Westfälische Wilhelms-Universität Münster
Übungen zur Vorlesung „Datenstrukturen und Algorithmen“ im SoSe 2017

Prof. Dr. Klaus Hinrichs
Aaron Scherzinger

Blatt 5

Abgabe im Learnweb bis zum 01.06.2017 um 10 Uhr

Aufgabe 15: ((2+2+2+2+2+2)+(2+2+2+2+4)=24 Punkte)

(a) Geben Sie jeweils Funktionen f und g an, sodass die folgenden Bedingungen erfüllt sind. Sollte es keine solche Funktionen geben, vermerken Sie dieses entsprechend mit einer kurzen Begründung.

(i) $f(n) \in o(g(n))$ und $f(n) \notin \Theta(g(n))$
(ii) $f(n) \notin \Omega(g(n))$ und $f(n) \notin \mathcal{O}(g(n))$
(iii) $f(n) \in \Theta(g(n))$ und $f(n) \in o(g(n))$
(iv) $f(n) \in \Theta(g(n))$ und $f(n) \notin \mathcal{O}(g(n))$
(v) $f(n) \in \Omega(g(n))$ und $f(n) \notin \mathcal{O}(g(n))$
(vi) $f(n) \in \Omega(g(n))$ und $f(n) \notin o(g(n))$

(b) Geben Sie für die folgenden Rekurrenzen jeweils eine scharfe asymptotische Schranke an. Begründen Sie Ihre Wahl.

(i) $T(n) = 4 \cdot T(n/2) + n$
(ii) $T(n) = 4 \cdot T(n/2) + n^2$
(iii) $T(n) = 4 \cdot T(n/2) + n^3$
(iv) $T(n) = 2 \cdot T(\sqrt{n}) + \log_2 n$
(v) $T(n) = 2 \cdot T(n/2) + n \log_2 n$

Aufgabe 16: (6 Punkte) Mehrere Gruppen von jeweils n Informatikern ($n \geq 2$) sind zu einem Vorstellungsgespräch eingeladen worden. Um die Teamfähigkeit der Kandidaten zu testen (es werden nur aus n Personen bestehende Teams eingestellt!), spielt der Personalchef mit jedem Team folgendes Spiel:

Jeder Kandidat hält sich in einem eigenen Warteraum auf und hat keine Möglichkeit, mit den anderen Kontakt aufzunehmen. Der Personalchef holt jeden Kandidaten genau einmal aus seinem Raum und führt ihn in das Besprechungszimmer. In diesem Zimmer befindet sich ein Schrank mit unendlich vielen Schubladen, die durchnummeriert sind. Zu Beginn des Spieles ist jede Schublade leer. Außerdem liegt noch ein Haufen mit unendlich vielen Kieselsteinen bereit, die so groß sind, dass in jede Schublade genau ein Stein passt.

Jeder Kandidat kann nun eine beliebige Anzahl von Schubladen öffnen und einen Stein hineinlegen bzw. herausnehmen. Vor Betreten und nach Verlassen des Raumes müssen alle Schubladen wieder geschlossen sein. Wenn der Personalchef den Kandidaten in seinen Warteraum zurückführt, stellt er ihm die Frage: „Sind Sie das letzte Mitglied Ihres Teams, das ich abgeholt habe?“ Das Team gewinnt, wenn kein Kandidat diese Frage falsch beantwortet.

Gewinnen mehrere Teams, so ist das Einstellungskriterium, welches Team die effizienteste Strategie hatte. Die Komplexität einer Strategie ist die maximale Anzahl Schubladen, die ein einzelnes Mitglied des Teams öffnen musste.

Die Kandidaten haben jedoch vor Beginn des Spieles die Möglichkeit, eine Strategie abzusprechen. Der Personalchef ist bei dieser Besprechung anwesend und kann sich somit eine Abhol-Reihenfolge überlegen, die es dem Team möglichst schwer macht.

Geben Sie detailliert eine Strategie an, deren Komplexität hinsichtlich der verwendeten Schubladen in $\mathcal{O}(\log_2 n)$ liegt und mit der das Team immer gewinnt. Begründen Sie die Korrektheit und die Komplexität der Strategie.

Aufgabe 17: (2+3+4=9 Punkte) Betrachten Sie folgenden Algorithmus:

```
public static <T extends Comparable<T>>
    boolean myAlgo(T x, T[] array, int l, int r)
{
    if (r-l==1) {
        return (array[l].compareTo(x)==0);
    }
    else {
        int m = (l+r)/2;
        return ( myAlgo(x, array, l, m) || myAlgo(x, array, m, r) );
    }
}
```

Bearbeiten Sie nun die folgenden Aufgaben:

(a) Was wird durch den Aufruf `myAlgo(x, array, 0, array.length)` bestimmt? Hierbei sei `x` eine Instanz einer Realisierung des Interfaces `Comparable` und `array` sei ein Array von Elementen der gleichen Realisierung.

(b) Bestimmen Sie die asymptotische (worst-case) Laufzeitkomplexität des Algorithmus, wobei die Eingabegröße n durch die Länge des Arrays gegeben sei.

(c) Beweisen Sie die Korrektheit dieses Algorithmus.

Aufgabe 18: (2+6+6+6=20 Punkte) Betrachten Sie ein zweidimensionales Array `int[][] array` der Größe $m \times n$, das eine Matrix mit m Spalten und n Zeilen repräsentiert. Für die Einträge dieses Arrays soll gelten:

- $\texttt{array}[i][j] \leq \texttt{array}[i][j+1] \quad \forall (i,j) \in \{0,\dots m-1\} \times \{0,\dots n-2\}$
- $\texttt{array}[i][j] \leq \texttt{array}[i+1][j] \quad \forall (i,j) \in \{0,\dots m-2\} \times \{0,\dots n-1\}$

Bearbeiten Sie nun die folgenden Aufgaben:

(a) Wählen Sie 35 paarweise verschiedene ganze Zahlen aus dem Intervall $[1,\dots,99]$ aus und geben Sie exemplarisch ein Array der Größe 5×7 an, das die obigen Bedingungen erfüllt.

(b) Entwickeln Sie einen Algorithmus, der mit einer Laufzeit von $\Theta(m+n)$ ermittelt, ob eine übergegebene ganze Zahl x im Array enthalten ist. Beschreiben Sie diesen Algorithmus umgangssprachlich und verdeutlichen Sie Ihre Aussagen durch Skizzen.

(c) Implementieren Sie Ihren Algorithmus in `JAVA` ausgehend von der folgenden Methodendefinition:

```
public boolean arrayContainsElem(int[][] array, int x)
```

(d) Zeigen Sie die Terminierung und Korrektheit Ihres Algorithmus. Begründen Sie zudem seine Laufzeitkomplexität.