Algorithmen und Datenstrukturen
Klausur SS 2020

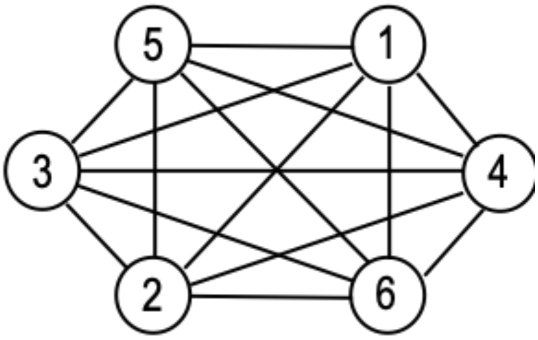
Angewandte Informatik Bachelor

Name	
Matrikelnummer	

Aufgabe 1	Tiefen- und Breitensuche in ungerichteten Graphen	12	
Aufgabe 2	B-Bäume	14	
Aufgabe 3	Algorithmus von Floyd	10	
Aufgabe 4	Minimal aufspannender Baum mit Algorithmus von Kruskal	14	
Aufgabe 5	Binäre und Binomiale Heaps	10	
Summe		**60**	

Aufgabe 1 Tiefen- und Breitensuche in Graphen (12 Punkte)

Ein vollständiger Graph ist ein ungerichteter Graph, bei dem jeder Knoten mit jedem anderen Knoten mit einer Kante verbunden ist. Folgende Abbildung zeigt einen vollständigen Graphen mit 6 Knoten.



a) Wieviel Kanten hat ein vollständiger Graph mit n Knoten allgemein? (3 Punkte)

n*(n-1)/2

b) Geben Sie für den oben abgebildeten Graphen 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. (2 Punkte)

1, 2, 3, 4, 5, 6

c) Geben Sie für den oben abgebildeten Graphen 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. (2 Punkte)

1, 2, 3, 4, 5, 6

d) Wie sieht die Reihenfolge in c) aus, wenn in dem Graphen die Kante (1,2) gelöscht wird? (2 Punkte)

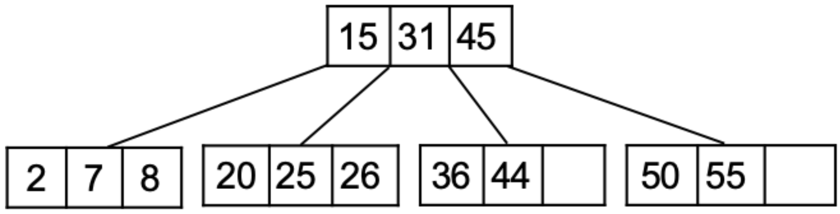
1, 3, 4, 5, 6, 2

e) Welches Problem entsteht bei der rekursiven Tiefensuche bei einem vollständigen Graphen mit sehr vielen Knoten? Zwei kurze Sätze genügen. (3 Punkte)

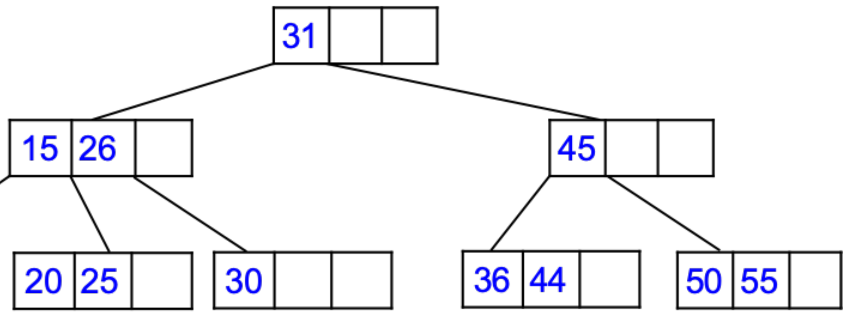
Maximale Rekursiontiefe = n-1.
Daher Stack-Overflow bei großem n.

Aufgabe 2 B-Bäume (14 Punkte)

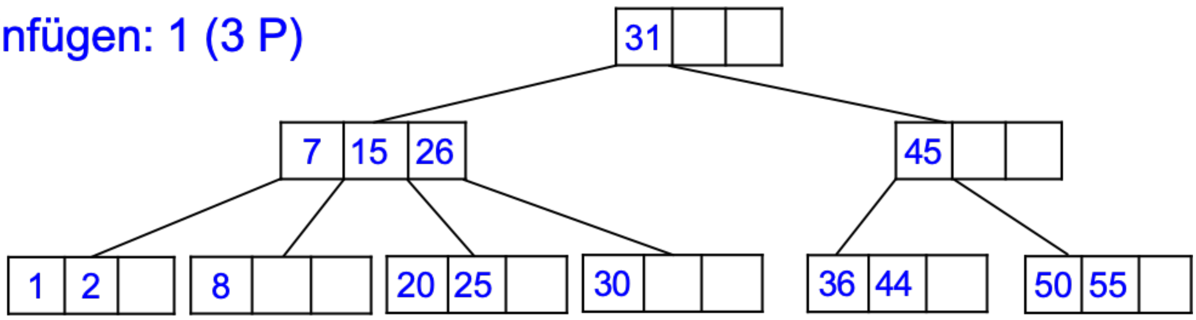
a) <u>Fügen</u> Sie in folgendem B-Baum (der Ordnung 4) die Schlüssel <u>30, 1 und 17 in genau dieser Reihenfolge ein</u>. (8 Punkte)



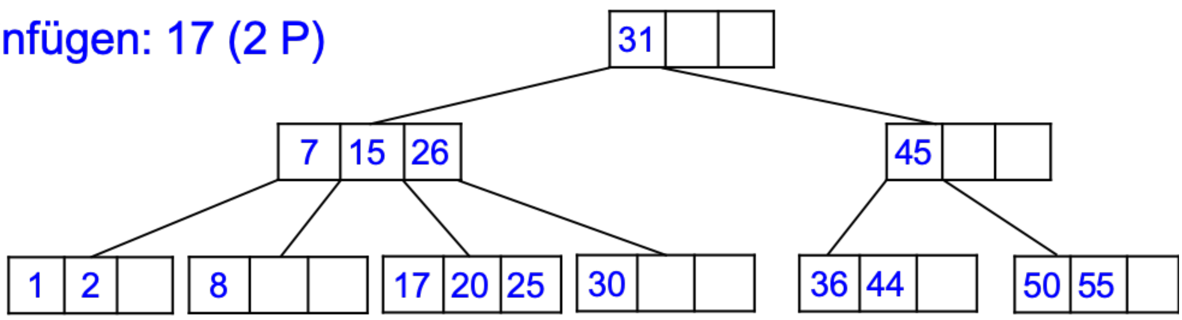
Einfügen: 30 (3 P)



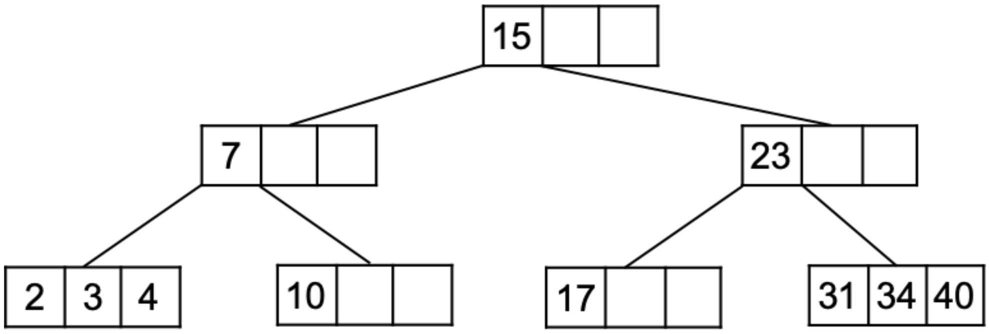
Einfügen: 1 (3 P)



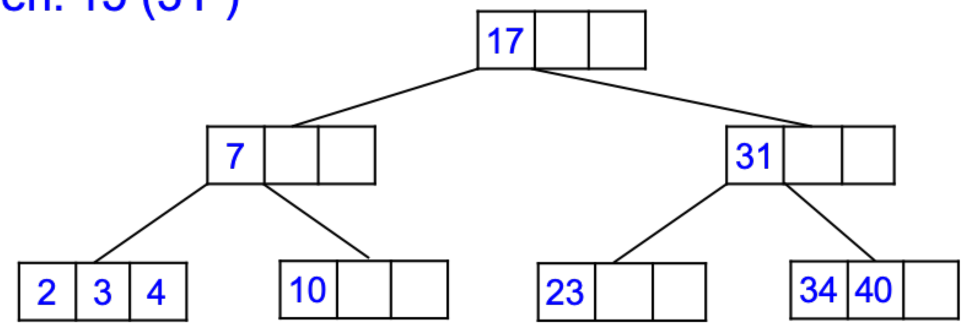
Einfügen: 17 (2 P)



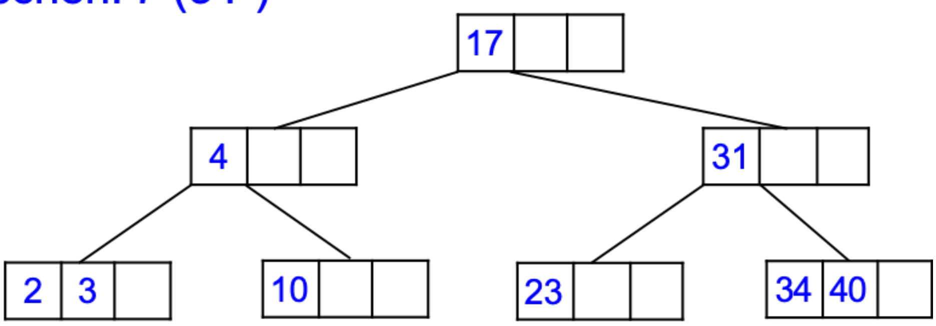
b) <u>Löschen</u> Sie in folgendem B-Baum (der Ordnung 4) den Schlüssel <u>15 und dann 7</u>. (6 Punkte)



Löschen: 15 (3 P)



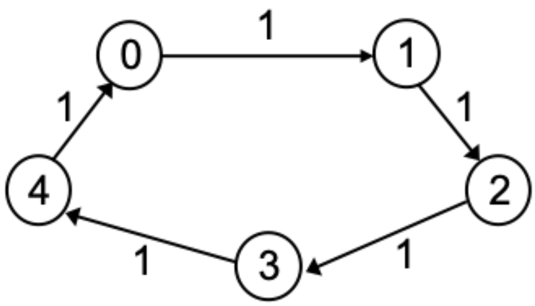
Löschen: 7 (3 P)



Aufgabe 3 Algorithmus von Floyd (10 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 kürzester Weg berechnet werden.

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



a) Geben Sie D^0 an. Zellen, deren Werte ∞ sind, dürfen einfach leer gelassen werden. (2 Punkte)

D^0

0	1			
	0	1		
		0	1	
			0	1
1				0

b) Geben Sie D^4 an, <u>ohne</u> den Algorithmus von Floyd schrittweise anzuwenden. Überlegen Sie sich stattdessen, was als Ergebnis herauskommen müsste. (4 Punkte).

D^4

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

c) Welches Problem entsteht, wenn in dem oben abgebildeten Graphen eine Kante von Knoten 2 nach Knoten 0 mit dem negativen Gewicht -3 eingefügt wird (ein bis zwei Sätze genügen)? Wie erkennt der Algorithmus von Floyd dieses Problem (ein Satz genügt)? (4 Punkte)

Es entsteht ein negativer Zyklus (0 -> 1 -> 2 -> 0) mit Weglänge = -1.
Existenz von kürzesten Wegen ist nicht gewährleistet, da negativer Zyklus mehrmals durchlaufen werden kann.

Das Diagonalelement D[0][0] wird negativ.

Aufgabe 4 Minimal aufspannende Bäume (14 Punkte)

Ein gewichteter, ungerichteter Graph mit der Knotenmenge V = {1, 2, 3, 4, 5, 6, 7, 8, 9} und 20 Kanten ist durch folgende Adjazenzmatrix gegeben. Aufgrund der besseren Lesbarkeit sind die Gewichte nur für eine Kantenrichtung eingetragen.

	1	2	3	4	5	6	7	8	9
1			1	12		6	10	4	
2			10	8	2		4	9	
3					15	5		6	
4					8		7	11	
5							3		
6							10	2	
7									
8									14
9									

a) Bestimmen Sie einen minimal aufspannenden Baum mit dem Algorithmus von Kruskal. Tragen Sie in die Tabelle auf der folgende Seite für jeden Schritt folgende Daten ein:

- die ausgewählte Kante,
- das Gewicht und
- die dazugehörende Union-Find-Struktur in graphischer Form als Menge von Bäumen.

Verwenden Sie den Union-By-Height-Algorithmus. (11 Punkte)

b) Geben Sie die Datenstruktur (Elternfeld p) für die im Schritt 7 erhaltene Union-Find-Struktur an. (2 Punkte)

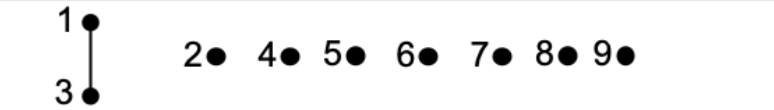
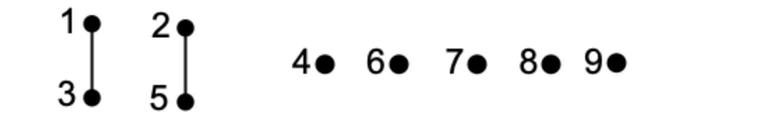
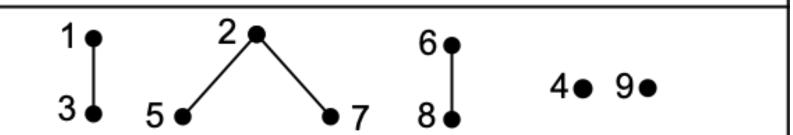
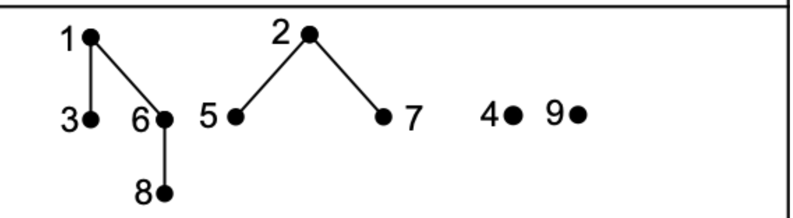
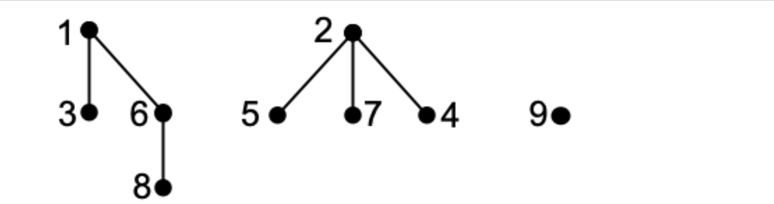
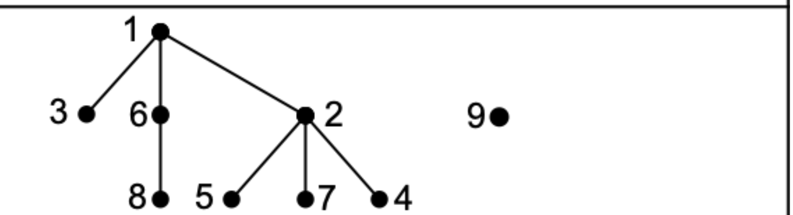
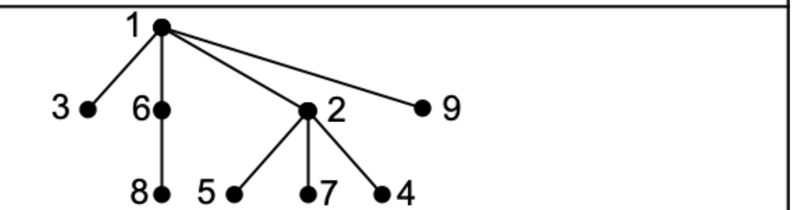
Knoten	1	2	3	4	5	6	7	8	9
p	-3	1	1	2	2	1	2	6	-1

c) Wieviel Kanten hat ein minimal aufspannender Baum für einen Graphen mit n Knoten? (1 Punkt)

n-1

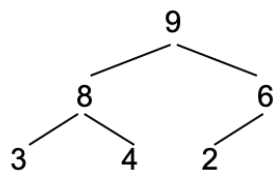
	1	2	3	4	5	6	7	8	9
1			1	12		6	10	4	
2			10	8	2		4	9	
3					15	5		6	
4					8		7	11	
5							3		
6							10	2	
7									
8									14
9									

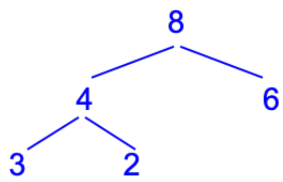
Adjazenzmatrix
(Aufgrund der besseren Lesbarkeit sind die Gewichte nur für eine Kantenrichtung eingetragen)

Schritt	Kante	Gewicht	Union-Find-Struktur
1	(1,3)	1	
2	(2,5)	2	
3	(6,8)	2	
4	(5,7)	3	
5	(1,8)	4	
6	(4,7)	7	
7	(2,8)	9	
8	(8,9)	14	

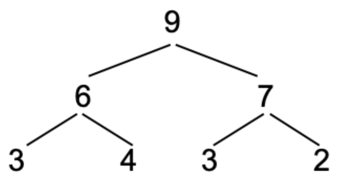
Aufgabe 5 Binäre und binomiale Heaps (10 Punkte)

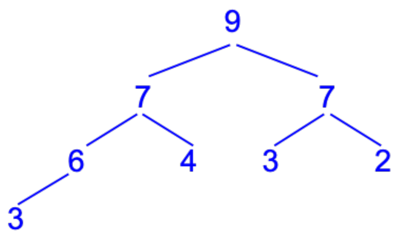
a) Löschen Sie in folgendem binären Heap die größte Zahl. (2 Punkte)



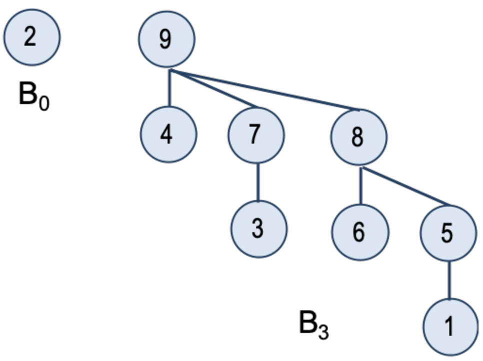


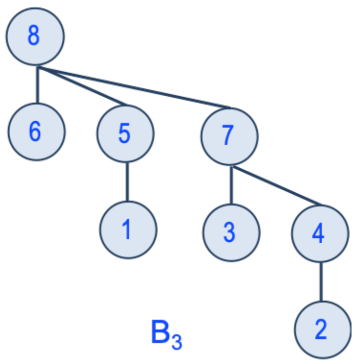
b) Fügen Sie in folgendem binären Heap die Zahl 7 ein. (2 Punkte)





c) Löschen Sie in folgendem binomialen Heap die größte Zahl. (3 Punkte)





d) Durch welche Eigenschaft eines Binomialbaums ist sein Namensbestandteil „Binomial" gerechtfertigt? (3 Punkte)

Ein Binomialbaum der Höhe n hat in der Ebene k $\binom{n}{k}$ viele Knoten.