Universität Konstanz
Fachbereich Informatik & Informationswissenschaft
Prof. Dr. Ulrik Brandes / Dr. Andreas Karrenbauer / Dr. Jürgen Lerner / Martin Mader

Algorithmen und Datenstrukturen
WS 2010/2011

5. Übungsblatt

Ausgabe: 17. November 2010 **Abgabe:** 24. November 2010, 10 Uhr
Die Bearbeitung in Zweiergruppen ist ausdrücklich erwünscht.

Aufgabe 1: Median — 5 Punkte

Sei der *Median* von $2n$ verschiedenen Elementen definiert als das Element, das größer als $n-1$ Elemente und kleiner als n Elemente ist.

Gegeben seien zwei Arrays A und B der Größe n die jeder für sich sortiert seien. Nehmen Sie zur Vereinfachung an, dass es keine zwei gleichen Elemente gibt und dass $n = 2^r$ gilt, für $r \in \mathbb{N}$. Geben Sie einen Algorithmus in Pseudocode an, der den Median dieser $2n$ Elemente in $\mathcal{O}(\log n)$ berechnet. Begründen Sie Korrektheit und Laufzeit.

Berechnen Sie mit Ihrem Algorithmus den Median von den folgenden zwei Arrays und dokumentieren Sie Ihre Schritte:

$$\begin{aligned} A &= [1, 4, 7, 9, 13, 15, 19, 20] \\ B &= [2, 3, 8, 10, 11, 12, 17, 18] \end{aligned}$$

Aufgabe 2: Binäre Suchbäume — 5 Punkte

(a) Gegeben sei die Menge der Schlüssel $\{1, 4, 5, 8, 10, 12, 13\}$. Zeichnen Sie binäre Suchbäume der Höhen $2, 3, 4, 5$ und 6.

(b) Fügen Sie die Elemente mit den Schlüsseln $2, 3, 11$ und 9 in dieser Reihenfolge in den Baum der Höhe 2 aus Aufgabe (a) ein.

(c) Löschen Sie aus dem Baum aus Aufgabe (b) die Elemente mit den Schlüsseln $1, 8, 9$ und 3 in dieser Reihenfolge.

(d) Was ist der Unterschied zwischen der Heap-Bedingung und der Suchbaumeigenschaft? Kann in einem Heap nach einem gegebenen Schlüssel gesucht werden; was ist die asymptotische Laufzeitklasse (Θ-Notation) für diese Suche?

Aufgabe 3: Selbstanordnende Listen **10 Punkte**

(a) Realisieren Sie das Interface `material.u05.IDictionary` durch eine generische einfach verkettete Liste ohne spezielle Update-Strategie. Leiten Sie dazu eine Klasse `u05.name.SimpleList` von der abstrakten Klasse `material.u05.AbstractList` ab. (2 Punkte)

(b) Leiten Sie weiterhin drei Klassen von `SimpleList` ab, die unterschiedliche Update-Strategien zur Selbstanordnung realisieren: (5 Punkte)

- `u05.name.MoveToFrontList` – angefragter Knoten wird an den Anfang verschoben.
- `u05.name.TransposeList` – angefragter Knoten wird mit seinem Vorgänger vertauscht.
- `u05.name.FrequencyCountList` – Knoten werden immer nach Zugriffshäufigkeit absteigend angeordnet.

(c) Geben Sie in einer Datei `u05_name_lists.pdf` die asymptotischen Laufzeiten ($\mathcal{O}$-Notation) der Methoden des Dictionaries bezüglich der Implementierungen an. Erläutern Sie außerdem folgende Fragen: Welche wesentlichen Änderungen ergeben sich bezüglich Einfachheit und Lesbarkeit des Quellcodes, praktischer Laufzeit und asymptotischer Laufzeit ($\mathcal{O}$-Notation), wenn

- eine einfach verkettete Liste ohne *tail*-Zeiger
- eine doppelt verkettete Liste (mit *tail*-Zeiger)

verwendet wird. (3 Punkte)

Hinweise:

- Beachten Sie bitte die Kommentare im Quellcode der Materialien. Verwenden Sie insbesondere die Klasse `ListNode` aus `AbstractList` für Ihre Implementierungen.
- Bei Aufgabe (c) lassen sich viele Fälle zusammenfassen. Antworten Sie möglichst kurz und präzise.