LV „ Programmierung “
Prof. Dr. H. Vogler / Dipl.-Math. R. Vater

SS 10

Aufgabenblatt zur 4. Übung

Zeitraum: 10.05. bis 14.05.2010

1. Aufgabe: (AGS 11.27*)

Es sei δ ein dreistelliges, γ ein einstelliges und α ein nullstelliges Funktionssymbol. Des Weiteren sei $V = \{x_1, \ldots, x_5\}$ eine Menge von Variablen.

Wenden Sie den Unifikationsalgorithmus jeweils auf die Terme t_1 und t_2 an und ermitteln Sie jeweils den allgemeinsten Unifikator:

- $t_1 = \delta(\gamma(x_1), \delta(\gamma(\alpha), \gamma(x_2), \gamma(\gamma(\alpha))), x_3)$
  $t_2 = \delta(\gamma(\gamma(x_5)), \delta(x_4, \gamma(x_2), \gamma(x_1)), \alpha)$
- $t_1 = \delta(\gamma(\alpha), x_3, \gamma(\gamma(x_3)))$
  $t_2 = \delta(x_1, \delta(\alpha, x_2, \alpha), x_2)$

2. Aufgabe: (AGS 11.32*)

Gegeben seinen folgende Definitionen von Funktionen:

```
 sumlist1 :: [Int] -> Int
sumlist1 [] = 0
sumlist1 (b:x) = b + (sumlist1 x)

sumlist2 :: [(Int,Int)] -> Int
sumlist2 [] = 0
sumlist2 ((a,b):y) = (a + b) + (sumlist2 y)

zip :: [t] -> [u] -> [(t,u)]
zip [] [] = []
zip (a:x) [] = []
zip [] (b:y) = []
zip (a:x) (b:y) = ((a,b):(zip x y))
```

Zeigen Sie mit Hilfe des Prinzips der Induktion über Listen, dass folgende Gleichung unter der Bedingung $\mathit{length}(\texttt{x1}) = \mathit{length}(\texttt{x2})$ gilt:

`sumlist2 (zip x1 x2) = (sumlist1 x1) + (sumlist1 x2)`

3. Aufgabe: (AGS 11.43)

Folgende Definitionen seien gegeben:

```
data Tree = Leaf Float | Branch Float Tree Tree

add :: Tree -> Float -> Tree
add (Leaf x)       a = Leaf (x+a)
add (Branch x l r) a = Branch (x+a/3) (add l (a/3)) (add r (a/3))

rev :: Tree -> Tree
rev (Leaf x)         = Leaf x
rev (Branch x l r)   = Branch x (rev r) (rev l)

sum :: Tree -> Float
sum (Leaf x)         = x
sum (Branch x l r)   = x + (sum l) + (sum r)
```

Zeigen Sie für die oben aufgeführten Definitionen mit Hilfe der strukturellen Induktion, dass die folgende Gleichung für einen beliebigen Baum `t :: Tree` und eine beliebige Zahl `a :: Float` erfüllt ist:

```
sum (add t a) = (sum (rev t)) + a
```

Geben Sie bei Umformungen die jeweils benutzten Gesetzmäßigkeiten / Definitionen an.

Zusatzaufgabe: (AGS 11.6*)

Gegeben sei folgende Definition von einfachen mathematischen Termen:

Ein Term repräsentiert entweder die Addition zweier Terme oder die Multiplikation zweier Terme oder eine ganze Zahl.

(a) Geben Sie einen algebraischen Datentyp `Term` für die Menge der beschriebenen Terme an.

(b) Geben Sie eine Haskell-Funktion `eval` an, die für einen gegebenen Term dessen repräsentierten Wert berechnet. Die berechnete ganze Zahl soll dabei als Term zurückgegeben werden. Hinweis: Programmieren Sie gegebenenfalls eine geeignete Hilfsfunktion.

(c) Geben Sie eine Haskell-Funktion `transform` an, welche eine Liste von Termen als Eingabe nimmt und diese in eine ebensolche wie folgt transformiert: Alle Terme, die negative Werte repräsentieren, sollen aus der Liste gestrichen werden; die anderen sollen in ihren repräsentierten Wert umgewandelt werden.

Achtung: Geben Sie immer auch die Typen der von Ihnen definierten Funktionen an.