Karlsruher Institut für Technologie

Algorithmen 1 – Sommersemester 2018
Zuverlässige Softwaresysteme im Kontext der Automobilindustrie
`http://verialg.iti.kit.edu`

Prof. Dr. Carsten Sinz – Markus Iser

Übungsblatt 6

Ausgabe: 30.05.2018 – 15:30
Abgabe: 06.06.2018 – 13:00

Allgemeiner Hinweis

In diesem und allen folgenden Übungsblättern bedeutet "implementieren", sofern nicht explizit anders gewünscht, die Angabe von Pseudocode mit Kommentaren.

A Permutationen sortieren

Gegeben sei ein Array A von n Elementen, die jeweils aus Schlüssel und Datum bestehen. Die Schlüssel der Elemente in dem Array bilden eine Permutation der Zahlen $\{1...n\}$. D.h. *jeder* Schlüssel eines Elements von A ist in $\{1...n\}$ enthalten und für jedes $z \in \{1...n\}$ exisitert *genau ein* Element im Array, das diesen Schlüssel hat.

A.1 Sortieren in Linearzeit (1 Punkt)

Implementieren und erklären Sie einen Algorithmus, der die Elemente aus A in $\Theta(n)$ nach ihrem Schlüssel sortiert in einem neuen Array B ablegt.

A.2 Sortieren in Linearzeit mit konstantem Speicherverbrauch (2 Punkte)

Implementieren und erklären Sie einen Algorithmus, der die Elemente aus A in $\Theta(n)$ nach ihrem Schlüssel sortiert und dabei nur $\mathcal{O}(1)$ zusätzlichen Platz benötigt, also *in-place* arbeitet.

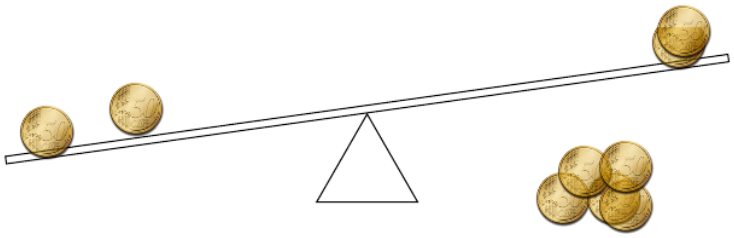
A.3 Schranke (1 Punkt)

Erläutern Sie kurz, warum Ihr Algorithmus kein Widerspruch zur unteren Schranke beim vergleichsbasierten Sortieren ist.

B Münzen wiegen (2 Punkte)

Sie haben 6 Euro in 50 ct Münzen, von denen eine gefälscht ist. Die falsche Münze hat ein anderes Gewicht als die echten Münzen. Wie kann mit höchstens drei Wiegevorgängen auf einer Balkenwaage die falsche Münze in jedem Fall identifiziert werden, und gleichzeitig, ob diese leichter oder schwerer ist? (Für Lösungen für zehn 50 ct Münzen gibt es einen Trostpunkt.)

Hinweis: Entwerfen Sie eine Schreibweise, die kurz aber immer noch präzise ist. Ein reiner Prosatext aus Fallunterscheidungen ist sehr schwer nachvollziehbar



C k-Way Merging

Gegeben seien n Elemente in k doppelt-verketteten sortierte Listen $L_1, \ldots, L_k$ der Länge $\lfloor \frac{n}{k} \rfloor$ (oder $\lceil \frac{n}{k} \rceil$).

C.1 Einfache Version (2 Punkte)

Implementieren und erklären Sie einen Algorithmus, der in Zeit $\mathcal{O}(n \cdot k)$ eine aufsteigend sortierte Liste L erzeugt, die genau die Elemente der k sortierten Listen $L_1, \ldots, L_k$ enthält. Verwenden Sie in dieser Aufgabe (im Gegensatz zu Aufgabe C.2) *keine* zusätzliche Datenstuktur. Begründen Sie die Laufzeit des Algorithmus.

C.2 Bessere Version (2 Punkte)

Implementieren und erklären Sie einen Algorithmus, der in Zeit $\mathcal{O}(n \cdot \log k)$ eine aufsteigend sortierte Liste L erzeugt, die genau die Elemente der k sortierten Listen $L_1, \ldots, L_k$ enthält. Verwenden Sie dazu intermediär eine geeignete Datenstruktur zur Zwischenspeicherung der Elemente. Begründen Sie die Laufzeit des Algorithmus.

Karlsruher Institut für Technologie

Algorithmen 1 – Sommersemester 2018
Zuverlässige Softwaresysteme im Kontext der Automobilindustrie
`http://verialg.iti.kit.edu`

Prof. Dr. Carsten Sinz – Markus Iser

Deckblatt Übungsblatt 6

Tutoriumsnummer:

Name	Matrikelnummer	Unterschrift

Mit unseren Unterschriften bestätigen wir, dass wir die Aufgaben eigenständig gelöst haben.

Hinweis: Das Übungsblatt darf in Gruppen von bis zu zwei Personen bearbeitet werden. Beide Personen müssen demselben Tutorium zugeteilt sein. Möchte jemand seine Abgaben-Gruppe innerhalb des Semesters wechseln, so ist dies im Voraus mit dem Tutor abzusprechen. **Bitte tragen Sie oben *groß* die Nummer Ihres Tutoriums ein.** Die Lösung des Übungsblattes ist in jedem Fall mit diesem Deckblatt abzugeben.

Bewertung (durch Tutor):