FACULTEIT EWI
Datum: 24 oktober 2011

Tentamen Functioneel Programmeren (211205)

2 november 2011
8.45 – 12.15 uur

Opmerkingen vooraf:
- U mag het dictaat *Functional Programming – an overview* bij dit tentamen gebruiken, verder niets.
- Voor Haskell: u mag alleen gebruik maken van die voorgedefinieerde Haskell functies die equivalent zijn aan de built-in Amanda functies.
- **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 (20 punten).

a. Een getal n is *perfect* als de som van alle delers van n (inclusief 1, exclusief n) geijk is aan n zelf. Bijvoorbeeld: 6 en 28 zijn perfecte getallen, want hun delers zijn respectievelijk 1, 2, 3 en 1, 2, 4, 7, 14, en de totalen daarvan zijn weer 6 en 28.

Schrijf een functie die de lijst van alle perfecte getallen kleiner dan een gegeven getal m oplevert.

b. Een lijst xs heet een *jolly jumper* als de absolute waarde van de verschillen tussen alle tweetallen opeenvolgende elementen precies alle getallen in de range $1, \ldots, n-1$ aannemen (n is de lengte van xs). Bijvoorbeeld: $[1, 4, 2, 3]$ is een jolly jumper, want de lijst van absolute waardes van de opeenvolgende verschillen is $[3, 2, 1]$.

Schrijf twee varianten van een functie die test of een gegeven lijst een jolly jumper is: één maal met recursie, één maal met hogere orde functies.

c. Een matrix is een lijst van lijsten van gelijke lengte, waarbij elke lijst een rij in de matrix is. Definieer drie varianten van een functie die een $n \times m$ met een $m \times k$ matrix vermenigvuldigt (in $a \times b$ is a het aantal rijen, en b het aantal kolommen): met recursie, met hogere orde functies, en met lijstcomprehensie.

Opgave 2 (20 punten).

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

b. Schrijf een functie `vervangDoorMax` die alle getallen in een boom van type `tree` vervangt door het maximum van de boom.

c. Schrijf een functie `som` die het totaal van alle getallen in een boom van type `tree` oplevert.

d. Schrijf een functie `diepte` die de diepte van een boom van type `tree` bepaalt (de diepte van een boom is de lengte van het pad vanuit de wortel van de boom naar het verste blad).

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 door de diepte van de subboom op dat punt.

Geef ook het type van de resulterende boom.

Opgave 3 (30 punten). Gegeven is het volgende type voor expressies:

```
opA   ::= Add | Mul | ...

opB   ::= EQ | GT | LT | ...

exprA ::= Num num
        | OpA opA exprA exprA
        | If exprB exprA exprA

exprB ::= OpB opB exprA exprA
```

Het type `opA` ("A" voor "Arithmetical") bevat rekenkundige operaties *optelling*, *vermenigvuldiging*, etc. Het type `opB` ("B" voor "Boolean") bevat operaties voor *gelijkheid*, *groter-dan*, *kleiner-dan*, etc. U hoeft onderstaande deelopgaven alleen te beantwoorden voor de hierboven genoemde operaties.

Rekenkundige expressies kunnen een enkele constante (bijv. `Num 3`) zijn, of een rekenkundige samenvoeging van twee expressies, bijvoorbeeld

```
OpA Add (Num 3) (Num 5)
```

of een "if-then-else"-expressie (die een boolean expressie bevat), bijvoorbeeld

```
If (OpB LT (Num 3) (Num 5)) (Num 1) (Num 0)
```

De laatste expressie representeert de volgende expressie

if 3<5 *then* 1 *else* 0

in "standaardnotatie".

a. Schrijf functies `evalA`, `evalB` die de waarde van rekenkundige en boolean expressies uitrekenen.

b. Schrijf een *generic* functie `toString` die rekenkundige en boolean expressies omzet in standaardnotatie.

c. Beschouw de volgende eigenschappen van rekenkundige expressies:

associativiteit: $(a+b)+c = a+(b+c)$
$(a \times b) \times c = a \times (b \times c)$

distributiviteit: $a \times (b+c) = (a \times b) + (a \times c)$

Een rekenkundige expressie is in *normaalvorm* als alle haakjes zoveel mogelijk naar rechts zijn verschoven en alle vermenigvuldigingen zoveel mogelijk "naar binnen" zijn geschoven. Preciezer gezegd: een expressie is in normaalvorm als bovenstaande eigenschappen niet meer van links naar rechts kunnen worden toegepast. Merk op: a, b, c kunnen zelf ook weer samengestelde expressies zijn die nog in normaalvorm moeten worden gebracht).

Schrijf een functie `toNF` die rekenkundige expressies omzet naar normaalvorm ("NF" staat voor "Normal Form").

Opgave 4 (30 punten). Deze opgave gaat over gerichte grafen, waarvan de nodes aangegeven zijn door natuurlijke getallen (elke node uiteraard door een ander getal). Gegeven zijn de types

```
node  == num
graph == [ (node,[node]) ]
```

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`.

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* $[a_0, a_1, \ldots, a_{n-1}]$ 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 weer (ook in de goeie richting) door een edge met de eerste node is verbonden. Bovendien mag iedere node slechts één keer in de cykel voorkomen.

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