Informatik III WS2004/2005

Übungsblatt 8

Institut Intelligente Systeme
Leiter: Prof. Dr. E. Lehmann
Dietmar Lippold

Besprechungstermin: 13.12.04 - 17.12.04

Aufgabe 1

Die folgende Prozedur gibt als Wert eine ver-n-fachende Prozedur zurück (also eine Prozedur, die ihr Argument mit n multipliziert):

```
(define (ver-n-fach n)
  (lambda (x)
    (* n x)))
```

1. Testen Sie `ver-n-fach`.
2. Erstellen Sie eine Prozedur höherer Ordnung `(verkette f g)`. Sie gibt die Verkettung $(f \circ g)(x) = f(g(x))$ zweier einstelliger Prozeduren $f(x)$ und $g(x)$ – also eine Prozedur – zurück.

   Man soll `verkette` so verwenden können: `((verkette f g) x)` – äquivalent dazu ist der Aufruf `(f (g x))`.
3. Erstellen Sie unter Verwendung von `verkette` und verschiedenen Ver-n-fachern, bei denen n Primzahlen sind, einen Ver-15-facher und einen Ver-18-facher.
4. Definieren Sie unter Benutzung von `verkette` und ohne Benutzung von `*` den Ver-n^3-facher `(ver-n-kub-facher n)`, der eine Prozedur zurückgibt, die ihr Argument mit n^3 multipliziert.
5. Erstellen Sie eine Prozedur `(k-fach f k)`, die die k-fache Verkettung der Prozedur $f^k(x) = f(f(f(\ldots(f(x))\ldots)))$ mit sich selbst zurückgibt.

   Stellen Sie aus Ver-n-fachern eine Prozedur zusammen, die ihr Argument ver-n^k-facht.
6. Man kann für assoziative Verkettung mit einem Aufwand von $O(\log k)$ auskommen (das ist bei f^k, der k-fachen Verkettung mit sich selbst, der Fall). Definieren Sie `(assoz-k-fach f k)` entsprechend.

Aufgabe 2 (schriftlich)

Es soll ein abstrakter Datentyp zur Darstellung von endlichen Mengen definiert werden. Im Gegensatz zu Listen gibt es bei Mengen keine Reihenfolge der Elemente und keine Wiederholung von Elementen. Stellen Sie Mengen intern als Liste der Elemente dar.

1. Der Konstruktor normalisiert die eingehenden Daten zur internen Darstellung. Definieren Sie einen Konstruktor `(menge elems)`, wobei `elems` eine Liste von Elementen ist.
2. Definieren Sie die folgenden Mengenoperationen:

`(hinein elem m)`	fügt ein Element zu `m` hinzu
`(enthalt? elem m)`	gibt zurück, ob das Element in `m` enthalten ist
`(entfern elem m)`	nimmt ein Element aus `m` heraus.
`(maecht m)`	gibt die Mächtigkeit (Elementanzahl) von `m` zurück
`(verein m1 m2)`	gibt die Vereinigungsmenge zurück
`(schnitt m1 m2)`	gibt die Schnittmenge zurück

   Zum Vergleich der Elemente benutzen Sie bitte `equal?`.

3. Geben Sie den Aufwand Ihrer in Aufgabenteil 2 definierten Prozeduren abhängig von der Mächtigkeit der beteiligten Mengen an.

   Beachten Sie dabei die Grundaufwände der Listenfunktionen: `member`, `length`, `append` haben Aufwand O(n), `cons`, `car` und `cdr` haben einen Aufwand von O(1).

4. Nennen Sie Ideen, wie man durch eine andere interne Repräsentation den Aufwand einiger Operationen reduzieren kann.

5. Angenommen, eine Anwendung fragt sehr häufig nach der Mächtigkeit von Mengen, während die anderen Mengenoperationen selten benutzt werden. Wie kann man mit einer speziellen Repräsentation die Effizienz verbessern?

Aufgabe 3

Im Folgenden sollen zusammengesetzte deutsche Wörter (Komposita) in ihre Bestandteile (Simplizia) zerlegt werden. Zu den dabei benötigen Prozeduren zur String-Verarbeitung lesen Sie bitte im R^5RS nach.
Gegeben sein folgende Liste von Simplizia: [1]

```
(define *wortliste* '("apfel" "alter" "arbeit" "bau" "bauer" "baum"
  "becken" "bluete" "ecken" "feuer" "fest" "freund"
  "feind" "gabe" "hof" "holz" "haus" "informatik"
  "leben" "mensch" "pruefung" "schule" "stau" "staub"
  "steil" "stein" "tag" "teil" "tisch" "zeit"))
```

1. Definieren Sie eine Prozedur `(bekannt? wort)`, die prüft, ob ein Wort in der oben gegeben Liste vorhanden ist.

2. Definieren Sie eine Prozedur `(zerlege wort trennpos)`, die ein Wort an der Position `trennpos` trennt, falls alle Teilworte bekannt sind. Der erste Buchstabe des übergebenen Wortes hat die Position 0. Beachten Sie dabei, daß bei manchen Komposita der letzte Buchstabe des ersten Wortes (e, n, s) weggelassen wird, zum Beispiel bei dem Wort *Schulhof* das "e" von *Schule*. Das Ergebnis soll als Liste zurückgegeben werden, wie in folgendem Beispiel:

   ```
   > (zerlege "apfelbaum" 5)
   ("apfel" "baum")
   ```

3. Schreiben Sie unter Verwendung der Prozedur `zerlege` eine Prozedur `(wortzerlegung wort)`, die ein Kompositum in die Liste der möglichen Simplizia zerlegt. Beispiel:

   ```
   > (wortzerlegung "staubecken")
   (("staub" "ecken") ("stau" "becken"))
   ```

4. Bei großen Wortlisten kann es sehr lange dauern, bis ein Ergebnis vorliegt. Woran liegt das und welche Verbesserungsmöglichkeiten gibt es?

---

[1] abrufbar unter `http://www.informatik.uni-stuttgart.de/ifi/is/Lehre/Vorlesung/info3.html`, Abschnitt Übungsblätter