4. Aufgabenblatt zu Funktionale Programmierung vom 02.11.2011. Fällig: 09.11.2011 / 16.11.2010 (jeweils 15:00 Uhr)

Themen: *Funktionen auf ganzen Zahlen, Listen und Matrizen*

Für dieses Aufgabenblatt sollen Sie die zur Lösung der unten angegebenen Aufgabenstellungen zu entwickelnden Haskell-Rechenvorschriften in einer Datei namens `Aufgabe4.lhs` im home-Verzeichnis Ihres Gruppenaccounts auf der Maschine g0 ablegen. Wie bei der Lösung zum dritten Aufgabenblatt sollen Sie auch dieses Mal ein "**literate Script**" schreiben. Versehen Sie wie auf den bisherigen Aufgabenblättern alle Funktionen, die Sie zur Lösung brauchen, mit ihren Typdeklarationen und kommentieren Sie Ihre Programme aussagekräftig. Benutzen Sie, wo sinnvoll, Hilfsfunktionen und Konstanten. Im einzelnen sollen Sie die im folgenden beschriebenen Problemstellungen bearbeiten.

Auf diesem Aufgabenblatt beschäftigen wir uns mit Operationen zur Vektor- und Matrizenrechnung. Wir benutzen die Bezeichnungen von Aufgabenblatt 3 (die wir hier noch einmal wiederholen) und einige weitere Bezeichnungen und Begriffe im Zusammenhang mit Matrizen:

- (m,n)-Matrix $\mathcal{M}$ für $m,n \in \mathit{IN}$:

$$\mathcal{M} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} = (a_{ik})_{(m,n)}$$

  Für $m \neq n$ heißt $\mathcal{M}$ *rechteckig* vom Typ (m,n), für $m = n$ heißt $\mathcal{M}$ *quadratisch* vom Typ (m,m).

- Zeilen- und Spaltenvektor vom Typ $(1,n)$ und $(m,1)$:

  Eine $(1,n)$-Matrix $\alpha^{(n)}$ heißt *Zeilenvektor* der Länge n; eine $(m,1)$-Matrix $\alpha_{(m)}$ heißt *Spaltenvektor* der Länge m.

$$\alpha^{(n)} = (a_1, a_2, \ldots, a_n), \quad \text{Zeilenvektor der Länge } n$$

$$\alpha_{(m)} = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{pmatrix}, \quad \text{Spaltenvektor der Länge } m$$

- i-ter Zeilen- und Spaltenvektor α^i (lies: α oben i) und α_i (lies: α unten i) einer (m,n)-Matrix $\mathcal{M}$:

$$\alpha^i = (a_{i1}, a_{i2}, \ldots, a_{in}), \quad i\text{-ter Zeilenvektor vom Typ } (1,n) \text{ von } \mathcal{M}$$

$$\alpha_i = \begin{pmatrix} a_{1i} \\ a_{2i} \\ \vdots \\ a_{mi} \end{pmatrix}, \quad i\text{-ter Spaltenvektor vom Typ } (m,1) \text{ von } \mathcal{M}$$

- Mit obigen Bezeichnungen gilt:

$$\mathcal{M} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} = (a_{ik})_{(m,n)} = \begin{pmatrix} \alpha^1 \\ \alpha^2 \\ \vdots \\ \alpha^m \end{pmatrix} = (\alpha_1, \alpha_2, \cdots, \alpha_n)$$

- Transponierte (n,m)-Matrix $\mathcal{T}$ einer (m,n)-Matrix $\mathcal{M}$ für $m,n \in \mathit{IN}$:

  Die (m,n)-Matrix $\mathcal{M}$ geht durch Vertauschen ihrer Zeilen und Spalten in ihre transponierte (n,m)-Matrix $\mathcal{T}$ über (und umgekehrt).

  Für $\mathcal{M}$

$$\mathcal{M} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} \quad \text{ergibt sich} \quad \mathcal{T} = \begin{pmatrix} a_{11} & a_{21} & \cdots & a_{m1} \\ a_{12} & a_{22} & \cdots & a_{m2} \\ \cdots & \cdots & \cdots & \cdots \\ a_{1n} & a_{2n} & \cdots & a_{mn} \end{pmatrix}$$

- Multiplikation mit Skalar

  Bei der Multiplikation eines Skalars s mit einem (Zeilen-/Spalten-)Vektor bzw. einer Matrix wird jede Komponente des (Zeilen-/Spalten-)Vektors bzw. der Matrix mit dem Skalar multipliziert. Das Resultat ist also wieder ein (Zeilen-/Spalten-)Vektor bzw. eine Matrix gleichen Typs.

$$s \cdot \alpha^n = s \cdot (a_1, a_2, \ldots, a_n) = (s \cdot a_1, s \cdot a_2, \ldots, s \cdot a_n)$$

$$s \cdot \alpha_m = \begin{pmatrix} s \cdot a_1 \\ s \cdot a_2 \\ \vdots \\ s \cdot a_m \end{pmatrix}$$

$$s \cdot \mathcal{M} = \begin{pmatrix} s \cdot a_{11} & s \cdot a_{12} & \cdots & s \cdot a_{1n} \\ s \cdot a_{21} & s \cdot a_{22} & \cdots & s \cdot a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ s \cdot a_{m1} & s \cdot a_{12} & \cdots & s \cdot a_{mn} \end{pmatrix} = s \cdot (a_{ik})_{(m,n)}$$

  Die Multiplikation mit einem Skalar ist kommutativ.

- Skalarprodukt

  Das *Skalarprodukt* zweier Zeilen- oder Spaltenvektoren der Länge n ist die Summe der Produkte ihrer sich entsprechenden Komponenten.

$$\alpha^{(n)}\beta^{(n)} = \alpha^{(n)}\beta_{(n)} = \alpha_{(n)}\beta^{(n)} = \alpha_{(n)}\beta_{(n)} = \Sigma_{j=1}^{n} a_j b_j$$

- Matrixprodukt

  Das Produkt einer (m, n)-Matrix $\mathcal{A}$ und einer (n, p)-Matrix $\mathcal{B}$ ist die (m, p)-Matrix $\mathcal{P}$, in der das Element p_{ik} sich als das skalare Produkt des i-ten Zeilenvektors α^i von $\mathcal{A}$ mit dem k-ten Spaltenvektor β_k von $\mathcal{B}$ ergibt:

$$\mathcal{P} = \mathcal{A} \cdot \mathcal{B} = (\alpha^i \beta_k)_{(m,p)} = (\Sigma_{l=1}^{n} a_{il} b_{lk})_{(m,p)} = (p_{ik})_{(m,p)}$$

  Die Matrixmultiplikation ist nicht kommutativ.

- Matrixsumme

  Die Summe zweier (m, n)-Matrizen $\mathcal{A}$ und $\mathcal{B}$ ist die (m, n)-Matrix $\mathcal{S}$, in der das Element s_{ik} sich als Summe der Elemente a_{ik} von $\mathcal{A}$ und b_{ik} von $\mathcal{B}$ ergibt:

$$\mathcal{S} = \mathcal{A} + \mathcal{B} = (a_{ik} + b_{ik})_{(m,n)} = (s_{ik})_{(m,n)}$$

- Matrixpotenz

  Das Produkt einer quadratischen (m, m)-Matrix $\mathcal{A}$ mit sich selbst ist wieder eine (m, m)-Matrix. Für quadratische (m, m)-Matrizen können wir daher eine Potenzfunktion festlegen. Sei $\mathcal{A}$ eine (m, m)-Matrix, $m \geq 1$, und sei n ganzzahlig, $n \geq 0$. Dann legen wir fest:

$$\mathcal{A}^n = \begin{cases} \mathcal{E}_{(m,m)} & \text{falls } n = 0 \\ \mathcal{A} & \text{falls } n = 1 \\ (\mathcal{A}^{n-1}) \cdot \mathcal{A} & \text{sonst} \end{cases}$$

  Dabei bezeichnet $\mathcal{E}_{(m,m)}$ die *Einheitsmatrix* vom Typ (m, m), d.h. eine (m, m)-Matrix, in der alle Elemente in der Hauptdiagonalen (d.h. von links oben nach rechts unten) 1 sind und alle übrigen Elemente 0.

In Haskell können wir Matrizen und (Zeilen-/Spalten-) Vektoren über ganzen Zahlen als Listen von Listen ganzer Zahlen implementieren:

```
type Matrix = [[Integer]]
```

Zeilen- und Spaltenvektoren ergeben sich dann als spezielle Matrizen(werte). So werden die Matrizen und Vektoren

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{pmatrix}, \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{pmatrix}, \quad (1,2,3,4), \quad \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix}$$

durch folgende Werte des Typs `Matrix` dargestellt:

```
m1 = [[1,2,3,4],[5,6,7,8],[9,10,11,12]]
m2 = [[1,2,3],[4,5,6],[7,8,9],[10,11,12]]
zv = [[1,2,3,4]]
sp = [[1],[2],[3],[4]]
```

Schreiben Sie folgende Haskell-Rechenvorschriften, wobei zur Abkürzung folgende Typsynonyme eingeführt werden:

```
type Skalar = Integer
type Zeilen = Integer
type Spalten = Integer
type SpaZei = Integer
type Fuellwert = Integer
type ProtoMatrix =  ([[Integer]],Zeilen,Spalten,Fuellwert)

type Typung_mnpw = (Zeilen,SpaZei,Spalten,Fuellwert)
type Typung_mnw = (Zeilen,Spalten,Fuellwert)
type Typung_mw  = (SpaZei,Fuellwert)
type Potenz = Integer
type ProtoprotoMatrix = [[Integer]]
```

1. Eine Haskell-Rechenvorschrift

   `msk :: ProtoMatrix -> Skalar -> Matrix`

   Angewendet auf eine Protomatrix p bestehend aus einer Komponente L, einer Zahl von Zeilen z und von Spalten s, sowie einem Skalar sk und einem Füllwert w liefert der Aufruf `msk` das Ergebnis der Multiplikation der Matrix m mit dem Skalar sk, wobei m aus der Argumentkomponente L durch Normalisierung im Sinne von Teilaufgabe 2) von Aufgabenblatt 3) zu einer Matrix mit z Zeilen und s Spalten unter Verwendung des Füllwerts w entsteht.

   Folgende Beispiele illustrieren das gewünschte Aufruf-/Rückgabeverhalten:

   ```
   msk ([[1,2,3],[4,5,6]],2,3,9 )  5  -> [[5,10,15],[20,25,30]]
   msk ([[1,2,3],[4,5,6]],3,4,1 )  5  -> [[5,10,15,5],[20,25,30,5],[5,5,5,5]]
   msk ([[1,2,3],[4,5,6]],2,2,9 )  5  -> [[5,10],[20,25]]
   ```

2. Eine Haskell-Rechenvorschrift

   `mm :: ProtoprotoMatrix -> ProtoprotoMatrix -> Typung_mnpw -> Matrix`

   Angewendet auf zwei Protoprotomatrizen $p1$ und $p2$ und einer Typung (m, n, p, w) ist das Ergebnis des Aufrufs `mm` das Matrixprodukt der (m, n)-Matrix $m1$ und der (n, p)-Matrix $m2$, wobei $m1$ und $m2$ aus $p1$ und $p2$ durch Normalisierung im Sinne von Teilaufgabe 2) von Aufgabenblatt 3) zu (m, n)- bzw. (n, p)-Matrizen unter Verwendung des Füllwerts w entstehen.

   Folgende Beispiele illustrieren das gewünschte Aufruf-/Rückgabeverhalten:

   ```
   mm [[1,2,3],[4,5,6]]  [[1,2],[3,4],[5,6]]  (2,3,2,9)  -> [[22,28],[49,64]]
   mm [[1,2,3],[4,5,6]]  [[1,2],[3,4],[5,6]]  (3,2,4,1)  -> [[7,10,3,3],[19,28,9,9],[4,6,2,2]]
   mm [[1,2,3],[4,5,6]]  [[1,2],[3,4],[5,6]]  (1,3,1,9)  -> [[22]]
   ```

3. Eine Haskell-Rechenvorschrift

   `ms :: ProtoprotoMatrix -> ProtoprotoMatrix -> Typung_mnw -> Matrix`

   Angewendet auf zwei Protoprotomatrizen $p1$ und $p2$ und einer Typung (m, n, w) ist das Ergebnis des Aufrufs `ms` die Matrixsumme der (m, n)-Matrizen $m1$ und $m2$, die aus $p1$ und $p2$ durch Normalisierung im Sinne von Teilaufgabe 2) von Aufgabenblatt 3) zu (m, n)-Matrizen unter Verwendung des Füllwerts w entstehen.

   Folgende Beispiele illustrieren das gewünschte Aufruf-/Rückgabeverhalten:

```
ms [[1,2,3],[4,5,6]]  [[1,2,3],[4,5,6]]  (2,3,9)  -> [[2,4,6],[8,10,12]]
ms [[1,2,3],[4,5,6]]  [[1,2,3],[4,5,6]]  (3,2,1)  -> [[2,4],[8,10],[2,2]]
ms [[1,2,3],[4,5,6]]  [[6,5,4],[3,2,1]]  (1,2,3)  -> [[7,7]]
```

4. Eine Haskell-Rechenvorschrift

   `mp :: ProtoprotoMatrix -> Typung_mw -> Potenz -> Matrix`

   Angewendet auf eine Protoprotomatrix p, eine Typung (m, w) und eine Potenz n ist das Ergebnis des Aufrufs `mp` die n-te Potenz der (m, m)-Matrix mx, die aus p durch Normalisierung im Sinne von Teilaufgabe 2) von Aufgabenblatt 3) zu einer (m, m)-Matrix unter Verwendung des Füllwerts w entsteht. Ist der Wert von n negativ wird der Aufruf von `mp` durch `error "unzulaessig"` abgebrochen.

   Folgende Beispiele illustrieren das gewünschte Aufruf-/Rückgabeverhalten:

```
mp [[1,2],[3,4]]  (2,9) 0 -> [[1,0],[0,1]]
mp [[1,2],[3,4]]  (2,9) 1 -> [[1,2],[3,4]]
mp [[1,2],[3,4]]  (2,9) 2 -> [[7,10],[15,22]]
mp [[1,2],[3,4]]  (2,9) 3 -> [[37,54],[81,118]]
```

Werden die obigen Funktionen mit nicht positiven Argumenten für Zeilen- oder/und Spaltendimensionierung aufgerufen, endet die Auswertung mit dem Aufruf `error "unzulaessig"`.

Hinweis: Wenn Sie einzelne Rechenvorschriften aus früheren Lösungen wieder verwenden möchten, so kopieren Sie diese in die neue Abgabedatei ein. Ein `import` schlägt für die Auswertung durch das Abgabeskript fehl, weil Ihre alte Lösung, aus der importiert wird, nicht mit abgesammelt wird.

Denken Sie bitte daran, dass Sie für die Lösung dieses Aufgabenblatts ein "literate" Haskell-Skript schreiben sollen!

Haskell Live

Am Freitag, den 04.11.2011, werden wir uns in *Haskell Live* mit Lösungsvorschlägen von Aufgabenblatt 1 beschäftigen, die von Ihnen eingebracht werden können, sowie mit einigen der schon speziell für *Haskell Live* gestellten Aufgaben.

Anmeldung abgeschlossen

Die Anmeldung zur Teilnahme an der Vorlesung ist abgeschlossen. Teilnehmen können alle, die sich über TISS zur Teilnahme angemeldet und über das Lehrstuhlanmeldungssystem einen Rechnerzugang erhalten haben. Rechnerzugänge für Studierende, die nicht über TISS angemeldet sind und nicht teilnehmen können, sind nicht mehr zugänglich. Für Gruppenzugänge ändert sich dadurch nichts. Verbliebene Gruppenangehörige können ihren individuellen und Gruppenzugang in der bisherigen Weise für die Teilnahme an der Übung weiternutzen.