3. Übungsblatt zu Grundzüge von Algorithmen und Datenstrukturen, WS 15/16

Prof. Markus Bläser
http://www-cc.cs.uni-saarland.de/course/50/

Abgabe: 12. November 2015, 12:00 Uhr

Aufgabe 3.1 Ein Array $a[1..n]$ heißt k-approximativ sortiert, falls für alle $1 \leq i \leq n-k$ gilt:

$$\sum_{j=i}^{i+k-1} a[j] \leq \sum_{j=i+1}^{i+k} a[j].$$

a) Was bedeutet es, wenn ein Array 1-approximativ sortiert ist?

b) Zeigen Sie: a ist k-approximativ sortiert genau dann wenn $a[i] \leq a[i+k]$ für alle $1 \leq i \leq n-k$ gilt.

c) Geben Sie einen Algorithmus mit Laufzeit $O(n \log \frac{n}{k})$ an, der ein Array k-approximativ sortiert.

Aufgabe 3.2 (Hinweis: Zu dieser Aufgabe existieren bestimmt Musterlösungen aus früheren Jahren. Die Datenstruktur ist jedoch wichtig. Machen Sie bitte die Aufgabe selbst.) Eine Prioritätswarteschlange ist eine Datenstruktur, die eine Menge S von Elementen verwaltet, von denen jedes eine bestimmte Priorität hat. Folgende Operationen muss eine Prioritätswarteschlange unterstützen:

- `INSERT`(S, x) – fügt Element x in die Menge S ein.
- `MAXIMUM`(S) – gibt das Element mit der höchsten Priorität zurück.
- `EXTRACT-MAX`(S) – gibt das Element mit der höchsten Priorität zurück und entfernt es aus S.
- `INCREASE-KEY`(S, x, k) – erhöht die Priorität des Elements x auf den neuen Wert k. Voraussetzung ist, dass k mindestens so groß wie die bisherige Priorität von x ist.

Zeigen Sie, wie man mit einem Heap eine Prioritätswarteschlange implementieren kann, so dass die obigen Operationen Laufzeit $O(\log n)$ haben, wenn $n = |S|$ ist.

Aufgabe 3.3 Wir untersuchen die Laufzeit von Quicksort auf einem Array mit n Elementen.

a) Nehmen Sie an, dass auf einer bestimmten Eingabe Quicksort die zu sortierenden Einträge in jedem Rekursionsschritt so aufteilt, dass der kleinere Teil mindestens 1/10 aller Einträge ausmacht. Zeigen Sie, dass Quicksort auf solchen Eingaben Laufzeit $O(n \log n)$ hat.

b) Geben Sie für jedes n ein Array an, so dass Quicksort (mit dem jeweils ersten Element als Pivot) $\Theta(n^{4/3})$ Vergleiche macht.