## Algorithmen und Programmierung 2, SS 2016 — 13. Übungsblatt

Abgabe bis Freitag, 8. Juli 2016, 12:00 Uhr

60. Verallgemeinerter Turm von Hanoi, Programmieraufgabe, 11 Punkte.

    Untersuchen Sie die Variante des Turmes von Hanoi, bei dem auch *gleich große* Scheiben erlaubt sind. Gleiche Scheiben dürfen übereinander liegen, aber eine Scheibe darf nach wie vor nicht über einer kleineren Scheibe liegen.

    Überlegen Sie, wie die Lösung für den klassischen Fall hergeleitet wird und warum diese Lösung optimal ist. Verallgemeinern Sie Ihre Überlegung für den Fall gleich großer Scheiben.

    Schreiben Sie eine Klasse `HanoiLoesungVerallgemeinert`, die die optimale Bewegungsfolge für diesen Fall berechnet. Zum Beispiel soll der Aufruf

    ```
    int[] a = {1,2,2,2};
    HanoiLoesungVerallgemeinert.löse(new Turm(a));
    ```

    das Problem mit 5 Bewegungen lösen.

61. Binärbäume, Rekursion, 5 Punkte

    Schreiben Sie (auf Papier) ein Java-Programm, das die Anzahl der Knoten in einem Binärbaum bestimmt, der durch folgende Klasse definiert ist.

    ```
    class Binärbaum <T> {
        T wert;
        Binärbaum <T> links, rechts; // Kinder
        ...
    }
    ```

62. *Radixsort* und Sortieren durch Fachverteilung, Programmieraufgabe, 14 Punkte.

    Programmieren Sie das Verfahren `RadixSort` zum Sortieren einer verketteten Liste von ganzzahligen `int`-Werten mit 32 Bits in linearer Zeit. Zerlegen Sie dazu die `int`-Werte $x$ in vier „Ziffern“ zu je 8 Bits. Diese Ziffern liegen im Bereich 0–255 und können durch Bitverschiebungen und Bitmasken aus $x$ bestimmt werden: `x & 0xff`, `(x>>>8)&0xff`, `(x>>>16)&0xff`, und `(x>>>24)`.

    Zum stabilen Sortieren nach einer Ziffer verwenden wir nicht Sortieren durch Zählen (Countingsort), sondern folgende Methode, die für verkettete Listen $L$ mit $n$ Elementen besser geeignet ist: Wir erstellen ein Feld von 256 leeren Listen $T_0, \ldots, T_{255}$; dann durchlaufen wir $L$ in $O(n)$ Zeit und verteilen die Elemente auf diese Listen, indem wir jedes Element $x$ am Ende der Teilliste $T_i$ anhängen, wenn die rechteste Ziffer von $x$ den Wert $i$ hat. Danach verketten wir die Listen $T_0, \ldots, T_{255}$ zu einer neuen Gesamtliste $L'$ und wiederholen das Verfahren für die zweite Ziffer von rechts, usw. Am Ende, bei der linkesten Ziffer, müssen wir zuerst die Fächer $T_{128}, T_{129}, \ldots, T_{255}$ zusammenhängen und anschließend die Fächer $T_0, T_1, \ldots, T_{127}$, damit das Vorzeichen richtig berücksichtigt wird und die negativen Zahlen vor den positiven kommen.

    (a) Schreiben Sie ein Programm für diesen Sortieralgorithmus. Benützen Sie für die Teillisten Ihre Lösung von Aufgabe 53. Zum Einlesen ganzer Zahlen von der Konsole oder aus einer Datei können Sie die Klasse `IntReader`[1] verwenden.

    (b) Zusatzfrage, 0 Punkte. Das Sortieren nach jeder Ziffer muss eigenlich stabil sein. Kann man trotzdem für die Teillisten auch Stapel statt Schlangen verwenden?

[1] `http://www.inf.fu-berlin.de/lehre/SS16/ALP2/IntReader.java`