EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

MATH. - NATURWISS. FAKULTÄT
FACHBEREICH INFORMATIK
KOGNITIVE SYSTEME · PROF. A. ZELL

Nachklausur Algorithmen, Sommersemester 2012

Prof. A. Zell

Vorname: ..... Name: .....

Matrikelnummer: ..... Studienrichtung: .....

Semester: .....

Hinweise zur Bearbeitung

- Lassen Sie die Aufgabenblätter zusammengeheftet
- Schreiben Sie die Lösungen direkt auf die Aufgabenblätter in den dafür vorgesehenen Freiraum, notfalls mit Verweis auf die letzte Seite.
- Wenn Sie zusätzliche Blätter benötigen, muss jedes davon mit Namen und Matrikelnummer versehen werden.
- Die alleinige Angabe von Endergebnissen genügt nicht, der Lösungsweg muss erkennbar sein.
- Es sind keine Hilfsmittel (z.B. Taschenrechner) zugelassen.

Aufgabe	Maximale Punktzahl	Erreichte Punktzahl
1	12	
2	10	
3	13	
4	11	
5	8	
6	6	
7	12	
8	9	
9	9	
Gesamtpunktzahl (max. 90)		
Klausurnote		
Übungspunktzahl (max. 220)		
Übungsnote		
Gesamtnote		

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

MATH. - NATURWISS. FAKULTÄT
FACHBEREICH INFORMATIK
KOGNITIVE SYSTEME · PROF. A. ZELL

Nachklausur Algorithmen, Sommersemester 2012

Prof. A. Zell

Vorname: ..... Name: .....

Matrikelnummer: ..... Studienrichtung: .....

Semester: .....

Hinweise zur Bearbeitung

- Lassen Sie die Aufgabenblätter zusammengeheftet
- Schreiben Sie die Lösungen direkt auf die Aufgabenblätter in den dafür vorgesehenen Freiraum, notfalls mit Verweis auf die letzte Seite.
- Wenn Sie zusätzliche Blätter benötigen, muss jedes davon mit Namen und Matrikelnummer versehen werden.
- Die alleinige Angabe von Endergebnissen genügt nicht, der Lösungsweg muss erkennbar sein.
- Es sind keine Hilfsmittel (z.B. Taschenrechner) zugelassen.

Aufgabe	Maximale Punktzahl	Erreichte Punktzahl
1	12	
2	10	
3	13	
4	11	
5	8	
6	6	
7	12	
8	9	
9	9	
Gesamtpunktzahl (max. 90)		
Klausurnote		
Übungspunktzahl (max. 220)		
Übungsnote		
Gesamtnote		

Aufgabe 2: Suchen und Auswählen (10 Punkte)

(a) Mit welcher Zeitkomplexität können Sie einen beliebigen Schlüssel in einem sortierten Array der Größe n^2 suchen? (2 Punkte)

(b) Welcher der folgenden Algorithmen benötigt im schlechtesten Fall weniger Schlüsselvergleiche, wenn man 10 verschiedene Elemente in einem unsortierten Array der Größe $n = 2^k$ sucht? (4 Punkte)

(i) Es wird 10-mal linear gesucht.

(ii) Das Feld wird mit Heapsort sortiert. Anschließend verwendet man zum Suchen 10-mal die Binärsuche. Nehmen Sie an, dass das gesamte Array in den Hauptspeicher passt und Heapsort $n \cdot \log n$ Schlüsselvergleiche benötigt.

Geben Sie die Zeitkomplexität für beide Varianten an. Wovon hängt die Antwort ab?

(c) Könnte man im Algorithmus zur Bestimmung des k-kleinsten Schlüssels von n Elementen auch jeweils den Median von 3 Schlüsseln bestimmen und von diesen Medianen den Median? Wäre der Algorithmus weiterhin korrekt? Wäre die Zeitkomplexität weiterhin $\mathcal{O}(n)$? Begründen Sie! (4 Punkte)

Aufgabe 3: Sortieren (13 Punkte)

(a) Heapsort:

(i) Stellen Sie das Feld $A = [1, 2, 3, 4, 5, 6, 7, 8]$ als Min-Heap dar. Zeichnen Sie den Min-Heap als Binärbaum. (2 Punkte)

(ii) Zeichnen Sie die Zwischenstufen und die Endform des Heaps nach Löschen des Wertes 1. (3 Punkte)

(iii) Beweisen oder widerlegen Sie die folgende Aussage: Wird ein Min-Heap in einem Feld A gespeichert, so ist die Reihenfolge der Schlüssel im Feld A aufsteigend sortiert. (2 Punkte)

(b) Mergesort und Radixsort:

(i) Wie viele Mischphasen benötigt Mergesort zum Sortieren von 100 Elementen, nachdem diese anfangs auf zwei Folgen mit je 50 Elementen verteilt wurden. (2 Punkte)

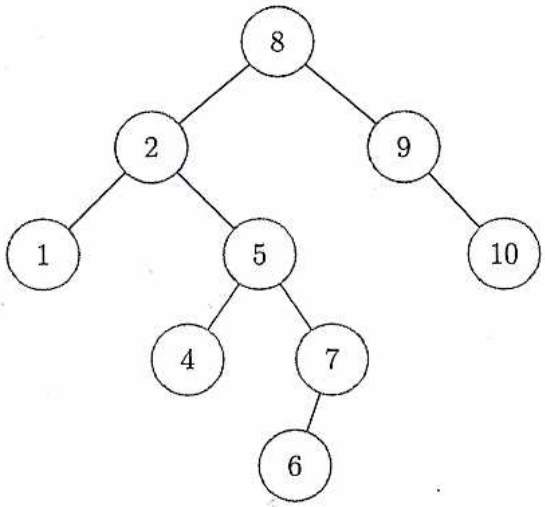
(ii) Sortieren Sie folgende Basen-Tripletts nach dem Verfahren Radixsort aus der Vorlesung (4 Punkte): atc, ctg, ctc, ccg, ttc, ctg, aaa, ctg.

Aufgabe 4: Binäre Suchbäume (11 Punkte)

(a) Gegeben sei der im Folgenden dargestellte unbalancierte Suchbaum.

(i) Fügen sie den Knoten 3 in den gegebenen binären Suchbaum ein. (2 Punkte)

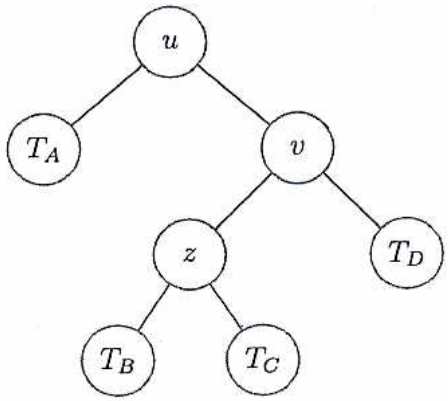
(ii) Löschen Sie aus dem Originalbaum(!) die Wurzel und zeichnen Sie den resultierenden Suchbaum nach der Reparatur rechts neben den Originalbaum. (2 Punkte)



(b) Beweisen Sie formal: Ein binärer Suchbaum mit Höhe $h > 0$ hat mindestens 1 Blatt und höchstens 2^h Blätter. (3 Punkte)

(c) Geben Sie den Java-Code an, mit dem bei einem AVL-Baum eine Doppelrotation nach links ausgeführt werden kann. Hierbei steht u für den lokalen Wurzelknoten, an dem die Rotation beginnt, und v für dessen existierenden rechten Sohn. Außerdem steht z für den existierenden linken Sohn von v. Alle drei Referenzen sind Objekte einer Klasse, die das Interface `TreeNode` implementiert. Verwenden Sie dessen vordefinierte Funktionen. Sie können davon ausgehen, dass u nicht die Wurzel des gesamten Baums ist und die Teilbäume T_A, T_B, T_C und T_D nicht leer sind. (4 Punkte)

```
interface TreeNode{
//Setzt das linke Kind
public void setLeftChild(TreeNode node);

//Setzt das rechte Kind
public void setRightChild(TreeNode node);

//Setzt den Elternknoten
public void setParent(TreeNode node);

//Gibt den Elternknoten aus
public TreeNode getParent();

//Gibt das linke Kind aus
public TreeNode getLeftChild();

//Gibt das rechte Kind aus
public TreeNode getRightChild();
}

// Fuehre doppelte Linksrotation durch
public static void doubleRotateLeft(
        TreeNode u, TreeNode v, TreeNode z){
// Ihr Code
```

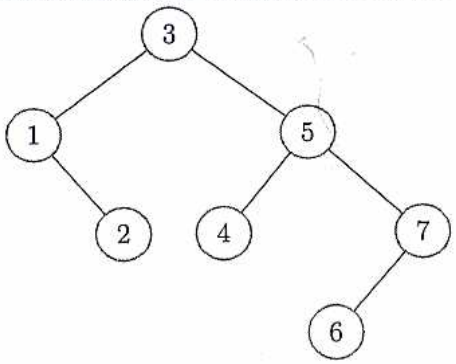



```
}
```

Aufgabe 5: AVL-Bäume, BB[α]-Bäume und B-Bäume (8 Punkte)

(a) Geben Sie einen BB[1/3]-Baum an, der kein AVL-Baum ist. (3 Punkte)

(b) Geben Sie die Wurzelbalancen für alle Knoten des folgenden BB[α]-Baums an. (2 Punkte)



$x =$	1	2	3	4	5	6	7
$\rho_x =$							

(c) Sei T ein B-Baum der Ordnung k mit Höhe h und n Blättern. Jeder innere Knoten außer der Wurzel hat dabei mindestens k und höchstens $2k - 1$ Kinder.
Beweisen Sie: $n \geq (2k - 1)^h$ (3 Punkte)

Aufgabe 6: Hashing (6 Punkte)

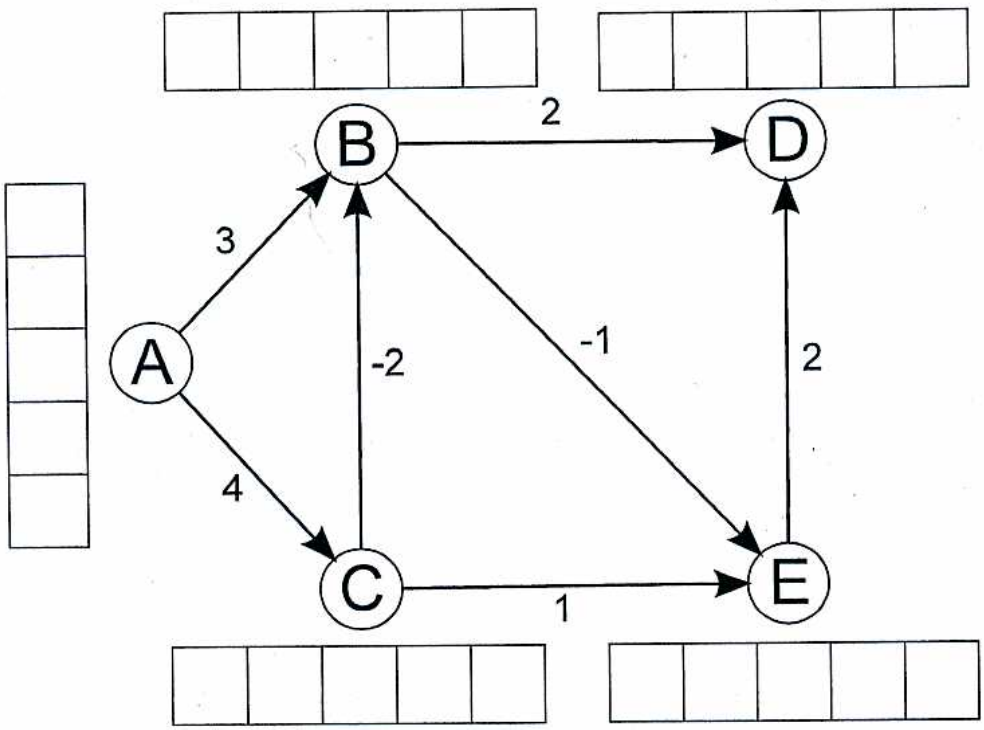
(a) Erläutern Sie die Idee des perfekten Hashings. (2 Punkte)

(b) Finden Sie für die Menge $S = \{3, 13, 23, 33, 44, 55\}$ eine injektive Hashfunktion des Typs $h_1(x) = ((x \mod N) \mod m)$, wobei $0 \leq m = 15 < N \leq 20$ gilt. Beweisen Sie die Injektivität. (4 Punkte)

Aufgabe 7: Graphen: Topologische Sortierung und Dijkstra (12 Punkte)

(a) Zeichnen Sie einen gerichteten Graphen $G = (V, E)$ mit 5 Knoten $V = \{A, B, C, D, E\}$ und einer von Ihnen wählbaren Zahl von Kanten, der keine topologische Sortierung erlaubt. (3 Punkte)

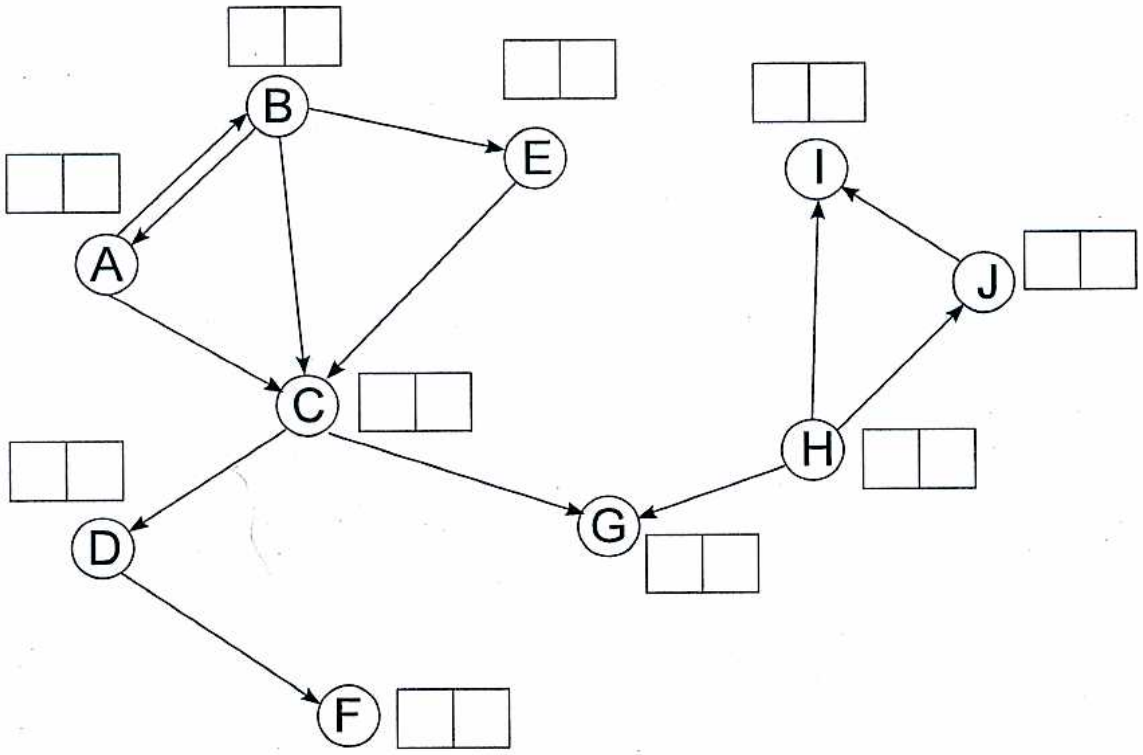
(b) Wenden Sie den Algorithmus von Bellman-Ford auf dem folgenden Graphen $G = (V, E)$ an. Bestimmen Sie dabei für jeden Knoten $v \in V$ die berechneten Distanzen für $i = 1, 2, 3, 4$ (6 Punkte).



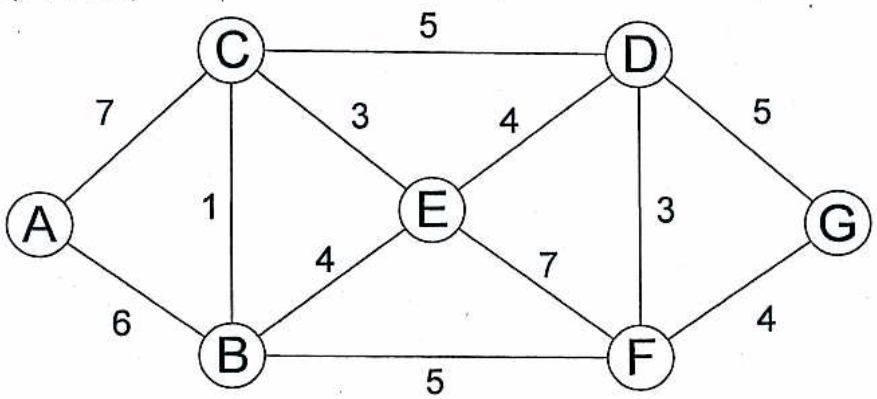
(c) Zeichnen Sie einen gerichteten Graphen $G = (V, E)$ mit 5 Knoten $V = \{A, B, C, D, E\}$ und einer von Ihnen wählbaren Zahl von Kanten, für den es zwar einen Weg von A nach D gibt, aber keinen kürzesten Weg. (3 Punkte)

Aufgabe 8: Graphen: Tiefensuche und minimal aufspannende Bäume (9 Punkte)

(a) Bestimmen Sie bei folgendem Graphen mit dem Startknoten A für jeden Knoten v die Parameter `dfsnum(v)` und `compnum(v)`. Diese geben für alle Knoten die Reihenfolge des Aufrufs bzw. des Abschlusses der Rekursion bei der Tiefensuche an. Betrachten Sie bei mehreren Alternativen immer zuerst den Knoten, der im Alphabet früher auftritt. Zeichnen Sie die Baumkanten fett und markieren Sie die Querkanten (C), Vorwärtskanten (F) und Rückwärtskanten (R). (6 Punkte)



(b) Bestimmen Sie mit dem Algorithmus von Prim einen minimal aufspannenden Baum T für den folgenden ungerichteten Graphen. Beginnen Sie mit dem Knoten A. Zeichnen Sie die entsprechenden Kanten fett. (3 Punkte)



Aufgabe 9: Zeichenketten (9 Punkte)

(a) Bestimmen Sie die Präfix-Funktion für den KMP-Algorithmus für folgendes Muster P: (3 Punkte)

P	a	a	c	a	a	a	e	a	c	a	a	c
i	1	2	3	4	5	6	7	8	9	10	11	12
$\pi(i)$												

(b) Führen Sie für das in der folgenden Tabelle dargestellte Beispiel den Algorithmus von Knuth-Morris-Pratt (aus der Vorlesung) durch. Geben Sie zusätzlich die Anzahl an Zeichenvergleichen an. (4 Punkte)

Text	a	b	d	a	c	a	b	c	d	e	b	c	a
Pattern	a	c	b	a	c								

(c) Warum ist der Algorithmus von Boyer-Moore dem Algorithmus von Knuth-Morris-Pratt vorzuziehen, wenn Muster und Textstring über einem größeren Alphabet Σ definiert sind und dessen Buchstaben wie im lateinischen Alphabet in stark unterschiedlicher Häufigkeit im Text vorkommen? (2 Punkte)