FACULTEIT EWI
Datum: 27 oktober 2013

Tentamen Functioneel Programmeren (211205)

29 oktober 2013
8:45 – 12:15 uur

Opmerkingen vooraf:
- U mag het dictaat *Functional Programming – an overview* en het overzicht van oparaties en functies bij dit tentamen gebruiken, verder niets.
- Voor Haskell: u mag alleen gebruik maken van functies uit de practicumomgeving, en uit de modules `Data.List` en `Data.Char`.
- **Geef bij *elke* functie die u definieert het type.**
- Beoordeling: er zijn vier opgaven, de zwaarte is bij elke opgave aangegeven.
- De elegantie van de oplossing zal ook een rol spelen, dus gebruik geen onnodige hulpfuncties.
- Succes!

Opgave 1 (25 punten). Een *bag* is een soort verzameling, maar een element kan er meerdere malen in voorkomen (in een verzameling zijn geen dubbele voorkomens van een element). Net als een verzameling zijn bags ongeordend (dus op dat punt verschillen ze van lijsten waar de ordening wel van belang is).

a. Definieer het type `Bag` door expliciet de multipliciteit van een element aan te geven, maar het element zelf maar één keer op te nemen (werk dus met tupels).

Bij onderstaande opgaven moet een element van multipliciteit nul uit de bag verwijderd worden.

b. Een bag `a` is een subbag van een bag `b` als alle elementen van `a` minstens even vaak voorkomen in `b`. Definieer een functie `subbag` die bepaalt of een bag een subbag is van een andere bag.

c. Schrijf een functie die de doorsnede van twee bags bepaalt. Het aantal voorkomens van een element is daarbij het minimum van dat element in beide bags.

d. Schrijf een functie die twee bags samenvoegt tot één bag, waarbij het aantal voorkomens van een element het totaal is van de voorkomens in de beide oorspronkelijke bags.

Opgave 2 (25 punten). Deze opgave gaat over het *splitsen* van een matrix in submatrices. Gegeven zijn de volgende types:

```
type Matrix  = [[Int]]
type MatrixM = [[Matrix]]
```

Splitsen wordt hieronder uitgelegd aan de hand van een voorbeeld, waarin een 5×6 matrix wordt gesplitst in 2×3 submatrices.

– Oorspronkelijke 5×6 matrix (van type `Matrix`):

$$\begin{bmatrix} 11 & 12 & 13 & 14 & 15 & 16 \\ 21 & 22 & 23 & 24 & 25 & 26 \\ 31 & 32 & 33 & 34 & 35 & 36 \\ 41 & 42 & 43 & 44 & 45 & 46 \\ 51 & 52 & 53 & 54 & 55 & 56 \end{bmatrix}$$

– Resultaat (van type `MatrixM`) van het splitsen in 2×3 matrices:

$$\left[\begin{bmatrix} 11 & 12 & 13 \\ 21 & 22 & 23 \end{bmatrix} \begin{bmatrix} 14 & 15 & 16 \\ 24 & 25 & 26 \end{bmatrix}\right.$$
$$\left[\begin{bmatrix} 31 & 32 & 33 \\ 41 & 42 & 43 \end{bmatrix} \begin{bmatrix} 34 & 35 & 36 \\ 44 & 45 & 46 \end{bmatrix}\right.$$
$$\left.\left[\begin{bmatrix} 51 & 52 & 53 \end{bmatrix} \begin{bmatrix} 54 & 55 & 56 \end{bmatrix}\right.\right]$$

Merk op dat aan de onderkant en/of de rechterkant eventuele "restmatrices" kunnen optreden (in het voorbeeld alleen aan de onderkant).

a. Schrijf een functie `splits` van type

```
splits :: Matrix -> Int -> Int -> MatrixM
```

zodat de expressie

```
splits mat m n
```

het resultaat levert van het splitsen van matrix `mat` in `m` $\times$ `n` submatrices.

b. Schrijf een functie die alle getallen in elke submatrix optelt (en dus weer een matrix van type `Matrix` oplevert).

Opgave 3 (25 punten).

a. Definieer een type `Tree` van bomen die aan elke node en aan elk blad een getal bevat, en op elke node een willekeurig aantal subbomen mag hebben.

b. De *mediaan* van een verzameling getallen is het middelste getal qua grootte, dat wil zeggen: de helft van de getallen is kleiner (of gelijk) aan de mediaan, en de andere helft is groter (of gelijk). Als de verzameling een even aantal getallen bevat, mag u zelf kiezen welke van de twee mogelijkheden u neemt als mediaan.

Schrijf een functie `minstensMed` die alle getallen in de boom die kleiner zijn dan de mediaan vervangt door nul.

c. In een *pad* door een boom wordt bij iedere stap de "afslag" aangegeven, waarbij een "afslag" in een node de index is die aangeeft welke subboom in die node gekozen moet worden. Een *pad* naar een node in een boom is de lijst van *afslagen* die in iedere node in het pad genomen moet worden, te beginnen in de wortel (de root) van de boom.

Schrijf een functie `subboom` die die subboom uit een boom oplevert waarvan de root wordt aangewezen door een pad. Als het pad te lang is, moet een foutmelding worden gegeven.

d. Schrijf een functie `padnaar` die gegeven een boom en een getal het pad oplevert dat van de root van de boom naar één van de voorkomens van dat getal in die boom gaat. Als het getal niet in de boom voorkomt moet een foutmelding worden gegeven.

e. Schrijf een functie `vervangDoorPaar` die elk getal in een boom van type `Tree` vervangt door een paar van getallen, bestaande uit het totaal van alle getallen in de subboom op dat punt, en uit de diepte van de subboom op dat punt.

Geef ook het type van de resulterende boom.

Opgave 4 (25 punten). Gegeven zijn de types

$$\begin{array}{lcl} \textbf{type } \mathit{Node} & = & \mathit{Int} \\ \textbf{type } \mathit{Graph} & = & [(\mathit{Node}, [\mathit{Node}])] \end{array}$$

Het type *Graph* is een lijst van geordende paren (n, ms), waarbij node n uitgaande edges heeft naar precies alle nodes in de lijst ms. Merk op dat de lijst ms leeg kan zijn.

a. Schrijf een functie *bereikbaar* die, gegeven een gerichte graaf van type *Graph* en een startnode, de lijst van alle nodes oplevert die vanuit die startnode bereikbaar zijn. Daarbij mogen edges alleen in de goede richting worden doorlopen.

b. De *ingraad* van een node is het aantal binnenkomende edges van die node. Schrijf een functie *ingraad* die de ingraad van een node bepaalt.

c. Een *cycle* $[a_0, a_1, \ldots, a_{n-1}, a_0]$ is een lijst van nodes zodanig dat twee opeenvolgende nodes steeds door een edge verbonden zijn (in de goeie richting), en dat de laatste node gelijk is aan de eerste node (in dit geval a_0). Bovendien mag iedere node slechts één keer in de cycle voorkomen.

Schrijf een functie *isCycle* die test of een lijst van nodes een cycle vormt.

d. Schrijf een functie *bevatCycle* die test of een gegeven graaf een cycle zoals hierboven beschreven bevat.