Algorithmen und Datenstrukturen
Klausur SS 2016

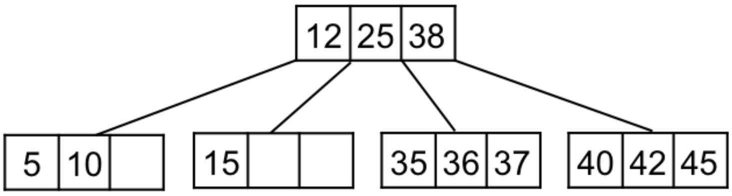
Angewandte Informatik Bachelor

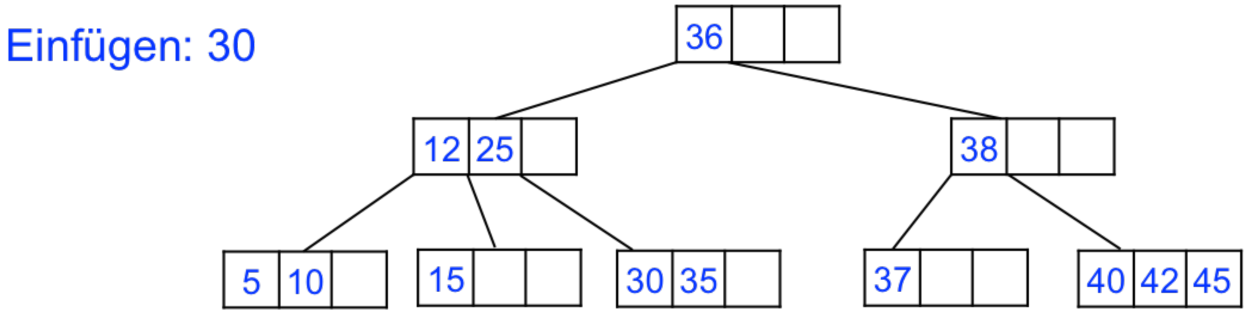
Name	
Matrikelnummer	

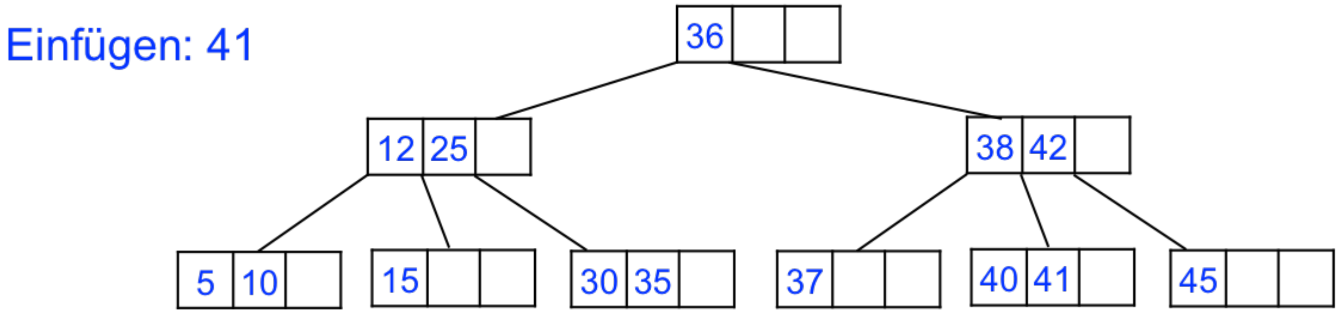
Aufgabe 1	B-Baum und Rot-Schwarz-Baum	21	
Aufgabe 2	Minimal aufspannender Baum mit Algorithmus von Kruskal	15	
Aufgabe 3	Tiefensuchbaum	12	
Aufgabe 4	Binäre und binomiale Heaps	12	
Summe		**60**	

Aufgabe 1   B-Bäume   (15 Punkte)

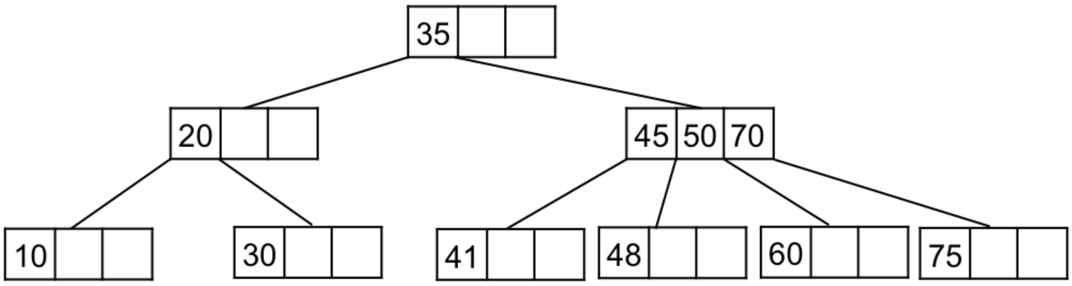
a) <u>Fügen</u> Sie in folgendem B-Baum (der Ordnung 4) die Schlüssel 30 und dann 41 ein.



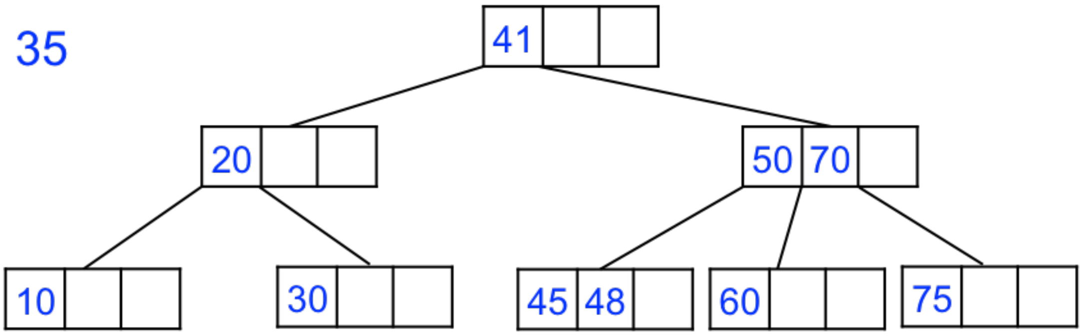




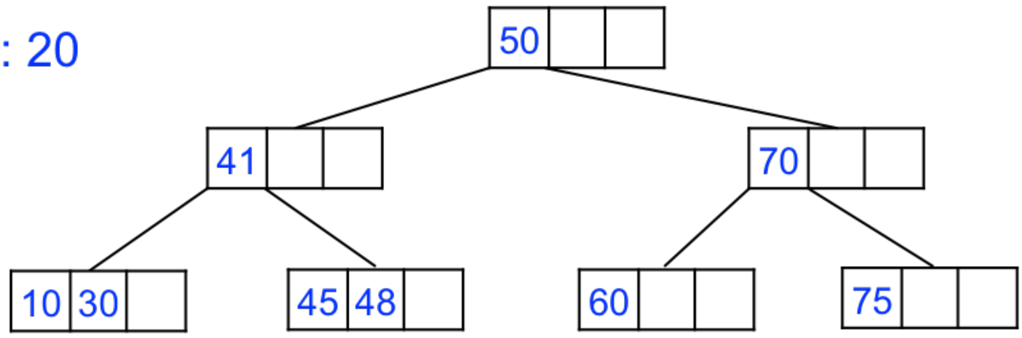
b) <u>Löschen</u> Sie in folgendem B-Baum (der Ordnung 4) die Schlüssel 35 und dann 20.



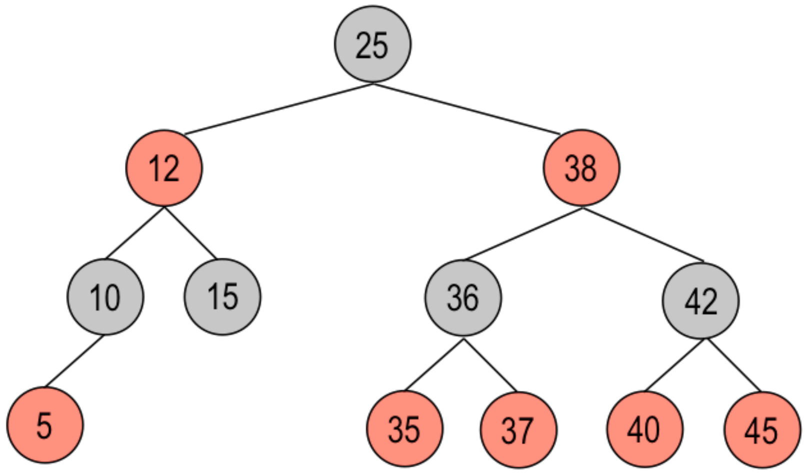
Löschen: 35



Löschen: 20

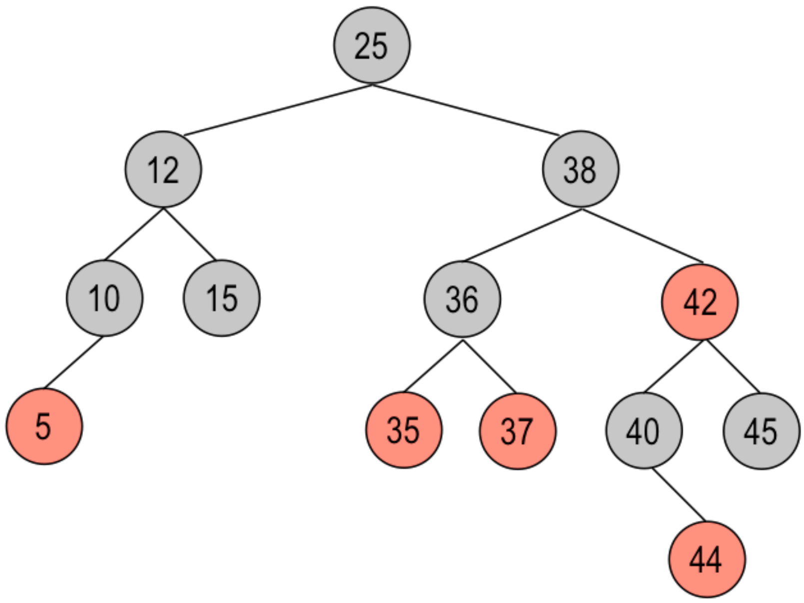


c) Geben Sie für den B-Baum aus Teilaufgabe a) den entsprechenden Rot-Schwarz-Baum an.



Lösung ist nicht eindeutig!

d) Fügen Sie in den Rot-Schwarz-Baum aus Aufgabe c) die Zahl 44 ein. Achten Sie auf korrekte Umfärbungen.



(Lösungshinweis:
44 wird wie bei einem binären Suchbaum eingefügt.
Zusätzlich müssen Umfärbungen vorgenommen werden:
zwei flip-Operationen und 44 bekommt als neuer Knoten die Farbe rot.)

Aufgabe 2 Minimal aufspannende Bäume (15 Punkte)

Ein gewichteter, ungerichteter Graph mit der Knotenmenge V = {1, 2, 3, 4, 5, 6, 7, 8, 9} und 18 Kanten ist durch folgende Adjazenzmatrix gegeben. Einfachheitshalber sind die Gewichte nur für eine Kantenrichtung eingetragen.

	1	2	3	4	5	6	7	8	9
1				6	4		8	6	
2				18		16			9
3				7	1		3	13	
4					6		12	5	17
5							2		
6									14
7								10	
8									
9									

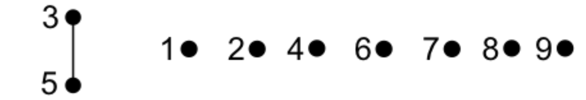
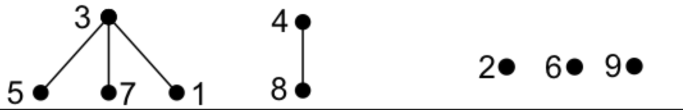
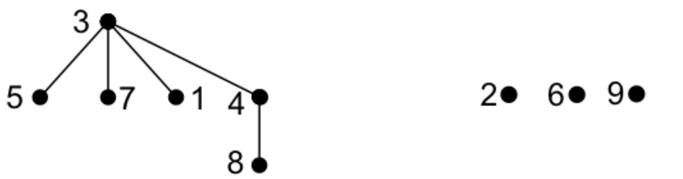
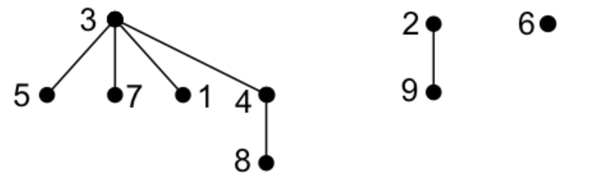
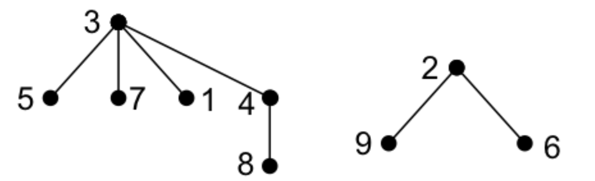
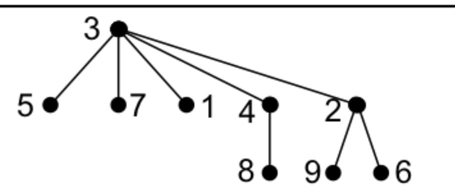
a) Bestimmen Sie einen minimal aufspannenden Baum mit dem Algorithmus von Kruskal. Tragen Sie in die Tabelle auf der folgende Seite für jeden Schritt folgende Daten ein:

- 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.

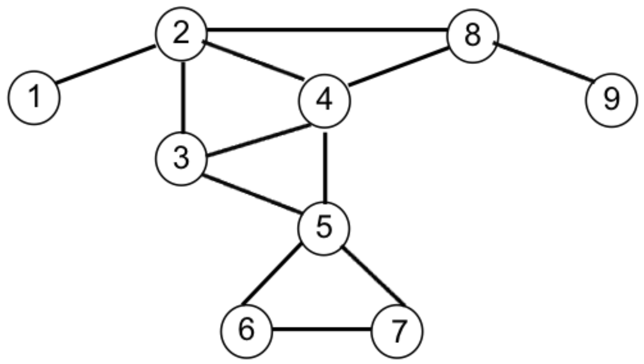
b) Geben Sie die Datenstruktur (Elternfeld p) für die im Schritt 7 erhaltene Union-Find-Struktur an.

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

Schritt	Kante	Gewicht	Union-Find-Struktur
1	(3,5)	1	
2	(5,7)	2	
3	(1,5)	4	
4	(4,8)	5	
5	(1,4)	6	
6	(2,9)	9	
7	(6,9)	14	
8	(4,9)	17	

Aufgabe 3 Tiefensuchbaum (12 Punkte)

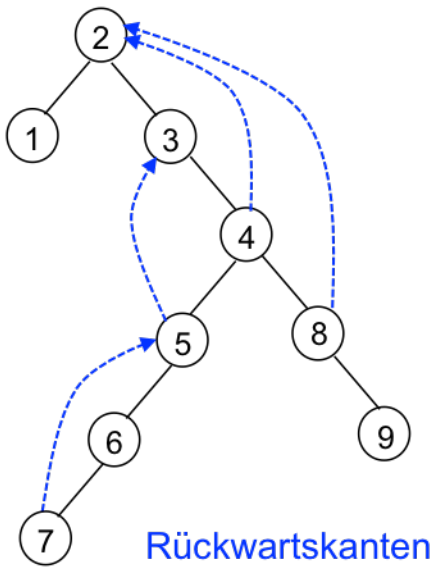
Gegeben sei folgender ungerichteter Graph:



a) Geben Sie alle Artikulationspunkte an.

Knoten 2, 5 und 8

b) Geben Sie den Tiefensuchbaum für diesen Graph mit Wurzel 2 an. Betrachten Sie die Nachbarn eines Knotens in der durch die Knotennummerierung gegebenen Reihenfolge. Berücksichtigen Sie, dass der Tiefensuchbaum auch sogenannte Rückwärtskanten enthält.

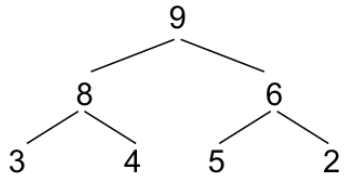


c) Sind Knoten 2 und Knoten 4 Artikulationspunkte (APe)? Begründen Sie über die Charakterisierung von Artikulationspunkten (APen) in einem Tiefensuchbaum (TSB).

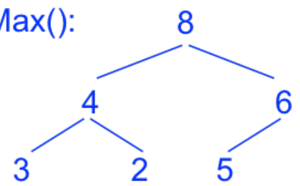
- Knoten 2 ist ein AP, weil er als Wurzel mehr als ein Kind hat.
- Knoten 4 ist kein AP, da es im TSB für alle Nachfolger von 4 (nämlich Knoten 5 und 8) einen Weg über eine Rückwärtskante zu einem Vorgänger von 4 gibt.

Aufgabe 4 *Binäre und binomiale Heaps* *(12 Punkte)*

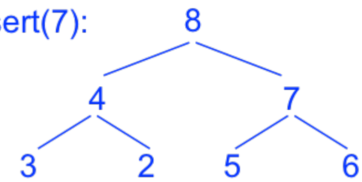
a) Löschen Sie in folgendem binären Heap zuerst die größte Zahl und fügen Sie dann die Zahl 7 ein. Der Heap ist absteigend heap-geordnet: d.h. in jeder Elternknoten ist größer oder gleich als seine Kindknoten.



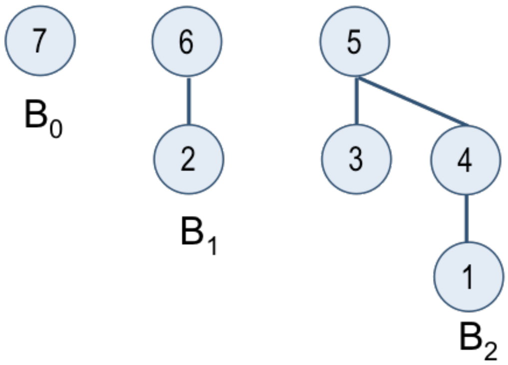
delMax():



insert(7):



b) Geben Sie eine Folge von Zahlen an, so dass deren Einfügen in einen leeren binomialen Heap den folgenden binomialen Heap ergibt. Hinweis: die Zahlenfolge ist nicht eindeutig.



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

c) Es sei heap1 ein binomialer Heap wie in b) gezeigt und heap2 eine Kopie von heap1. Wie sieht die Verschmelzung von heap1 und heap2 aus (merge-Operation)?