Aufgaben zur Vorlesung "*Algorithmen und Datenstrukturen*"

Aufgabe 1:

(a) Lösen Sie die Rekursions-Gleichung

$$a_{n+2} = a_n + 2$$

für die Anfangs-Bedingungen $a_0 = 2$ und $a_1 = 1$. (10 Punkte)

(b) Lösen Sie die Rekursions-Gleichung

$$a_{n+2} = 2 \cdot a_n - a_{n+1}$$

für die Anfangs-Bedingungen $a_0 = 0$ und $a_1 = 3$. (10 Punkte)

Hinweis:

(a) Bei der Lösung der folgenden Aufgabe sind selbstverständlich die in der Vorlesung vorgestellten Algorithmen zu verwenden.

(b) Sie können bei der folgenden Aufgabe das Ergebnis graphisch als Baum angeben.

Aufgabe 2: Der AVL-Baum t sei durch den folgenden Term gegeben, wobei zur Vereinfachung auf die Angabe der Werte, die mit den Schlüsseln assoziiert sind, verzichtet wurde.

$$t = \mathit{node}(17, \mathit{node}(8, \mathit{node}(2, \mathit{nil}, \mathit{nil}), \mathit{node}(10, \mathit{nil}, \mathit{nil})), \mathit{node}(23, \mathit{nil}, \mathit{nil}))$$

(a) Fügen Sie in diesem Baum den Schlüssel `13` ein und geben Sie den resultierenden Baum an. (4 Punkte)

(b) Fügen Sie in dem Baum aus Teil den Schlüssel `15` ein und geben Sie den resultierenden Baum an. (3 Punkte)

(c) Entfernen Sie den Schlüssel 2 aus dem unter Teil (b) berechneten Baum und geben Sie den resultierenden Baum an. (4 Punkte)

Aufgabe 3: Betrachten Sie das folgende Programm:

```
sum := procedure(n) {
    i := 0;
    s := 0;
    while (i <= n) {
        s := i + s;
        i := i + 1;
    }
    return s;
}
```

Die Funktion *sum* soll die folgende Spezifikation erfüllen:

$$sum(n) = \tfrac{1}{2} \cdot n \cdot (n+1)$$

(a) Weisen Sie mit Hilfe des Hoare-Kalküls nach, dass das Programm korrekt ist.

(b) Beweisen Sie mit Hilfe der Methode der symbolischen Programm-Ausführung, dass das Programm korrekt ist.

Aufgabe 4: Im Abschnitt 8.2 des Skriptes werden Gleichungen angegeben, die das Einfügen und Löschen in einem Heap beschreiben. In diesem Zusammenhang sollen Sie in dieser Aufgabe einige zusätzliche Methoden auf binären Bäumen durch bedingte Gleichungen spezifizieren.

(a) Spezifizieren Sie eine Methode *isHeap*, so dass für einen binären Baum $b \in \mathcal{B}$ der Ausdruck b.`isHeap`() genau dann den Wert `true` hat, wenn b die *Heap-Bedingung* erfüllt.

(b) Spezifizieren Sie eine Methode *isBalanced*, so dass für einen binären Baum $b \in \mathcal{B}$ der Ausdruck b.`isBalanced`() genau dann den Wert `true` hat, wenn b die *Balancierungs-Bedingung* für *Heaps* erfüllt. (10 Punkte)

Aufgabe 5: Es sei $f(n) := \left(\sum_{i=1}^{n} \frac{1}{i}\right) - \ln(n)$. Zeigen Sie $f(n) \in \mathcal{O}\big(1\big)$. (12 Punkte)

Hinweis: Zeigen Sie die Ungleichung

$$0 \leq \left(\sum_{i=1}^{n} \tfrac{1}{i}\right) - \ln(n) \leq 1$$

indem Sie die Summe $\sum_{i=1}^{n} \frac{1}{i}$ durch geeignete Integrale abschätzen.

Aufgabe 6: Es sei $\mathcal{B}$ die Menge der binären Bäume, die im Skript definiert wird.

(a) Spezifizieren Sie eine Methode

$$isOrdered : \mathcal{B} \to \mathbb{B}$$

durch bedingte Gleichungen. Für einen binären Baum b soll der Aufruf $b.isOrdered()$ genau dann `true` zurück liefern, wenn $b \in \mathcal{B}_<$ gilt. (8 Punkte)

Hinweis: Definieren Sie sich geeignete Hilfsfunktionen.

(b) Es sei *insert*() die in Abschnitt 6.1 definierte Methode. Nehmen Sie an, dass Sie für alle $b \in \mathcal{B}_<$, alle Schlüssel k und alle Werte v die Gleichung

$$b.insert(k, v).isOrdered() = \texttt{true}$$

beweisen sollen. Geben Sie an, welche Lemmata über die in Teil (a) definierten Hilfsfunktionen zu einem solchen Beweis benötigt werden. (4 Punkte)

(c) Zeigen Sie nun für geordnete binäre Bäume die Gleichung

$$b.insert(k, v).isOrdered() = \texttt{true}$$ (12 Punkte)

Sie dürfen dabei die Lemmata, die Sie in Teil (b) angeben sollen, benutzen.

Aufgabe 7: Es gelte $\Sigma = \{\texttt{a}, \texttt{b}, \texttt{c}, \texttt{d}, \texttt{e}, \texttt{f}\}$. Die Häufigkeit, mit der diese Buchstaben in dem zu kodierenden String s auftreten, sei durch die folgende Tabelle gegeben:

Buchstabe	a	b	c	d	e	f
Häufigkeit	8	9	10	11	12	13

(a) Berechnen sie einen optimalen Kodierungs-Baum für die angegebenen Häufigkeiten.

(b) Geben Sie die Kodierung der einzelnen Buchstaben an, die sich aus diesem Baum ergibt.

Aufgabe 8:

(a) Use the LZW algorithm to encode the string "`aabbaaabbb`". Compute the compression factor for this string.

(b) Decode the list

$$[97, 98, 98, 128, 131]$$

using the LZW algorithm.