Einführung in die Programmierung

Präsenzpraktikum

Hashing

Aufgabe 1

Schreiben Sie ein Programm, das die Operationen `insert()` und `delete()` auf einer Hashtabelle implementiert und dabei den Besuch einer Zelle mitprotokolliert (vgl. *hierzu Einsendeaufgabe 6* zu LE *Dynamische Datenstrukturen und spezifische Algorithmen*). Kennzeichnen Sie ein gelöschtes Feld entsprechend. Verwenden Sie zur Kollisionsbehandlung quadratisches Sondieren mit alternierendem Vorzeichen, dh.

$h_{2i-1}(x) = (h(x) + i^2)$ **mod** m
$h_{2i}(x) = (h(x) - i^2)$ **mod** m

für $1 \leq i \leq \frac{m-1}{2}$

Verwenden Sie die Datei `./PROG/Hashing/HT.java`, die auch ein kurzes Testprogramm enthält.

Aufgabe 2

Ist eine Menge von Schlüsseln im voraus bekannt (und damit konstant), so kann man versuchen, die Hashfunktion *h* so zu wählen, dass sie injektiv ist, d.h. dass es keine Kollisionen gibt (perfektes Hashing).

a) Wie groß muss eine Hashtabelle mindestens sein, um perfektes Hashing für die Abspeicherung von Flughafen-Codes zu erlauben. Flughafen-Codes setzen sich dabei aus drei Großbuchstaben zusammen.

b) Entwickeln Sie eine perfekte Hashfunktion (A=1, B=2, .., Z=26), die möglichst einfach sein soll und die nachfolgenden 10 Flughafen-Codes in eine möglichst kleine Tabelle abbildet:

München (MUC)	Stuttgart (STR)	Palma (PMI)
Luxemburg (LUX)	Hamburg (HAM)	Friedrichshafen (FDH)
Valencia (VLC)	Köln (CGN)	Malaga (AGP)
Faro (FAO)		

Schreiben Sie dazu ein Programm mit einer Hashfunktion, die den Wert der jeweiligen Buchstaben mit unterschiedlichen Faktoren multipliziert und kollisionsfrei auf die 10 Behälter verteilt.

Verwenden Sie das im Verzeichnis `./PROG/Hashing/Perfekt.java` stehende Rahmenprogramm.