Lehrstuhl Praktische Informatik III
Universität Mannheim
Prof. Dr. Guido Moerkotte
Thorsten Fiebig

September 1999

Vordiplomklausur

Datenstrukturen und Programmierverfahren

Sommersemester 1999

Name: .....
Vorname: .....
Matrikelnummer: .....
Studienfach: .....

Wichtige Hinweise:

1. Prüfen Sie Ihr Klausurexemplar auf Vollständigkeit.
2. Alle 8 Aufgaben sind zu bearbeiten.
3. Es sind keine Hilfsmittel zugelassen.
4. Die Klausur dauert 80 Minuten.
5. Das Deckblatt sowie alle Aufgabenblätter und Lösungsblätter sind abzugeben.

	maximale Anzahl Punkte	erreichte Anzahl Punkte
Aufgabe 1	22	
Aufgabe 2	9	
Aufgabe 3	11	
Aufgabe 4	5	
Aufgabe 5	5	
Aufgabe 6	7	
Aufgabe 7	7	
Aufgabe 8	9	
	75	

1. (11 + 11 Punkte)

   In der Vorlesung wurden Operationen auf einer Menge S vorgestellt, die Elemente enthält, auf deren Schlüssel eine totale Ordnung definiert ist.

Search(S,k):	sucht das Element mit dem Schlüssel k in der Menge S
Insert(S,x):	fügt das Element x in S ein
Delete(S,x):	löscht das Element x aus S
Minimum(S):	liefert das Element mit dem minimalen Schlüssel aus S
Maximum(S):	liefert das Element mit dem maximalen Schlüssel aus S
Successor(S,x):	liefert zu x das Element mit dem nächstgrößeren Schlüssel
Predecessor(S,x):	liefert zu x das Element mit dem nächstkleineren Schlüssel

   (Beachten Sie, daß x einen Verweis auf ein Element darstellt und nicht den Schlüssel eines Elements!)

   Außerdem wurden verschiedene Möglichkeiten zur Mengenimplementierung vorgestellt. Die Laufzeit einer Mengenoperation ist abhängig von der Mengenimplementierung. In den folgenden Tabellen sollen nun in Abhängigkeit von den Mengenimplementierungen die Laufzeiten der Mengenoperationen eingetragen werden. Verwenden Sie die Θ-Notation. Gehen Sie davon aus, daß $|S| = n$ ist.

   (a) (11 Punkte)

   Tragen Sie in folgende Tabelle die Best-case-Laufzeiten ein.

	Hash-Tabelle der Größe m mit Kollisionslisten	unsortierte einfach verkette Liste	Rot-Schwarzbaum	Binary Heap
Search(S,k)				
Insert(S,x)				
Delete(S,x)				
Minimum(S)				
Maximum(S)				
Successor(S,x)				
Predecessor(S,x)				

(b) (11 Punkte)
Tragen Sie in folgende Tabelle die Worst-case-Laufzeiten ein.

	Hash-Tabelle der Größe m mit Kollisionslisten	unsortierte einfach verkette Liste	Rot-Schwarzbaum	Binary Heap
Search(S,k)				
Insert(S,x)				
Delete(S,x)				
Minimum(S)				
Maximum(S)				
Successor(S,x)				
Predecessor(S,x)				

2. (3 + 3 + 3 Punkte)

Zeigen oder widerlegen Sie folgende Aussagen:

(a) (3 Punkte)
$2^{n+1} = O(2^n)$

(b) (3 Punkte)
$f(n) = O(g(n)) \Rightarrow 2^{f(n)} = O(2^{g(n)})$

(c) (3 Punkte)
$f(n) = o(f(n/2))$

---

3. (3 + 3 + 5 Punkte)

Bestimmen Sie eine asymptotische Lösung (Θ) für folgende Rekurrenzen:

(a) (3 Punkte)
$T(n) = 3T(n/2) + n \lg n$

(b) (3 Punkte)
$T(n) = T(2n/3) + n$

(c) (5 Punkte)
$T(n) = 2T(n/2) + n \lg n$

---

4. (2 + 3 Punkte)

Sei zu $n \in N$ der ungerichtete Graph $G_n = (V_n, E_n)$ definiert durch: $V_n = \{1, \ldots, n\}$ und $E_n = \{(i, j) | i, j \in V_n \text{ und } i + j \text{ ungerade}\}$

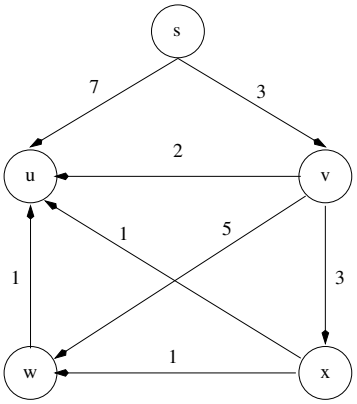
(a) (2 Punkte)
Stellen Sie die Graphen G_3 und G_4 graphisch dar.

(b) (3 Punkte)
Beantworten Sie für beliebige $n \in N$ folgende Fragen. Begründen Sie jeweils kurz Ihre Aussage.

i. Ist G_n vollständig?
ii. Ist G_n zusammenhängend?
iii. Ist G_n bipartit?

5. (5 Punkte)

Es sei folgender Graph gegeben:



Führen Sie ausgehend vom Knoten s den Dijkstra-Algorithmus auf dem Graphen aus. Geben Sie nach jeder Iteration der *while*-Schleife den d- und π-Wert für jeden Knoten sowie die Menge S an.

---

6. (7 Punkte)

Sei $G = (V, E)$ ein ungerichteter gewichteter Graph und T ein minimaler Spannbaum zu G. Sei $e = (u, v)$ eine Kante mit $u, v \in V$ und dem Gewicht w_e. Geben Sie einen Algorithmus an, der in $O(|V|)$ einen minimalen Spannbaum zu $G' = (V, E \cup \{e\})$ unter Berücksichtigung von T bestimmt. Geben Sie den Algorithmus in Pseudo-Code an.

---

7. (7 Punkte)

Sei $A[1, \ldots, n]$ ein Array mit natürlichen Zahlen. Geben Sie einen iterativen Algorithmus an, der mit maximal $3n/2$ Elementvergleichen das Maximum und das Minimum in A bestimmt.

---

8. (5 + 4 Punkte)

Zu einer gegebenen Menge von reellen Zahlen $R = \{r_1, r_2, \ldots, r_n\}$ soll eine minimale Menge von geschlossenen Intervallen I mit der Länge eins bestimmt werden, so daß für alle $r \in R$ ein $[i, j] \in I$ existiert mit $r \in [i, j]$.

(a) (5 Punkte)
Geben Sie einen Algorithmus in Pseudo-Code an, der zu einer gegebenen Menge R die Menge I bestimmt. Geben Sie die Laufzeit Ihres Algorithmus mit Hilfe der Θ-Notation an.

(b) (4 Punkte)
Geben Sie einen Beweis für die Korrektheit Ihres Algorithmus an.