Informatik III WS2004/2005

Übungsblatt 10

Besprechungstermin: 17.01.05 - 21.01.05

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

Aufgabe 1 (schriftlich)

Die folgende Prozedur berechnet den Binomialkoeffizienten $\binom{n}{m} = \frac{n(n-1)(n-2)...(n-m+1)}{m(m-1)...1}$ mit der Rekursionsformel $\binom{n}{m} = \binom{n-1}{m-1} + \binom{n-1}{m}$

```
(define (bink n m)
  (cond ((= m 0) 1)
        ((= m n) 1)
        (else (+ (bink (- n 1) (- m 1))
                 (bink (- n 1) m)) )))
```

1. Erstellen sie wie in Aufgabe 2.3 von Blatt 4 ein Protokoll der Prozeduraufrufe und der aufbewahrten Operationen für den Aufruf `(bink 3 2)`.
2. Ist der Prozeß rekursiv oder iterativ? Begründen Sie Ihre Antwort. [1]
3. Kann man die Prozedur auf einfache Weise ändern, so daß der andere Prozeß entsteht? Wenn ja, wie, wenn nein, woran liegt das?
4. Welchen Aufwand (abhänging von n) hat die Prozedur, wenn $m = n/2$ ist? Wie sieht es für $m = 2$ aus?
5. Wie könnte man diesen Aufwand senken?

Aufgabe 2

In der Vorlesungen wurde fünf verschiedene Prozeduren zur Erzeugung und Verarbeitung von Sequenzen (flachen Listen) vorgestellt.

1. Nennen die die Prozeduren mit ihren Parametern und definieren Sie deren Funktion.
2. Die nachfolgend angegebene Prozedur `accumulate-n` gleicht der Prozedur `accumulate` aus der Vorlesung, nur hat sie als drittes Argument eine Sequenz von Sequenzen, von denen angenommen wird, daß sie alle die selbe Anzahl von Elementen haben. Sie wendet die übergebene Akkumulationsprozedur `op` auf alle ersten Elemente der Sequenzen aus `sequences`, alle zweiten Elemente usw. an und liefert eine Sequenz von Sequenzen mit den Ergebnissen.

   Geben Sie die durch `<?1>` und `<?2>` bezeichneten fehlenden Ausdrücke in der Definition von `accumulate-n` an.

```
(define (accumulate-n op init sequences)
  (if (null? (car sequences))
      '()
      (cons (accumulate op init <?1>)
            (accumulate-n op init <?2>))))
```

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

3. Implementieren Sie analog eine Prozedur `(map-n proc sequences)`, die die Prozedur `proc` auf alle ersten Elemente von `sequences`, auf alle zweiten Elemente usw. anwendet und eine Sequenz von Sequenzen mit den Ergebnissen liefert.

4. Welche Ähnlichkeiten und Unterschiede bestehen zwischen der Standard-Prozedur `map` und der neu definierten Prozedur `map` sowie den Prozeduren `accumulate-n` und `map-n`?

Aufgabe 3

Vektoren von Zahlen kann man als Sequenzen (Listen) darstellen. Matrizen lassen sich als Sequenzen (Listen) von Vektoren als Zeilen der Matrix darstellen. Zum Beispiel würde die Matrix

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 5 & 6 & 6 \\ 6 & 7 & 8 & 9 \end{pmatrix}$$

dargestellt werden als die Sequenz ((1 2 3 4) (4 5 6 6) (6 7 8 9)). Auf Basis dieser Darstellung können die grundlegenden Operationen für Vektoren und Matrizen mit Hilfe der Operationen für Sequenzen ausgedrückt werden.
Implementieren Sie dementsprechend folgende Prozeduren:

- `(skalar-produkt v w)` liefert zu den Vektoren v und w die Summe $\sum_i v_i * w_i$
- `(transponiere m)` liefert zu der Matrix m die Matrix n mit $n_{ij} = m_{ji}$
- `(matrix-*-vektor m v)` liefert zu der Matrix m und dem Vektor v den Vektor t mit $t_i = \sum_j m_{ij} * v_j$
- `(matrix-*-matrix m n)` liefert zu den Matrizen m und n die Matrix p mit $p_{ij} = \sum_k m_{ik} * n_{kj}$

Verwenden Sie die in der Vorlesung dargestellten Prozeduren für Sequenzen, die Prozeduren `accumulate-n` und `map-n` aus der vorhergehenden Aufgabe sowie die jeweils schon definierten Prozeduren dieser Aufgabe.

Aufgabe 4

In Scheme gibt es den Datentyp Vektor, ein eindimensionales Array.

1. Definieren Sie eine Prozedur `(int-vek n)`, die einen Vektor mit den Zahlen von 0 bis $n-1$ liefert.

2. Definieren Sie eine Prozedur `(ohne-3er vek)`, die alle durch 3 teilbaren Zahlen im Vektor `vek` durch 0 ersetzt.

3. Definieren Sie eine Prozedur `(summe vek)`, die die Summe der Zahlen im Vektor `vek` berechnet.

4. Definieren Sie analog zum Listen–Akkumulator (vgl. Blatt 5 Aufgabe 1) einen Vektor–Akkumulator `(vek-akku vek f komb neutral)`, der auf jedes Element von `vek` die Prozedur `f` anwendet und die Ergebnisse mit der Prozedur `komp` verknüpft, wobei `neutral` das neutrale Element der Verknüpfung ist.