Informatik II SS 2006

Übungsblatt 9

Abgabe: 4.7.06 - 7.7.06

Aufgabe 1 (Hashing) Punkte: 2+2

a) Fügen Sie die Strings Berlin, Dortmund, Frankfurt, Gelsenkirchen, Hamburg, Hannover, Kaiserslautern, Koeln, Leipzig, Muenchen, Nuernberg und Stuttgart in eine Hashtabelle der Länge 17 ein. Verwenden Sie die Hashfunktion $h(k) = 2 \cdot (ord(k_1) - ord('A')) + 3 \cdot (ord(k_3) - ord('a'))$ (*ord()* ist eine Python-Funktion). Hierbei bezeichnet k_i den i-ten Buchstaben des Strings k. Kollisionen sollen durch Double-Hashing mit $s(j,k) = j \cdot len(k)$ aufgelöst werden. Geben Sie die ausgefüllte Hashtabelle sowie für jeden Eintrag den Hashwert $h(k)$ und alle die im Zuge der Kollisionsauflösung berechneten Offsets $s(j,k)$ an.

b) Gegeben sei die Hashtabelle der Größe 7 mit der Belegung

0	1	2	3	4	5	6
1	164	8	21	73	22	89

und der Hashfunktion $h(k) = quersumme(k) \mod 7$. Bei Kollision wird quadratische Sondierung verwendet.

Geben Sie alle Reihenfolgen an, in denen die Schlüssel in die anfangs leere Hashtabelle eingefügt worden sein können.Gibt es eine andere Reihenfolge, die zu einer geringeren durchschnittlichen Anzahl zu inspizierender Hashtabellenplätze bei der erfolgreichen Suche führt, wenn die Suche nach jedem Schlüssel gleich wahrscheinlich ist? Wenn ja, geben Sie eine an, wenn nicht Begründen Sie wieso nicht.

Aufgabe 2 (Worst-Case-Betrachtung) Punkte: 2

Wie viele Schritte werden im schlechtesten Fall benötigt um in eine anfangs leere Hashtabelle n Schlüssel einzufügen. wenn für die Kollisionsbehandlung *Chaining* verwendet wird? Wie viele Schritte benötigt man um nach jedem der n eingefügten Schlüssel einmal zu suchen?

Aufgabe 3 (Divisions-Rest-Methode) Punkte: 3

Betrachten Sie die Hashfunktion $h_{a,m}(x) = (a \cdot x) \mod m$ mit m prim.
Zeigen Sie, daß $\{h_{a,m} | 0 \leq a < m\} = \{h_{a,m} | a \in \mathbb{Z}\}$.
Zeigen Sie weiterhin daß $h_{a,m}(x),\ x = 0, 1, 2, 3....$ mit $a \neq c \cdot m, c \in \mathbb{Z}$ Periode m hat und jede Periode eine Permutation von $(0, 1, ..., m-1)$ darstellt.

Aufgabe 4 (Open Addressing) Punkte: 3

Zeigen Sie daß in einer Hashtabelle der Länge m bei einer Sondierung mit $s(j,k) = j^2$ für einen festen Hashwert höchstens $\lceil \frac{m}{2} \rceil$ Einträge erreicht werden.

Aufgabe 5 (Hashtabellen in Python) Punkte: 0.5+4.5+2

Schreiben Sie ein Programm, welches Hashing mit $h_{a,m}(x) = x \mod m$ implementiert. Für die Kollisionsbehandlung soll *Open Addressing* mit linearer Sondierung ($s(j,k) = j$) verwendet werden. Implementieren Sie dazu

a) eine Klasse *TableEntry*, welche die Datenstruktur für einen Eintrag der Hashtabelle enthält,

b) eine Klasse *HashTable*, welche die Hashtabelle selbst implementiert. Die Klasse soll die Funktion *h(self,key)*, *s(self,j,k)*, *insert(self,key)* und *delete(self,key)* enthalten.

Schreiben Sie ein Testprogramm, welches die Hashtabelle mit gleichverteilten Zufallszahlen zu 50% füllt und danach $2 \cdot m$ mal abwechselnd ein neues Element in die Hashtabelle einfügt und ein Element aus der Hashtabelle löscht, d.h. $4m \cdot (insert(),\ delete())$. Auch hier sollen je ein zufälliger Werte eingefügt und ein zufällig gewähltes Element , das in der Hashtabelle enthalten ist, gelöscht werden. Geben Sie die Anzahl der Kollisionen für *insert* und *delete* aus. Führen Sie die Tests für m = 11, 101 und 1009 durch. Mitteln Sie je 10 Werte (Anzahl an Kollisionen) für m = 1009.