Lehrstuhl für Praktische Informatik III
UNIVERSITÄT MANNHEIM
Prof. Dr. Guido Moerkotte
B6, 29, Raum C0.10
68131 Mannheim
Telefon: (0621) 181–2582
Email: moer@pi3.informatik.uni-mannheim.de

Norman May
Email: norman@pi3.informatik.uni-mannheim.de

Matthias Brantner
Email: msb@pi3.informatik.uni-mannheim.de

Algorithmen und Datenstrukturen — Vordiplomsklausur
Sommersemester 2005 — 23. September 2005

Name: ....

Vorname: ....

Matrikelnummer: ....

Studienfach: ....

Hinweise:

1. Prüfen Sie Ihr Klausurexemplar auf Vollständigkeit (17 Seiten).
2. Alle 9 Aufgaben sind zu bearbeiten.
3. Es sind keine Hilfsmittel zugelassen.
4. Die Klausur dauert 100 Minuten.
5. Tragen Sie (soweit möglich) Ihre Lösungen in den dafür vorgesehen Platz unter der jeweiligen Aufgabenstellung ein.
6. Vermerken Sie auf jedem Lösungsblatt Ihren Namen und Ihre Matrikelnummer.
7. Unterschreiben Sie die Klausur auf dem letzten Blatt.

	maximale Anzahl Punkte	erreichte Anzahl Punkte
Aufgabe 1	20	
Aufgabe 2	9	
Aufgabe 3	8	
Aufgabe 4	10	
Aufgabe 5	6	
Aufgabe 6	10	
Aufgabe 7	15	
Aufgabe 8	8	
Aufgabe 9	14	
	100	

Aufgabe 1	20 Punkte

Lösen Sie folgende Rekurrenzen. Geben Sie jeweils die Komplexität in der Θ-Notation an. Gehen Sie davon aus, daß $T(c) = \Theta(1)$ für eine kleine Konstante $c > 0$.

Aufgabe 1 a) 5 Punkte

$T(n) = 3T(n/4) + \sqrt{n}$

Aufgabe 1 b) 5 Punkte

$T(n) = \frac{1}{2} \cdot T(n/2) + \frac{1}{n}$

Aufgabe 1 c) 5 Punkte

$T(n) = 4T(n/2) + n^3$

Aufgabe 1 d) 5 Punkte

$T(n) = T(n/2) + n^2$

Aufgabe 2 — 9 Punkte

Gegeben sei ein Min-Heap `A` mit `n` Elementen.

Aufgabe 2 a) — 7 Punkte

Schreiben Sie eine Funktion mit der Signatur `HEAP-DECREASE-KEY(A,i,k)`. Die Funktion soll in einen bestehenden Heap den Schlüssel an der Stelle `i` des Arrays `A` durch den neuen, kleineren Schlüssel `k` ersetzen. Die Laufzeit soll $O(\lg n)$ betragen.

Aufgabe 2 b) — 2 Punkte

Begründen Sie, warum Ihr Algorithmus die geforderte Laufzeit einhält.

Aufgabe 3 — 8 Punkte

Entscheiden Sie für jedes der folgenden Szenarien, welche Hauptspeicherdatenstruktur in dem entsprechenden Fall am besten eingesetzt werden kann. Begründen Sie in jedem Fall ihre Entscheidung mit den charakteristischen Eigenschaften der gewählten Datenstruktur.

Aufgabe 3 a) — 2 Punkte

Eine Menge von k Fließkommazahlen soll mit randomisiertem Quicksort sortiert werden. In welcher Datenstruktur sollten die k Zahlen vorliegen?

Aufgabe 3 b) — 2 Punkte

Zum Suchen in einer Menge M von annähernd gleichverteilten Zahlen soll eine effiziente `Lookup`-Methode unterstützt werden. Die maximale Anzahl der Elemente in der Menge ($|M|$) ist bekannt. Die Elemente werden schrittweise in die Datenstruktur eingefügt. Löschenoperationen werden keine ausgeführt.

Aufgabe 3 c) 2 Punkte

Eine Menge M von Zahlen soll sortiert verwaltet werden. Die Anzahl der zu verwaltenden Zahlen ist im voraus nicht bekannt. Einfüge- und Löschoperationen können in beliebiger Kombination vorkommen. Die Suche, das Einfügen und das Löschen von Elementen soll effizient unterstützt werden.

Aufgabe 3 d) 2 Punkte

In einem ungerichteten Graphen sollen Zusammenhangskomponenten gesucht werden. Welche Datenstruktur bietet sich für die Darstellung der Zusammenhangskomponenten in einem solchen Algorithmus an?

Aufgabe 4	10 Punkte

Gegeben sei das Muster $P = abaaba$ und der Text $T = ababaabaaaba$. Führen Sie für P und T die Vorverarbeitung mit dem `Z-Algorithmus` aus. Tragen Sie für jede Stelle i den Wert Z_i in die folgende Tabelle ein. Leiten Sie aus den Werten von Z_i her, wo im Text das Muster P vorkommt.

i	1	2	3	4	5	6	-	7	8	9	10	11	12	13	14	15	16	17	18
$S[i]$	a	b	a	a	b	a	x	a	b	a	b	a	a	b	a	a	a	b	a
Z_i																			

Gegeben sei folgender gerichteter Graph $G(V, E)$ mit Wurzel A.

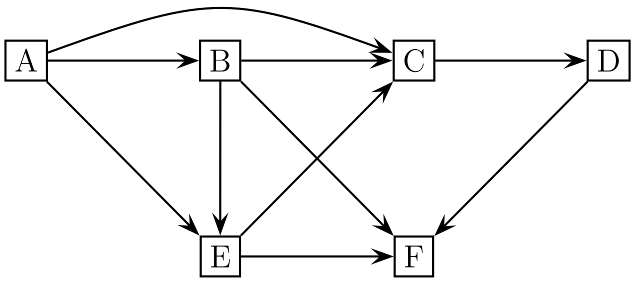


Abbildung 1: Graph G

Führen Sie mit einem der beiden in der Vorlesung vorgestellten Verfahren eine topologische Sortierung auf dem Graphen durch. Geben Sie alle Zwischenergebnisse der topologischen Sortierung an bei dem von ihnen gewählten Verfahren an. Wenn mehrere Möglichkeiten zur Anordnung von Knoten bestehen, sortieren Sie die Knoten alphabetisch.

Ein Intervall-Baum ist ein Rot-Schwarz Baum, der eine beliebige Menge von Intervallen verwaltet. Jeder Knoten x enthält ein geschlossenes Intervall $int[x]$. Ein geschlossenes Intervall ist ein geordnetes Paar $[t_1, t_2]$ zweier reeller Zahlen t_1 ($low[int[x]]$) und t_2 ($high[int[x]]$) mit $t_1 \leq t_2$. Das Intervall entspricht der Menge $\{t \in \Re : t_1 \leq t \leq t_2\}$.

Der Schlüssel eines Knotens x ist der Anfangswert $low[int[x]]$ des Intervalls. Weiterhin enthält jeder Knoten x einen Wert $max[x]$, der dem größten Wert aller Intervalle im Teilbaum mit Wurzel x entspricht.

Gegeben Sei der Intervall Baum aus Abbildung 2. Die Knoten mit schwarzem Hintergrund besitzen die Eigenschaft "schwarz", die weißen Knoten die Eigenschaft "rot".

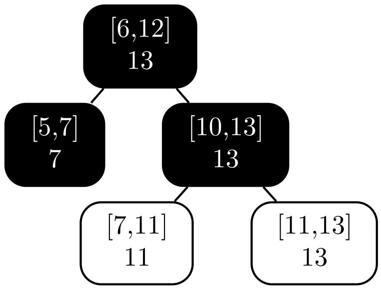
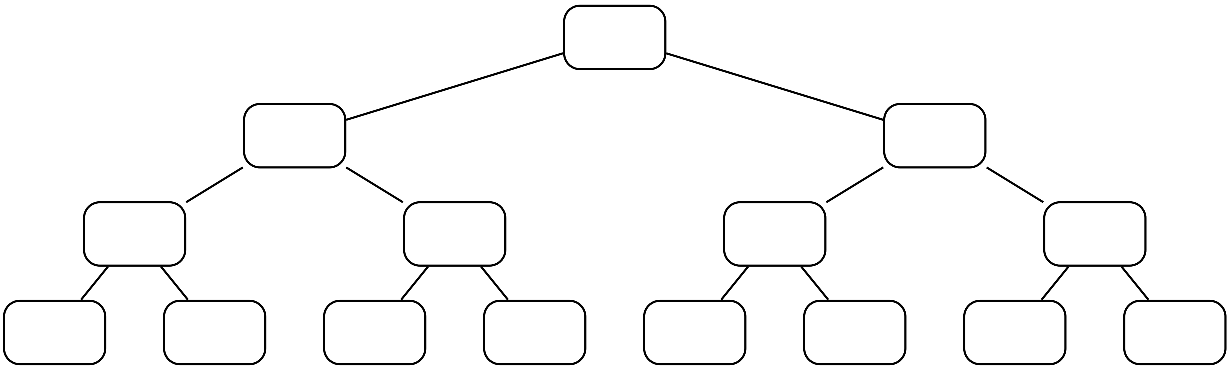


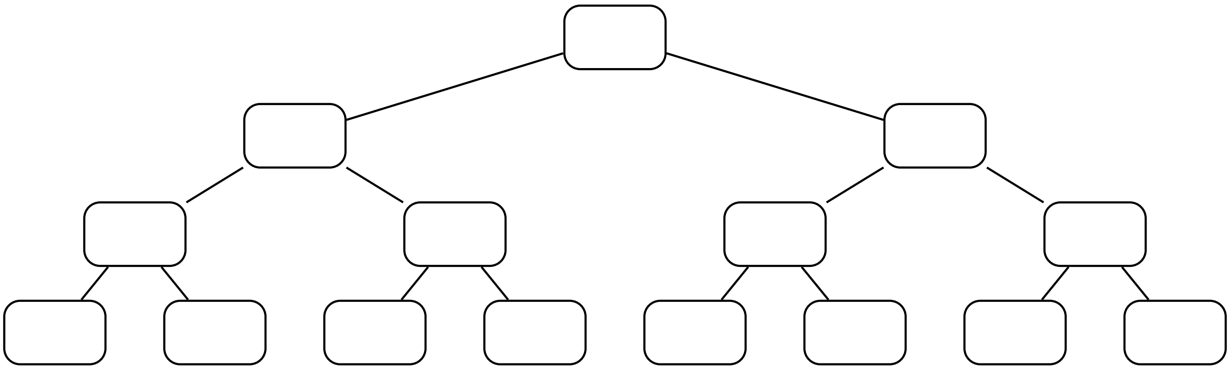
Abbildung 2: Intervall-Baum

Fügen Sie einen neuen Knoten mit dem Intervall $[8, 12]$ in den gegeben Baum ein. Geben Sie den resultierenden Baum nach dem Einfügen, sowie den gültigen reparierten Baum an. Verwenden Sie für Ihre Lösung die vorgedruckten Bäume auf der nächsten Seite. Kennzeichnen Sie schwarze Knoten.

Einfügen:



Reparieren:



Aufgabe 7 — 15 Punkte

Aufgabe 7 a) — 10 Punkte

Sei S eine endliche Menge und $\mathcal{F}$ eine endliche Familie von Teilmengen von S. Es gilt: $\cup_{s \in \mathcal{F}} s = S$ und $\forall s \in \mathcal{F}$: $s \subseteq S$.

Geben Sie einen Algorithmus an, der eine Menge $T \subseteq \mathcal{F}$ findet mit $\cup_{t \in T} t = S$, für die $|T|$ minimal ist. Mit anderen Worten, die Menge T ist die kleinste Menge von Teilmengen von S aus $\mathcal{F}$, die S vollständig überdeckt.

Ihr Algorithmus muß nicht effizient sein. Geben Sie für Ihren Algorithmus die Laufzeit an.

Aufgabe 7 b) 5 Punkte

Kann durch dynamisches Programmieren ein Algorithmus gefunden werden, der das obige Problem in polynomieller Zeit optimal löst? Begründen Sie Ihre Antwort.

Aufgabe 8 — 8 Punkte

Gegeben sei eine endliche Menge S mit n Elementen.

Aufgabe 8 a) — 2 Punkte

Eine *Permutation* der Menge S ist eine geordnete Sequenz aller Elemente in S, wobei jedes Element genau einmal in der Sequenz vorkommt. Wieviele verschiedenen Permutationen der Menge S gibt es?

Aufgabe 8 b) — 2 Punkte

Eine *k-Permutation* der Menge S ist eine geordnete Sequenz der Länge k von Elementen aus S, wobei jedes Element nur einmal in der Sequenz vorkommt. Wieviele k-Permutationen der Menge S gibt es?

Aufgabe 8 c) — 2 Punkte

Eine *k-Kombination* der Menge S ist eine k-elementige Teilmenge von S. Wieviele k-Kombinationen der Menge S gibt es?

Aufgabe 8 d) — 2 Punkte

Wieviele Strings der Länge k gibt es über einem Alphabet mit m Zeichen?

Aufgabe 9	14 Punkte

Sei der gewichtete, gerichtete Graph aus Abbildung 3 gegeben:

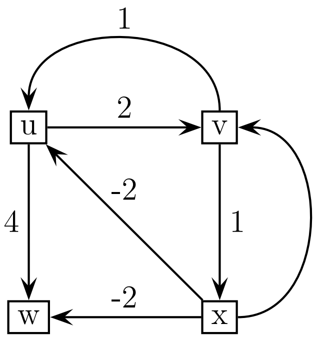


Abbildung 3: Gewichteter Graph

Aufgabe 9 a) 8 Punkte

Führen Sie den Bellman-Ford-Algorithmus auf dem gegebenen Graphen mit Ausgangsknoten u aus.

Geben Sie die Werte π und d für jeden Knoten nach der Initialisierung sowie **nach jedem Durchgang durch die Kantenmenge an**. Führen Sie die Relaxationsschritte entsprechend der lexikographischen Ordnung der Knotenbezeichner aus. Die Kante (u, v) wird beispielsweise vor der Kante (u, w) besucht.

Knoten	Initialisierung		Durchgang 1		Durchgang 2		Durchgang 3	
	π	d	π	d	π	d	π	d
u								
v								
w								
x								

Aufgabe 9 b) 1 Punkte

Geben Sie den kürzesten Pfad von Knoten u nach Knoten w mit dem entsprechenden Gesamtgewicht an.

Aufgabe 9 c) 5 Punkte

Sei $G = (V, E)$ ein gewichteter, gerichteter Graph. G enthält keine negativen Zyklen. Geben Sie einen modifizierten Bellman-Ford-Algorithmus (`Bellman-Ford-T(`G`,`w`,`s`)`) in Pseudocode an, der so früh wie möglich terminiert.