

Datenstrukturen und Algorithmen – Informatik I

7. Übung

Abgabe der Lösungen: 21. Juni 2004

Hinweis: Die Implementierungsaufgaben sind in Java oder C/C++ zu lösen. Der Quellcode sowie ein dokumentierter Beispiellauf sind beim Betreuer der Kleingruppe abzugeben.

Aufgabe 30: Heapsort

4 Punkte

Führen Sie das in der Vorlesung behandelte Sortierverfahren Heapsort manuell auf der folgenden Sequenz von Schlüsseln durch. Geben Sie dabei sämtliche Zwischenschritte für beide Phasen des Algorithmus (Heapaufbau und Sortierung) an.

16, 5, 2, 7, 4, 8, 10, 12, 11, 6, 13, 14, 15, 3, 9, 1

Aufgabe 31: Programmieraufgabe: Paarsuche

4 Punkte

Gegeben sei eine sortierte Liste L von n natürlichen Zahlen und eine natürliche Zahl x . Gesucht wird ein Paar (a, b) von Elementen aus L , für das gilt: $x = a + b$.

Implementieren Sie einen Algorithmus, der diese Aufgabe in $O(n)$ Schritten löst. Dabei soll das Verfahren (bis auf die Speicherung der Liste) nur konstanten Platz benötigen. Der Algorithmus soll ein solches Paar zurückliefern, falls eines existiert.

Aufgabe 32: Multiplikative Hashfunktion

3 Punkte

Das multiplikative Hashverfahren verwendet die folgende Hashfunktion:

$$h(x) = \lfloor m(Ax \bmod 1) \rfloor$$

Zeigen Sie, daß für $x, y \in \mathbb{N}$, $x \neq y$ und $A \in \mathbb{R} \setminus \mathbb{Q}$ gilt:

$$Ax \bmod 1 \neq Ay \bmod 1$$

Aufgabe 33: Prioritätswarteschlangen

4 Punkte

Unter einer Prioritätswarteschlange versteht man einen Datentyp, der folgende Operationen unterstützt:

Put(x) fügt Element x in die Warteschlange ein.

Min() gibt das Element mit dem kleinsten Schlüssel zurück.

GetMin() entfernt das Element mit dem kleinsten Schlüssel.

ReplaceMin(x) ersetzt das Element mit dem kleinsten Schlüssel durch x .

Welche Laufzeitkomplexitäten haben die genannten Operationen, wenn man die Prioritätswarteschlange mit den unten aufgeführten Datenstrukturen realisiert? Unterscheiden Sie gegebenenfalls zwischen worst- und best-case und begründen Sie Ihre Antworten!

- unsortierte Liste

- sortierte Liste
- binärer Suchbaum
- Heap (*Tip*: Passen Sie die Heap-Eigenschaft geeignet an.)

Welche Datenstruktur eignet sich am besten?

Aufgabe 34: Hashing

6 Punkte

In einem kleinen Studentenkino mit $m = 19$ Plätzen wird ein Film der Filmförderung-NRW gezeigt. Um dem Ansturm auf die wenigen guten Plätze Einhalt zu gebieten, sollen die Plätze auf die in einer Schlange stehenden Studentinnen und Studenten mit einem Hashverfahren verteilt werden.

Als Schlüsselnummer werden dabei nur die beiden letzten Ziffern der Matrikelnummern der in der Schlange stehenden verwendet. Als Adressen stehen die Platznummern 0 bis 18 zur Verfügung. Welche Platznummern erhalten die folgenden in einer Schlange anstehenden Besucher

83, 52, 37, 52, 44, 91, 82, 95, 63, 12, 54, 41, 19, 01, 88, 15, 01, 17, 08

(angegeben sind jeweils nur die beiden letzten Ziffern der Matrikelnummern) bei Verwendung

- 1) der Quersumme als Hashfunktion?
- 2) der Divisions-Rest-Methode als Hashfunktion?
- 3) der multiplikativen Methode auf Aufgabe 32 mit $A := \frac{\sqrt{5}-1}{2}$?

Lösen Sie für alle angegebenen Hashverfahren evtl. auftretende Platzkollisionen durch

- a) lineares Sondieren (s.u.)
- b) quadratisches Sondieren (s.u.)

für eine geeignete Primzahl m .

Zu Aufgabe 34:

Sei $x \in U$ ein Schlüssel aus dem Universum und m eine (geeignet zu wählende) Primzahl. Die (Re)Hashfunktionen $h(x, j)$, $j = 0, 1, 2, \dots, m-1$ sollen so gewählt werden, daß für jedes x der Reihe nach sämtliche m Zellen der Hashtabelle inspiziert werden. Zwei Möglichkeiten sind:

Lineares Sondieren:

$$h(x, j) = (h(x) + j) \pmod{m} \quad 0 \leq j \leq m-1$$

Quadratisches Sondieren:

$$h(x, j) = (h(x) + j^2) \pmod{m} \quad 0 \leq j \leq m-1$$