8. Aufgabenblatt zu Funktionale Programmierung vom 17.12.2003, fällig: 07.01.2004

Themen: *Funktionale, Listen und Matrizen*

Für dieses Aufgabenblatt sollen Sie Haskell-Rechenvorschriften zur Lösung der im folgenden angegebenen Aufgabenstellungen entwickeln und für die Abgabe in einer Datei namens `Aufgabe8.hs` auf oberstem Niveau in Ihrem Homeverzeichnis ablegen. Kommentieren Sie Ihre Programme wieder aussagekräftig und benutzen Sie, wo sinnvoll, Hilfsfunktionen und Konstanten.

Ihre Lösungen sind besonders gut, wenn sich die Rümpfe der zu implementierenden Funktionen möglichst ausschließlich auf andere Funktionale, Funktionen und Funktionskomposition abstützen, nicht aber auf Fallunterscheidungen, elementare Operatoren wie $+$, $*$, etc.

- In dieser Aufgabe soll ein curryfiziertes polymorphes Haskell-Funktional `listMan` zur Manipulation von Listen von Listen geschrieben werden. Neben einer Liste von Listen als zweites Argument erwartet `listMan` als erstes Argument eine Funktion, die festlegt, wie die Elemente der Listenelemente des zweiten Arguments zu manipulieren sind. Beispielsweise soll der Aufruf `listMan (*2) [[2],[2,4,3,5],[],[34,5]]` das Ergebnis `[[4],[4,8,6,10],[],[68,10]]` liefern.

  Geben Sie eine entsprechende Implementierung von `listMan` an.

  Um dieses Funktional zu testen, sollen zusätzlich (Manipulations-) Funktionen `mOne`, `mTwo` und `mThree` implementiert werden, mit denen Elementlisten über Zeichenreihen und ganzen Zahlen manipuliert werden können. Dabei wird `mOne` die Zeichenreihen umkehren, `mTwo` dies entsprechend für ganze Zahlen leisten und `mThree` die ganzzahligen Elemente einer Liste aufaddieren. Die folgenden Beispiele illustrieren die gewünschten Funktionalitäten.

  1. `listMan mOne [["abc","de"],["XYz"],[],["rTcE","ZBN"]] = [["cba","ed"],["zYX"],[],["EcTr","NBZ"]]`
  2. `listMan mTwo [[234,45622],[24,35],[],[56]] = [[432,22654],[42,53],[],[65]]`
  3. `listMan mThree [[1,2,3],[2],[2,4,3,5],[],[34,5]] = [[6],[2],[14],[],[39]]`

- (Zweidimensionale) Matrizen mit m Zeilen und n Spalten (sog. $m \times n$-Matrizen) können in Haskell als Listen mit m Elementlisten (Zeilen) der Länge n (Spalten) repräsentiert werden.

  1. In dieser Teilaufgabe wollen wir ein polymorphes Haskell-Funktional `matTransp` schreiben, das $m \times n$-Matrizen (in der aus der Mathematik bekannten Weise) transponiert, also Zeilen und Spalten vertauscht.
     Angewendet auf die Matrix

     $$\begin{array}{r} [\ [x_{11} \ldots x_{1m}], \\ \ldots \\ [x_{n1} \ldots x_{nm}]\ ] \end{array}$$

     soll die Funkion `matTransp` die transponierte Matrix

     $$\begin{array}{r} [\ [x_{11} \ldots x_{n1}], \\ \ldots \\ [x_{1m} \ldots x_{nm}]\ ] \end{array}$$

     zurückliefern. Im Rumpf von `matTransp` sollen Sie nach Möglichkeit nur den Listenkonstruktionsoperator : und Funktionskomposition verwenden.
  2. Auf gleichdimensionalen Matrizen definieren wir die Grundrechenarten dadurch, dass jeweils die indexgleichen Komponenten der beiden Argumentmatrizen addiert, subtrahiert, etc. werden. Schreiben Sie ein curryfiziertes polymorphes Haskell-Funktional `matGRA`, das die vier Grundrechenarten auf gleichdimensionalen Matrizen realisiert. Dabei erwartet `matGRA` zunächst die Grundrechenart und anschließend die beiden gleichdimensionalen Argumentmatrizen.
  3. Vektoren können als Spezialfall von Matrizen angesehen werden. $1 \times m$-Matrizen werden dabei üblicherweise als Zeilenvektoren, $n \times 1$-Matrizen als Spaltenvektoren bezeichnet. In dieser Aufgabe wollen wir Vektoren noch einfacher als nichtleere Listen der Länge n ansehen. Das (Skalar-) Produkt zweier Vektoren U und V der Länge n ist dann durch folgende Summe definiert:

     $$\sum_{j=1}^{n} u_j * v_j$$

Schreiben Sie eine curryfizierte polymorphe Haskell-Rechenvorschrift `skalProd`, die das Skalarprodukt zweier gleichdimensionierter Vektoren berechnet.

4. Sei A eine $p \times q$- und B eine $q \times r$-Matrix. Dann ist das (Matrix-) Produkt von A und B eine $p \times r$-Matrix C mit

$$c_{ik} = \sum_{j=1}^{q} a_{ij} * b_{jk}$$

Im folgenden Beispiel ergibt sich etwa:

$$\begin{pmatrix} 2 & -3 & 5 \\ 1 & 2 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 & -3 \\ 5 & 0 \\ 2 & 4 \end{pmatrix} = \begin{pmatrix} -3 & 14 \\ 11 & -3 \end{pmatrix}$$

Schreiben Sie ein curryfiziertes polymorphes Haskell-Funktional `matMult` zur Matrixmultiplikation zweier jeweils passend dimensionierter Matrizen. (Bemerkung am Rande: Matrixmultiplikation ist nicht kommutativ.)

Frohe Weihnachten,
einen guten Rutsch und
viel Erfolg im neuen Jahr!