten der inneren Schleife $3c_2$ und die Kosten der zweiten Schleife insgesamt $((3 \cdot c_2) + c_3) \cdot n$ (für geeignete Konstanten c_2, c_3).
- Als Gesamtkosten ergeben sich also Kosten, die proportional zu n sind. Damit gilt die Behauptung.

Aufgabe 4 (9+3 Punkte)
Gegeben sei die Folge

$$S = (3, 15, 8, 2, 7, 1, 9, 12, 16, 4)$$

a) Sortieren Sie die Folge mit dem in der Vorlesung gezeigten rekursiven (top-down) Mergesort-Algorithmus. Geben Sie hierzu die Teil-Arrays für die jeweiligen rekursiven Aufrufe sowie die Ergebnisse der jeweiligen Merge-Operationen an.

b) Wie oft wird die Funktion `mrg_sort()` insgesamt aufgerufen, also einschließlich des ursprünglichen Aufrufs?

Lösung

a)
```
MergeSort( [3, 15, 8, 2, 7, 1, 9, 12, 16, 4] )
MergeSort( [3, 15, 8, 2, 7] )
MergeSort( [3, 15] )
MergeSort( [3] )
MergeSort( [15] )
Merge( [3] , [15] ) =>  [3, 15]
MergeSort( [8, 2, 7] )
MergeSort( [8] )
MergeSort( [2, 7] )
MergeSort( [2] )
MergeSort( [7] )
Merge( [2] , [7] ) =>  [2, 7]
Merge( [8] , [2, 7] ) =>  [2, 7, 8]
Merge( [3, 15] , [2, 7, 8] ) =>  [2, 3, 7, 8, 15]
MergeSort( [1, 9, 12, 16, 4] )
MergeSort( [1, 9] )
MergeSort( [1] )
MergeSort( [9] )
Merge( [1] , [9] ) =>  [1, 9]
MergeSort( [12, 16, 4] )
MergeSort( [12] )
MergeSort( [16, 4] )
MergeSort( [16] )
MergeSort( [4] )
Merge( [16] , [4] ) =>  [4, 16]
Merge( [12] , [4, 16] ) =>  [4, 12, 16]
Merge( [1, 9] , [4, 12, 16] ) =>  [1, 4, 9, 12, 16]
Merge( [2, 3, 7, 8, 15] , [1, 4, 9, 12, 16] ) =>  [1, 2, 3, 4, 7, 8, 9, 12, 15, 16]
```

b) 19 Aufrufe

Fortsetzung

Aufgabe 5 (3+5 Punkte)

a) Geben Sie für die Rekurrenzrelation G eine möglichst kleine $\mathcal{O}$-Schranke an.

$$G(n) = 27 \cdot G(\left\lfloor \frac{n}{3} \right\rfloor) + 4n^2 - 2n + 42 \quad \text{für } n \in \mathbb{N}$$

b) Der Kompressionsalgorithmus SC zerlegt Graphiken in 3 überlappende Teilbilder, die jeweils die halbe Pixelzahl des Originals haben. Jedes Teilbild mit mehr als einem Pixel wird rekursiv komprimiert, dann werden die komprimierten Teilbilder zusammengefasst. Der Aufwand für das Zusammenfassen von Teilbildern mit der Pixelzahl n ist $z(n) = 4n^3 - n^2 + 20$.

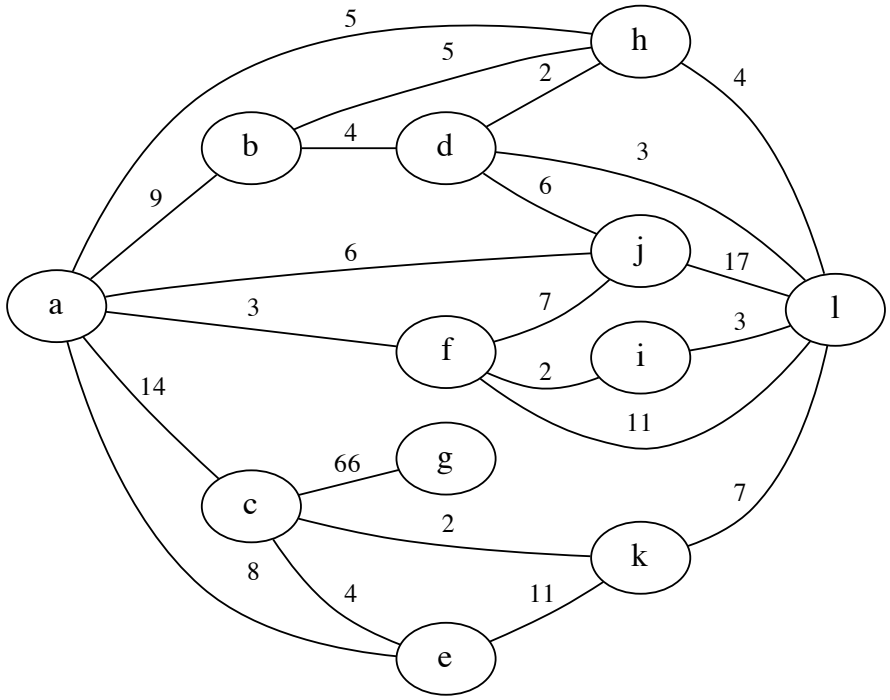
Bestimmen Sie die Komplexität von SC möglichst genau mit Hilfe der $\mathcal{O}$-Notation und begründen Sie Ihr Ergebnis.

Lösung

a) Master-Theorem mit $a = 27, b = 3, d = 2$, also Fall 3: $G \in \Theta(n^{\log_3 27}) = \Theta(n^3)$.

b) $SC(n) = 3SC(\frac{n}{2}) + c(n)$ mit $c(n) \in \Theta(n^3)$. Master-Theorem mit $a = 3, b = 2, d = 3$, also $a < b^d$, also $SC \in \Theta(n^3)$

Aufgabe 6 (5+1+8 Punkte)
Gegeben Sei der folgende Graph G.

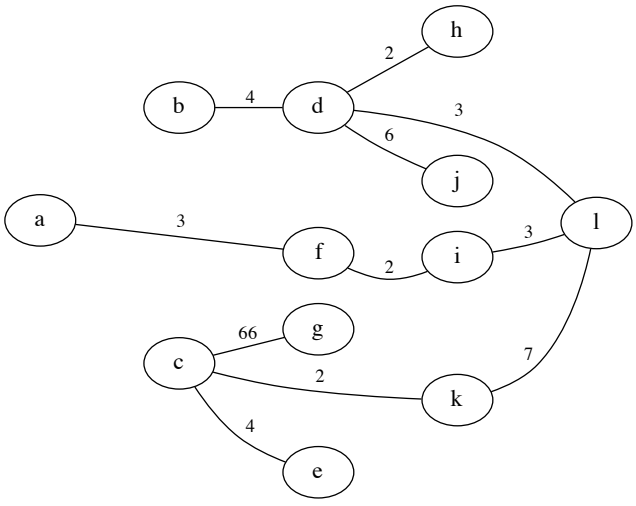


a) Bestimmen Sie für G ausgehend vom Knoten **g** einen minimalen Spannbaum mit Hilfe des Prim-Algorithmus. Sie können hierzu die benutzten Kanten im Bild *sauber* markieren oder eine Liste der verwendeten Kanten angeben. Wie hoch ist das Gesamtgewicht der Kanten des minimalen Spannbaums?

b) Sei (V, E) ein beliebiger ungerichteter zusammenhängender Graph mit $|V| = n$ und der Kantengewichtsfunktion $e : E \to \mathbb{N}$ definiert durch $e(x) = 3$ für alle x. Welches Gesamtgewicht hat der minimale Spannbaum, der aus (V, E) durch die Anwendung von Prims Algorithmus entsteht? Begründen Sie Ihr Ergebnis.

c) Verwenden Sie den Algorithmus von Dijkstra, um die minimale Entfernung aller Knoten in G vom Knoten **a** zu bestimmen. Auf der nächsten Seite finden Sie eine Kopie des Graphen und eine Tabelle für das Ergebnis.

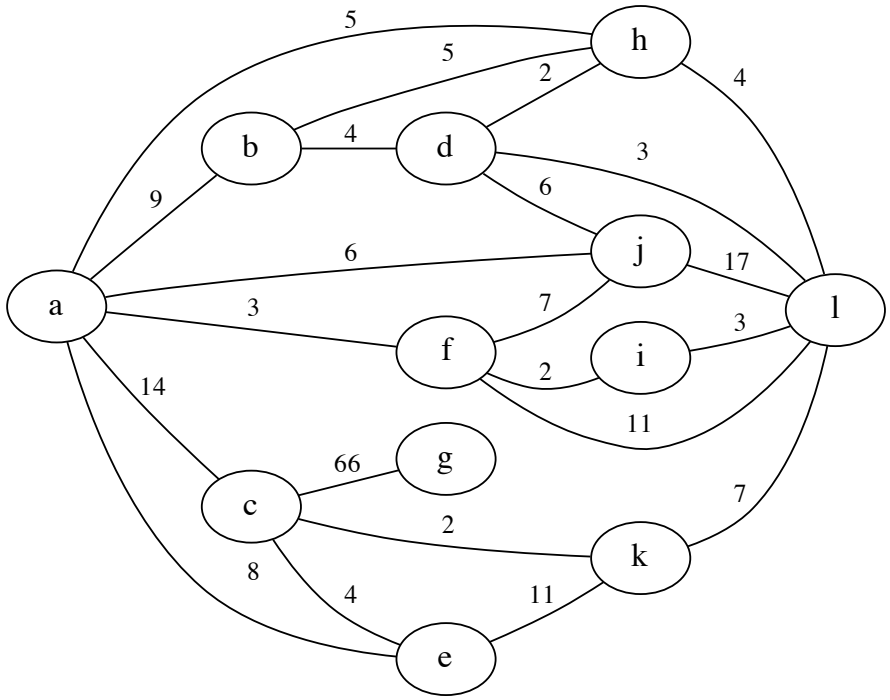
Lösung



a) Gesamtgewicht ist 102.

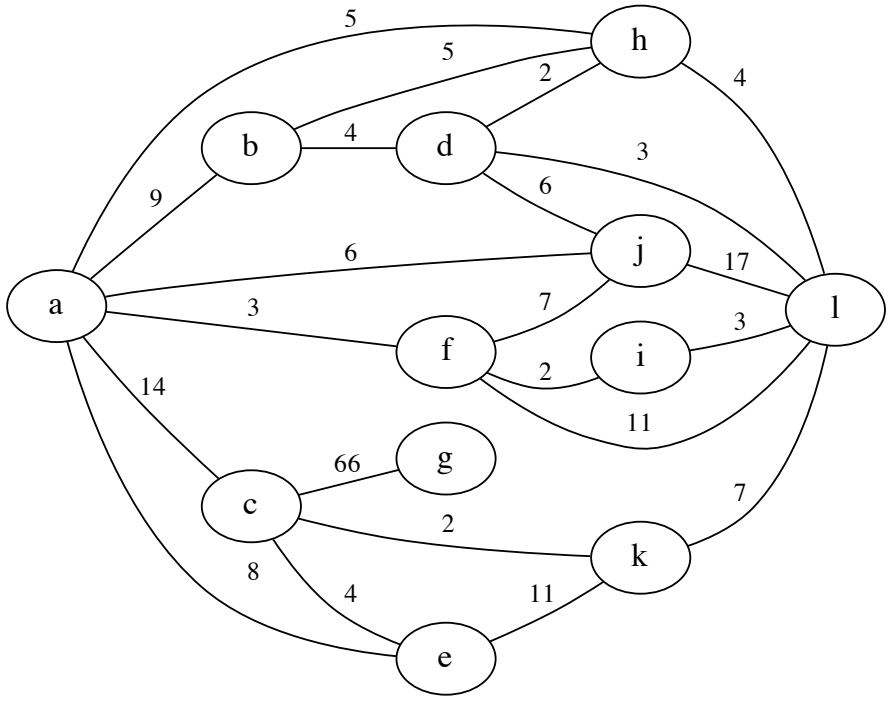
b) Der Baum hat das Gewicht $3n - 3$, da jeder Knoten außer dem ersten durch genau eine Kante an den Spannbaum angebunden wird, und jede Kante das Gewicht 3 hat.

Fortsetzung



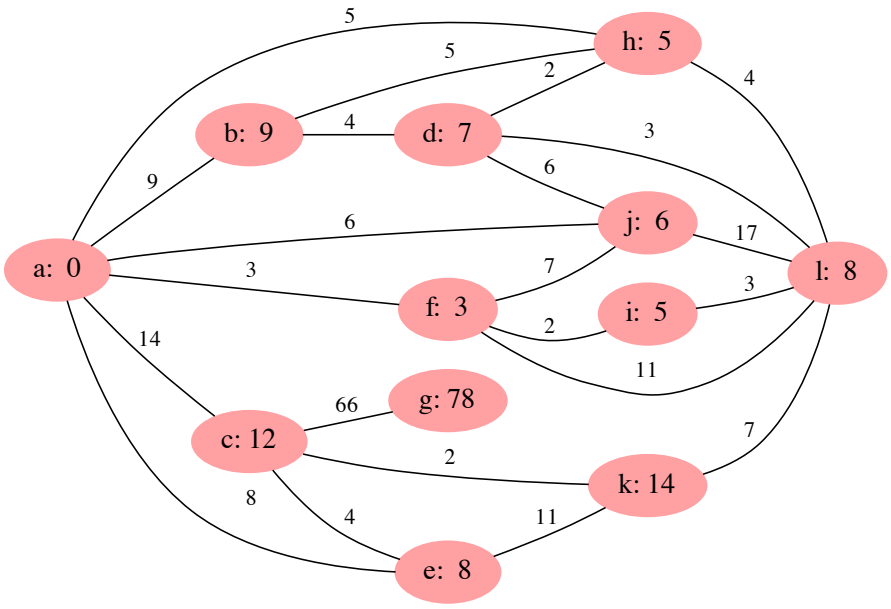
Knoten	Abstand
a	0
b	
c	
d	
e	
f	
g	
h	
i	
j	
k	
l	

Fortsetzung



Lösung

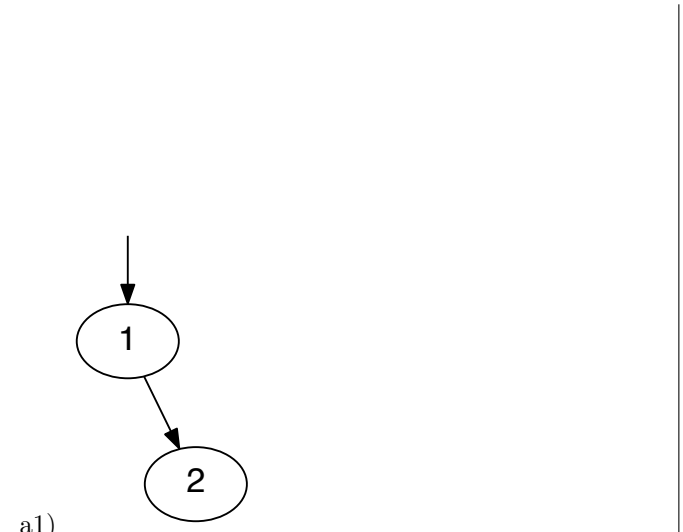
c)



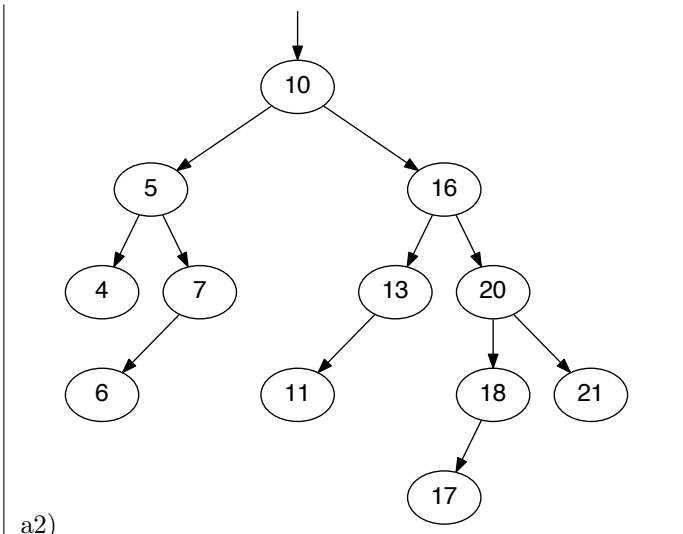
Knoten	Abstand
a	0
b	9
c	12
d	7
e	8
f	3
g	78
h	5
i	5
j	6
k	14
l	8

Aufgabe 7 (4+4+4 Punkte)

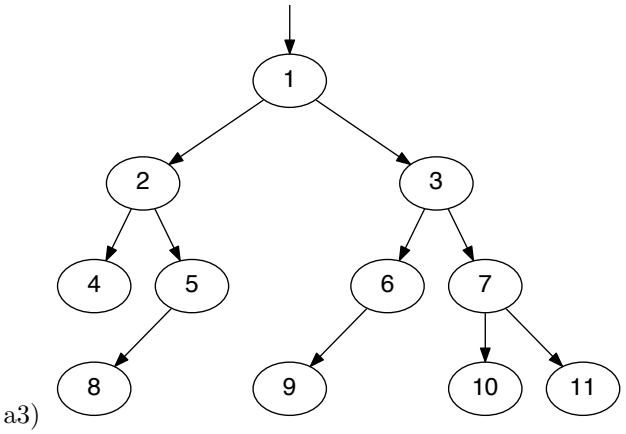
a) Betrachten Sie die folgenden Bäume mit Schlüsseln aus der geordneten Menge $(\mathbb{N}, >)$, d.h. den natürlichen Zahlen mit der normalen Größer-Relation. Geben Sie für jeden Baum an, ob er ein AVL-Baum ist oder nicht. Begründen Sie Ihre Aussage jeweils kurz.



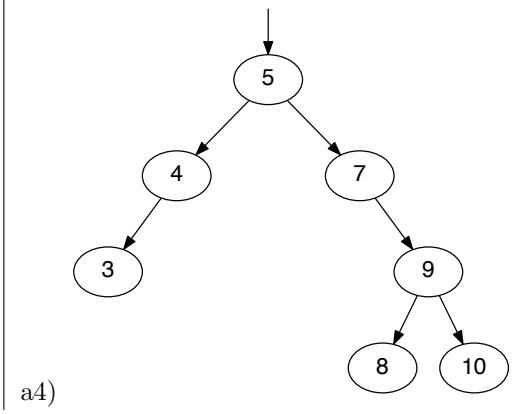
a1)



a2)



a3)



a4)

b) Betrachten Sie den AVL-Baum B auf der folgenden Seite.

- Fügen Sie den Schlüssel **38** zunächst wie in einem normalem binären Suchbaum ein. Sie können die vorhandene Graphik auf den nächsten Seite entsprechend ergänzen.
- Bestimmen Sie danach die Balance-Faktoren an allen Knoten auf dem Pfad von der Wurzel zum neuen Knoten und tragen Sie diese in der Graphik ein.
- Stellen Sie nun die AVL-Eigenschaft mit dem in der Vorlesung erläuterten Algorithmus wieder her. Zeichnen Sie das Ergebnis.

c) Wiederholen Sie Aufgabenteil b) mit dem Originalbaum B (nicht dem Ergebnis von Teil b!) und dem neuen Schlüssel **4**. Sie finden eine Kopie von Baum B für Aufgabenteil c) auf der übernächsten Seite.

Lösung

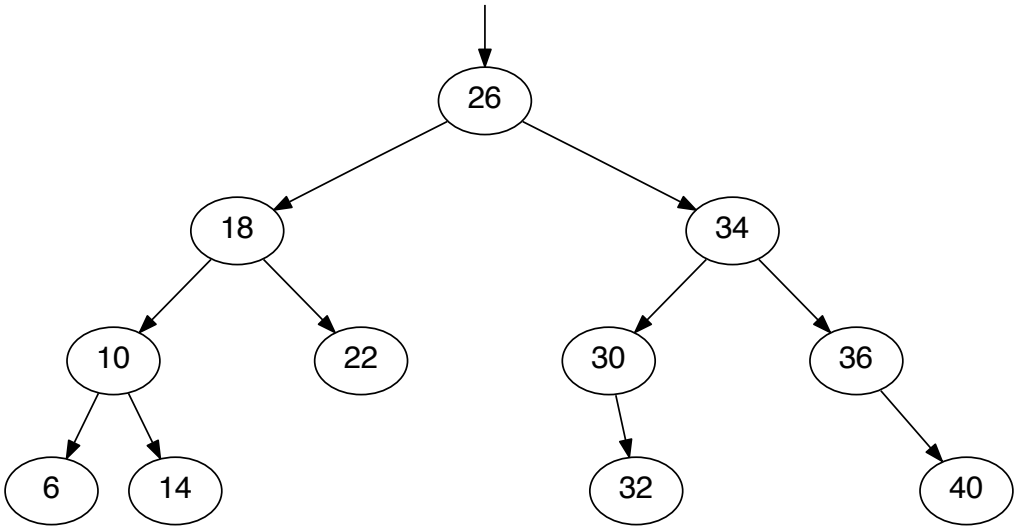
a1) Ist AVL-Baum, da BST und Betrag Höhenbalance höchstens +1.

a2) Ist AVL-Baum, da BST und Betrag Höhenbalance höchstens +1.

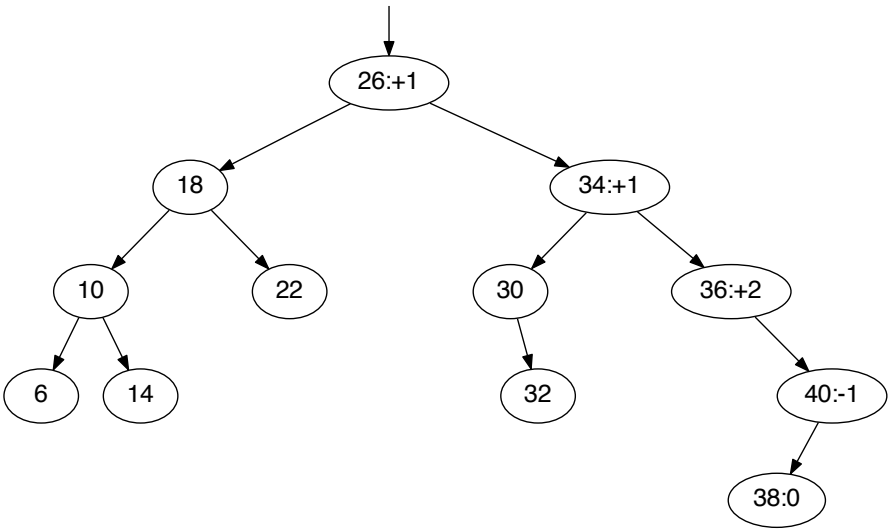
a3) Kein AVL-Baum, da kein BST (2>1, aber im linken Teilbaum)

a4) Kein AVL-Baum, da Balancce an Knoten 7 +2, damit > 1.

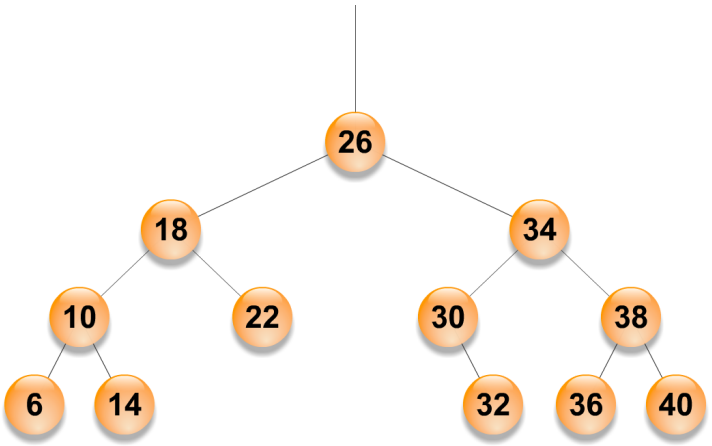
Fortsetzung
Baum B für Aufgabenteil b) (einzufügen: 38)



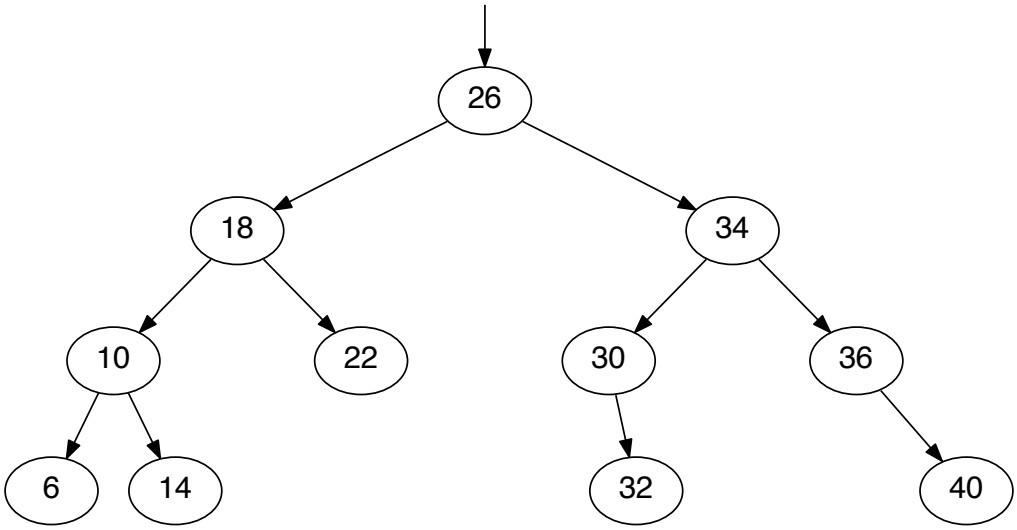
Lösung



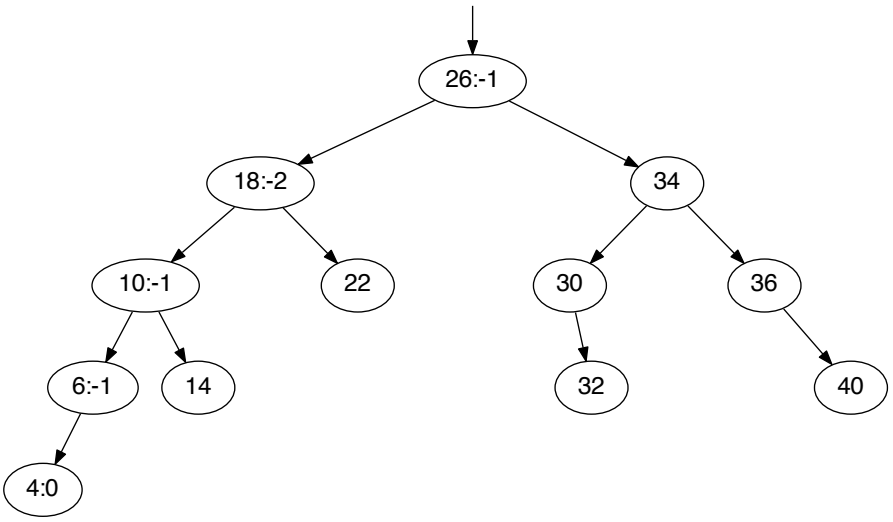
b)



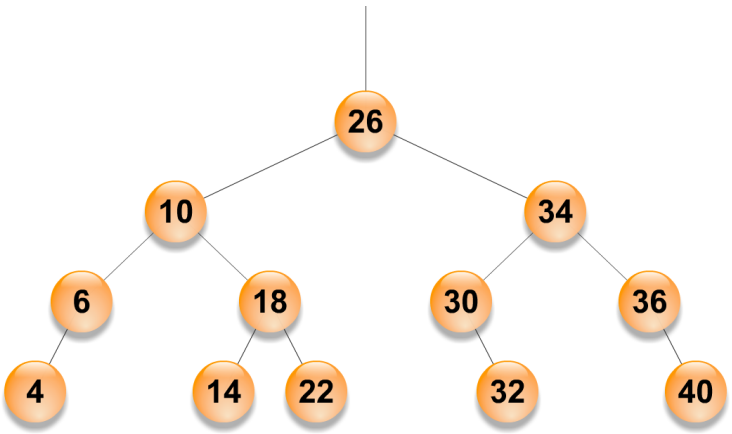
Fortsetzung
Baum B für Aufgabenteil b) (einzufügen: 4)



Lösung



c)



– Ende der Klausur –