Einsendeaufgabe 7

Stefan Berger

1.

Ein Verein habe 756 Mitglieder, und auch in der Zukunft werde der Verein nicht mehr als 1000 Mitglieder haben. Eine Mitglieds-Nummer gebe es nicht. Für diesen Verein ist eine Mitglieder-Datei anzulegen. Die Mitglieder-Datei ist als eine Hashtabelle mit offener Adressierung und einem initialen Belegungsfaktor von 1/2 zu implementieren. Dabei sind die ersten fünf Buchstaben des Nachnamens für das Hashing zu verwenden. Ist der Nachname kürzer, dann ist er durch Anhängen von Blanks entsprechend zu verlängern. Das zugrunde liegende Alphabet bestehe dabei aus 26 Buchstaben und dem Blank, eine Unterscheidung zwischen Groß- und Kleinbuchstaben werde nicht gemacht.

a) Spezifizieren Sie eine Hashfunktion.

$n = 1000$,
$\alpha = 0.5$,
$l =$ Nachname, $n_l =$ Länge von l,
$k = \begin{cases} l\{\textvisiblespace\}^{5-n_l} & \text{falls } n_l < 5 \\ l_0 l_1 \dots l_4 & \text{sonst} \end{cases}$,
$m =$ eine Zweierpotenz $\geq n/\alpha \geq 8 = 2048$,
$a = 8\lfloor m/23 \rfloor + 5 = 717$,
$h(k) = (ak_1 + a^2 k_2 + \dots + a^5 k_5) \mod m$

b) Bestimmen Sie den aktuellen Belegungsfaktor.

$\alpha = n/m = 756/2048 \approx 0.37$

c) Welche mittlere Länge hätten die Kollisionsketten, wenn zu Hashing mit geschlossener Adressierung übergegangen werden würde?

Die mittlere Länge der Kollisionskette wäre gleich dem Belegungsfaktor. Bzw. wäre die Länge der Kollisionsketten im mittleren Fall 0, denn $\alpha < 0.5$, d.h. bei einer Kollision wird im Mittel an der nächsten Position in der Hashtabelle ein Platz zur Verfügung stehen ($\lfloor \alpha + 0.5 \rfloor$).

2.

Was geben in Java die folgenden Anweisungen aus? Erklären Sie kurz, wie die HashFunktion in diesen drei Klassen implementiert ist.

```
System.out.println("11".hashCode());
```

Ausgabe: 1568

$$h(k) = \begin{cases} 0 & \text{falls } n = 0 \\ s[0] \cdot 31^{n-1} + s[1] \cdot 31^{n-2} + \ldots + s[n-1] & \textit{sonst} \end{cases}$$

wobei $s[i]$ der Zeichencode des $i$ten Zeichen des Strings und n die Länge des Strings ist.

```
System.out.println(new Integer(1000).hashCode());
```

Ausgabe: 1000

$h(k) = k$

```
ArrayList<Integer> arl = new ArrayList<Integer>();
System.out.println(arl.hashCode());
```

Ausgabe: 1

```
arl.add(1);
System.out.println(arl.hashCode());
```

Ausgabe: 32

```
arl.add(1);
System.out.println(arl.hashCode());
```

Ausgabe: 993

$$h(k) = \begin{cases} 1 & \text{falls } n = 0 \\ 31^n + h_g(l[0]) \cdot 31^{n-1} + h_g(l[1]) \cdot 31^{n-2} + \ldots + h_g(l[n-1]) & \text{sonst} \end{cases}$$

wobei $l[i]$ das $i$te Listenelement und h_g die Hashfunktion für den Typparameter (hier Integer) ist.