Ludwig-Maximilians-Universität München
Institut für Informatik
PD Dr. Matthias Renz
Dr. Erich Schubert, Daniel Kaltenthaler

München, 6.7.2015

Algorithmen und Datenstrukturen

SS 2015

Übungsblatt 12: Klausurvorbereitung

Wird nicht besprochen – Musterlösung ab 15.7.2015

Die Aufgaben auf diesem Blatt sind zusätzliches Übungsmaterial, und Wiederholungen früherer Aufgaben.

Diese Aufgaben decken nur einen kleinen Teil des Vorlesungsstoff ab, und sind vom Umfang bewusst groß gewählt. Sie können aber mit diesen Aufgaben üben, ob sie die Algorithmischen Abläufe dieser Verfahren verstanden haben, sie sollten aber zur Klausurvorbereitung dennoch auch alle anderen Übungsblätter wiederholen, und das Skript studieren. Es handelt sich bei diesen Aufgaben nur um Zusatzmaterial, dass wir mit geringem Aufwand bereitstellen können!

Aufgabe 12-1 HeapSort

Sortieren Sie das folgende Array mit dem HeapSort-Verfahren *exakt wie es in der Vorlesung besprochen wurde*.

Geben Sie dazu eine Tabelle ab, mit:

- 1. Spalte: Interne Repräsentation als Array (inkl. serialisiertem Heap und bereits sortierten Daten)
- 2. Spalte: Interpretation als Baumstruktur (graphisch, ohne bereits sortiere Daten)

In die erste Zeile tragen Sie die initialen Daten ein (inkl. graphischer Interpretation als Baum!)

Nun erzeugen Sie den Heap, indem Sie ihn von unten nach oben korrigieren. Geben Sie die Daten in einer neuen Zeile immer dann an, *nachdem* ein Level *fertig korrigiert* ist. Kennzeichnen Sie in beiden Repräsentationen, welcher Teil des Heaps noch nicht korrigiert wurde. Alle Korrekturen im gleichen Level können Sie zusammen in einem Schritt durchführen, da diese voneinander unabhängig sind.

Sie sollten jetzt 4 Zeilen gefüllt haben. Kontrollieren Sie, ob Sie einen korrekten Heap haben!
Wenn Sie mit einem fehlerhaften Heap fortfahren, wird der Rest der Aufgabe mit 0 Punkten bewertet!

Anschließend führen Sie die zweite Phase des HeapSort durch, und zeichnen den Heap *nach* jedem Sortierschritt (d.h. auch *nach* dem Korrigieren des Heaps). Beachten Sie dass nach jedem Schritt ein korrekter Heap vorhanden sein sollte, wenn Sie keinen Fehler gemacht haben.
Wenn Sie mit einem fehlerhaften Heap fortfahren, wird der Rest der Aufgabe mit 0 Punkten bewertet!

Zu sortieren ist die folgende Liste von Zahlen: 15, 98, 47, 42, 41, 29, 59, 70, 52, 82

Aufgabe 12-2 Quick-Sort

Sortieren Sie die folgenden Zahlen mit dem Quick-Sort Algorithmus aus der Vorlesung:
36, 9, 58, 6, 57, 34, 27, 67, 38, 80, 79, 55, 53, 47, 72, 2, 5, 11, 3, 48

Anfang:	36	9	58	6	57	34	27	67	38	80	79	55	53	47	72	2	5	11	3	48

Vor Pivot:																				
Nach Pivot:																				
Vor Pivot:																				
Nach Pivot:																				
Vor Pivot:																				
Nach Pivot:																				
Vor Pivot:																				
Nach Pivot:																				
Vor Pivot:																				
Nach Pivot:																				
Vor Pivot:																				
Nach Pivot:																				
Vor Pivot:																				
Nach Pivot:																				
Vor Pivot:																				
Nach Pivot:																				
Vor Pivot:																				
Nach Pivot:																				
Vor Pivot:																				
Nach Pivot:																				
Vor Pivot:																				
Nach Pivot:																				
Vor Pivot:																				
Nach Pivot:																				
Vor Pivot:																				
Nach Pivot:																				
Vor Pivot:																				
Nach Pivot:																				

Aufgabe 12-3 B-Baum

Gegeben Sei ein leerer B-Baum der Ordnung 2.

- Fügen Sie folgende Schlüssel (in dieser Reihenfolge) ein:
  98, 7, 97, 60, 33, 73, 56, 94, 86, 95, 67, 3, 44, 23, 19, 61, 25, 22, 4, 47, 79, 21
- Entfernen Sie folgende Schlüssel (in dieser Reihenfolge):
  60, 19, 3, 4, 98, 25, 33, 56, 86, 94, 23, 21, 67, 95, 97, 73, 79, 61, 44, 47, 7

Aufgabe 12-4 Lineares Hashing mit Partiellen Erweiterungen

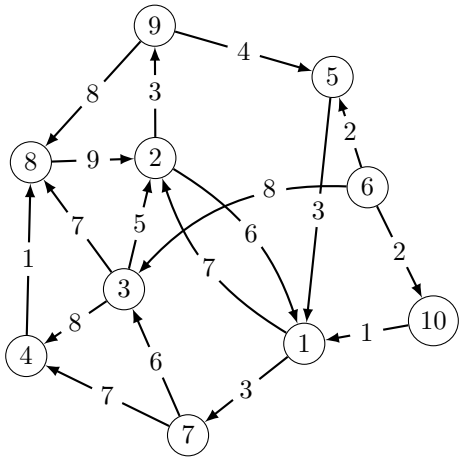
Gegeben Sei eine leere Hashtabelle mit der Strategie "Lineares Hashing mit Partiellen Erweiterungen" und folgenden Parametern:

Hashfunktion: $h_L(1,k) := k \bmod (2 \cdot 2^L)$
Hashfunktion: $h_L(2,k) := k \bmod (3 \cdot 2^L)$
Anfangsgröße: 2, Seitengröße: 2, Overflowseitengröße: 1
Füllgrad maximal: 0.70

Fügen Sie (in dieser Reihenfolge) folgende Objekte in die Hashtabelle ein:
35, 87, 78, 52, 22, 83, 40, 20, 27, 88, 32, 49, 4, 15, 14, 97, 95, 45, 76, 48

Aufgabe 12-5 Graphalgorithmen

Gegeben ist folgender Graph:



(a) Erstellen Sie die Adjazenzmatrix und Adjazenzlisten des Graphen.
Auf der Diagonalen und statt ∞ dürfen Sie auch $-$ schreiben.

(b) Führen Sie, ausgehend vom Knoten **6** einen Breitendurchlauf und einen Tiefendurchlauf durch den Graphen durch. Zeichnen Sie jeweils den Ergebnisspannbaum, der sich aus Breiten- und Tiefendurchlauf ergibt. Erstellen Sie den Spannbaum so, dass bei Mehrdeutigkeiten zuerst der Knoten mit der kleinsten Knoten-ID besucht wird.

(c) Berechnen Sie die kürzesten Wege ausgehend von Knoten **6** zu allen anderen Knoten. Verwenden Sie dazu den Algorithmus von Dijkstra. Zeichnen Sie für jeden Schritt einen Baum, der nur die kürzesten Wege enthält. Kennzeichnen Sie besuchte Knoten, um sie von "offenen" Knoten zu unterscheiden.

(d) Wenden Sie den Algorithmus von Floyd auf den Graphen an. Füllen Sie entsprechend dem Algorithmus die Kostenmatrix A. Die Einträge von A sollen dabei nicht nur die Kosten, sondern zusätzlich den Nachfolgerknoten (= den nächsten Schritt) des kostenminimalen Pfades in Klammern beinhalten. Die Kostenmatrix und die *pathCost*-Matrix werden somit in *einer* Tabelle dargestellt.

(e) Betrachten Sie den Graphen nun als ungerichteten Graphen. Konstruieren Sie mit Hilfe des Algorithmus von Kruskal den minimalen Spannbaum. Entscheiden Sie, ob das Ergebnis des Algorithmus eindeutig ist und begründen Sie Ihre Antwort.