PS Algorithmen und Datenstrukturen 2026

Aufgabenblatt 6

Aufgabe 16

Eine *Queue* ist eine *FIFO*-Struktur (first-in-first-out). Das heißt, Elemente werden genau in der gleichen Reihenfolge abgerufen (dequeue), in der sie zuvor gespeichert (enqueue) wurden. Geben Sie die beiden Operationen $\mathrm{enqueue(Q,x)}$ und $\mathrm{dequeue(S)}$ in Pseudocode an, wenn für die Queue Q eine einfach verkettete Liste verwendet wird. Das heißt, die Elemente sollen lediglich einen key und einen next-Verweis auf das nächste Element besitzen. Beide Operationen sollen Zeitkomplexität $O(1)$ aufweisen, eventuell sind konstant viele Zeiger hinzuzufügen.

Aufgabe 17

Betrachten Sie eine Hashtabelle der Größe $m = 11$, die Verkettung für die Kollisionsverwaltung verwendet. Wir möchten Einträge speichern, bei denen Strings als Schlüssel genutzt werden. Für Strings $s \in \Sigma^*$ verwenden wir die folgende, gängige Hashfunktion (alle Operationen werden modulo m ausgeführt):

$$h(s) = \begin{cases} 0 & \text{falls } s = \epsilon, \\ 31 \cdot h(t) + c & \text{falls } s = tc,\ t \in \Sigma^*,\ c \in \Sigma \end{cases}$$

Der Wert eines Zeichens c entspricht seiner Position im Alphabet, d.h. wie in folgender Tabelle angegeben:

a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**	**l**	**m**
1	2	3	4	5	6	7	8	9	10	11	12	13
n	**o**	**p**	**q**	**r**	**s**	**t**	**u**	**v**	**w**	**x**	**y**	**z**
14	15	16	17	18	19	20	21	22	23	24	25	26

Beantworten Sie die folgenden Fragen:

1. Angenommen es wird uniformes Hashing genutzt, wie lange (Anzahl an Listenzugriffen, auch auf NIL) dauert eine erfolgreiche bzw. erfolglose Suche, wenn aktuell $n \in \{5, 8, 11, 22, 121\}$ Elemente gespeichert sind?
2. Führen Sie die unten stehenden Operationen in die gegebene Hashtabelle durch. Was ist der Lastfaktor? Wie lange dauert erwartet eine erfolgreiche Suche in dieser Hashtabelle nach Ausführung der Operationen (angenommen, es wird ein zufälliges Suchobjekt gewählt)? Wie lange dauert eine erfolglose Suche nach dem Suchobjekt "math"?

a. insert(a)
b. insert(al)
c. insert(alg)
d. insert(algo)
e. insert(d)
f. insert(da)
g. insert(dat)
h. insert(data)

Aufgabe 18

Gegeben sei ein binärer Suchbaum, in dem n Zahlen gespeichert sind. Formulieren Sie einen Algorithmus, der den Median dieser Zahlen in Zeit $O(n)$ bestimmt und dabei nur $O(1)$ zusätzlichen Speicher benötigt. **Algorithmen, die auf Rekursion beruhen, benötigen in der Regel $O(h)$ zusätzlichen Speicher und sind somit nicht geeignet.**

Hinweis: Überlegen Sie zunächst einen Algorithmus zur Bestimmung der Anzahl der Knoten im Baum.