Algorithmen und Datenstrukturen
Klausur SS 2019

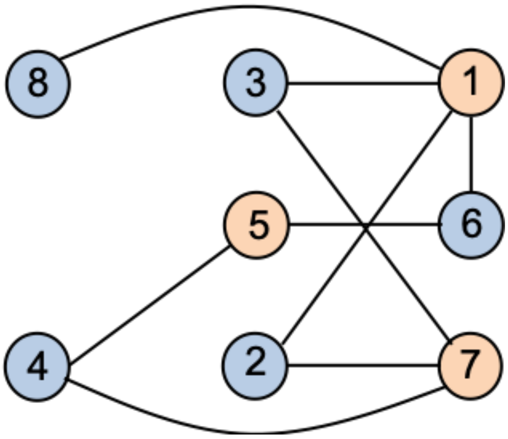
Angewandte Informatik Bachelor

Name	
Matrikelnummer	

Aufgabe 1	Tiefen- und Breitensuche in Graphen	9	
Aufgabe 2	AVL-Bäume	11	
Aufgabe 3	Rot-Schwarz-Bäume	9	
Aufgabe 4	Algorithmus von Dijkstra	11	
Aufgabe 5	Algorithmus von Floyd	9	
Aufgabe 6	Union-Find-Struktur	11	
Summe		**60**	

Aufgabe 1 Tiefen- und Breitensuche in Graphen (9 Punkte)

Gegeben ist ein ungerichteter Graph:



a) Geben Sie die Reihenfolge der besuchten Knoten an, wenn der Graph mit Tiefensuche mit Startknoten 1 traversiert wird. Betrachten Sie die Nachbarn eines Knotens in der durch die Knotennummerierung gegebenen Reihenfolge.

1, 2, 7, 3, 4, 5, 6, 8

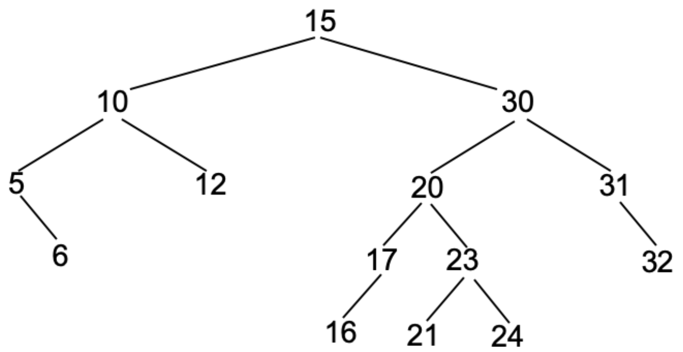
b) Geben Sie die Reihenfolge der besuchten Knoten an, wenn der Graph mit Breitensuche mit Startknoten 1 traversiert wird. Betrachten Sie die Nachbarn eines Knotens in der durch die Knotennummerierung gegebenen Reihenfolge.

1, 2, 3, 6, 8, 7, 5, 4

c) Ein Graph ist bipartit, wenn sich seine Knotenmenge disjunkt in A und B zerlegen lässt, so dass es nur Kanten zwischen A und B gibt. Prüfen Sie mittels Tiefensuche, ob der Graph bipartit ist. Falls der Graph bipartit, färben Sie die Knoten aus A und B in zwei unterschiedliche Farben ein (z.B. A = rot und B = blau).

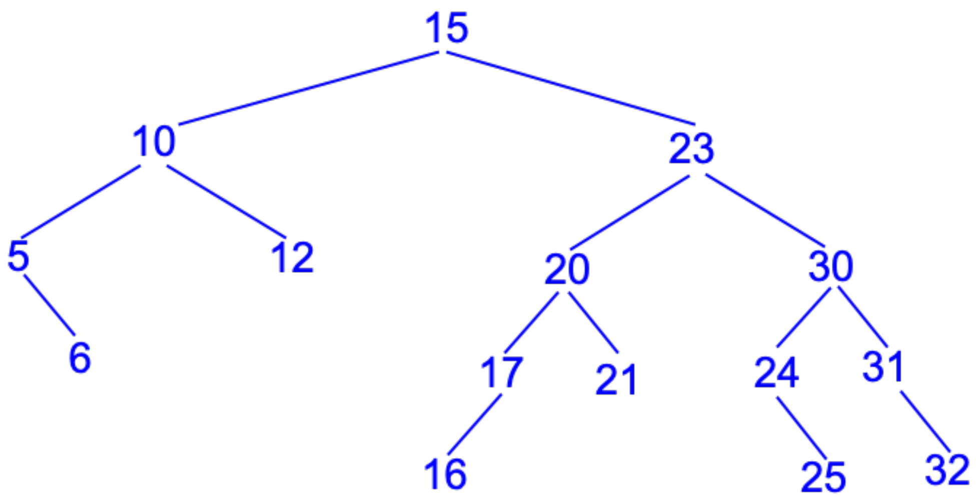
Aufgabe 2 AVL-Bäume (11 Punkte)

a) In folgendem binären Suchbaum ist der Tiefenunterschied zwischen den Blättern 16, 21, 24 und dem Blatt 12 gleich 2. Wieso ist Baum dennoch ein AVL-Baum?

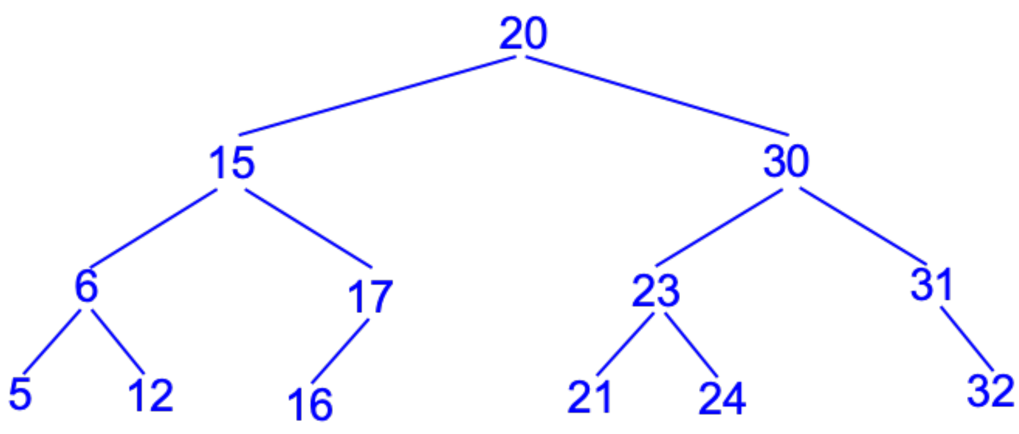


Für jeden Knoten k unterscheiden sich die Höhen der beiden Teilbäume von k um höchstens 1.

b) <u>Fügen</u> Sie im AVL-Baum <u>aus a)</u> die Zahl 25 ein.

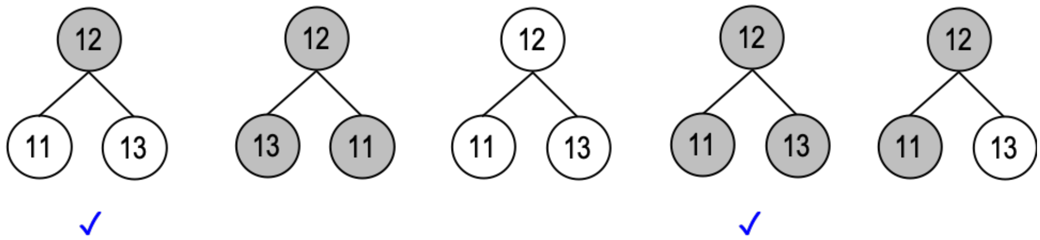


c) <u>Löschen</u> Sie im AVL-Baum <u>aus a)</u> die Zahl 10.

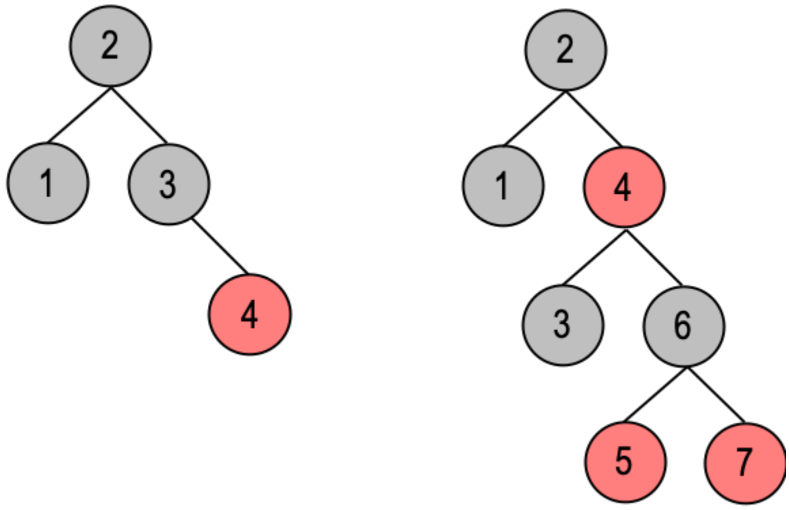


Aufgabe 3 Rot-Schwarz-Bäume (9 Punkte)

a) Welche der folgenden 5 Binärbäume sind korrekte Rot-Schwarz-Bäume (rot = weiss unterlegt, schwarz = grau unterlegt)? Kennzeichnen Sie mit einem Haken. Falsche Antworten ergeben Abzüge.



b) Welcher Rot-Schwarz-Baum entsteht, wenn die Zahlen 1, 2, 3, 4 in einem leeren Baum eingefügt werden? Welcher Rot-Schwarz-Baum ergibt sich, wenn Sie danach noch die Zahlen 5, 6, 7 einfügen?



Aufgabe 4 Algorithmus von Dijkstra (11 Punkte)

Ein gewichteter, gerichteter Graph mit der Knotenmenge V = {1, 2, 3, 4, 5, 6} ist durch folgende Adjazenzmatrix gegeben. Bestimmen Sie mit dem Algorithmus von Dijkstra vom Startknoten s = 1 zu allen anderen Knoten jeweils einen günstigsten Weg.

	1	2	3	4	5	6
1		5	1	3		5
2					3	
3		3		1		
4		1			5	2
5						
6					1	

a) Tragen Sie in folgende Tabelle nach jedem Besuchsschritt folgendes ein:

- der besuchte Knoten b
- die Kosten d[v] für den günstigsten Weg von Startknoten s nach v
- den Vorgängerknoten p[v] für den günstigsten Weg von Startknoten s nach v.

Wichtig: Haben mehrere Kandidaten denselben d-Wert, dann wird der Kandidat mit kleinster Nummer als nächster Knoten besucht.

Hinweis: Es brauchen nur die d- und p-Werte eingetragen werden, die sich geändert haben. Die endgültigen p- und d-Werte können durch Umrandung besonders gekennzeichnet werden.

b	d[1]	d[2]	d[3]	d[4]	d[5]	d[6]	p[1]	p[2]	p[3]	p[4]	p[5]	p[6]
1	0	5	1	3	∞	5	-	1	1	1	-	1
3		4		2				3		3		
4		3			7	4		4			4	4
2					6						2	
6					5						6	
5												

b) Geben Sie den gefundenen günstigsten Weg von 1 nach 5 an.

1 – 3 – 4 – 6 – 5

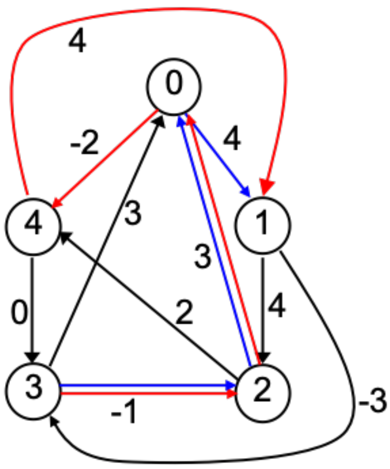
c) Welche Kosten hat der günstigste Weg von 1 nach 5?

5

Aufgabe 5 Algorithmus von Floyd (9 Punkte)

Gegeben ist ein gerichteter Graph mit der Knotenmenge V = {0, 1, 2, 3, 4}. Mit dem Algorithmus von Floyd soll für jedes Knotenpaar ein günstigster Weg berechnet werden.

Der Algorithmus von Floyd berechnet für k = 0, 1, ... 4 die Distanzmatrix D^k und die Vorgängermatrix P^k. $D^k(i, j)$ gibt die Länge eines günstigsten Wegs von i nach j an, wobei nur Wege von i nach j berücksichtigt werden, die über Knoten aus {0, 1, …, k} gehen.



$D^3[3][1] = 6$

$D^4[3][1] = 4$

a) Berechnen Sie D^4 und P^4 mit Hilfe von D^3 und P^3.

D^3

0	4	0	1	−2
−1	0	−4	−3	−3
3	7	0	2	1
2	6	−1	0	0
2	4	−1	0	0

P^3

–	0	3	1	0
2	–	3	1	0
2	0	–	2	0
2	0	3	–	0
2	4	3	4	–

D^4

0	2	−3	−2	−2
−1	0	−4	−3	−3
3	5	0	1	1
2	4	−1	0	0
2	4	−1	0	0

P^4

–	4	3	4	0
2	–	3	1	0
2	4	–	4	0
2	4	3	–	0
2	4	3	4	–

b) Zeichnen Sie im oben dargestellten Graphen den kürzesten Weg von Knoten 3 nach Knoten 1 für P^3 und für P^4 ein. Geben Sie außerdem die Längen der kürzesten Wege gemäß D^3 und D^4 an.

Aufgabe 6 Union-Find-Struktur (11 Punkte)

In einem ungerichteten Graphen G sind zwei Knoten u und v genau dann gegenseitig erreichbar, falls es einen Weg von u nach v gibt. Die gegenseitige Erreichbarkeit zweier Knoten soll mit einer Union-Find-Struktur geprüft werden. Die Union-Find-Struktur teilt die Menge V so in disjunkte Teilmengen auf, dass in einer Teilmenge alle diejenigen Knoten enthalten sind, die gegenseitig erreichbar sind. Sie können voraussetzen, dass die Menge der Knoten V = {0, 1, 2, ..., n-1} ist.

a) Skizzieren Sie ein Verfahren (in Pseudo-Code), das für einen Graphen G eine Union-Find-Struktur aufbaut. Die Funktionen union(u, v) und find(v) dürfen als gegeben angenommen werden!

```
initialisiere Union-Find-Struktur mit {{0}, {1}, ..., {n-1}};
for (alle Kanten (u,v) im Graph)
    if (find(u) ≠ find(v)
        union(find(u), find(v));
```

b) Wie kann mit der aufgebauten Union-Find-Struktur die Erreichbarkeit zweier Knoten u und v geprüft werden?

u und v sind erreichbar, falls find(u) = find(v)

c) Geben Sie für Ihre Verfahren in a) und b) mit Hilfe der O-Notation den Aufwand in Abhängigkeit von der Anzahl der Knoten |V| bzw. der Anzahl der Kanten |E| an.

a) Aufbau der Union-Find-Struktur	O(|E| log(|V|)
b) Prüfen der Erreichbarkeit	O(log(|V|)