5. Übungsblatt

Typ-basiertes Programmieren und Schließen in Funktionalen Sprachen

Jun.-Prof. Dr. Janis Voigtländer / Dipl.-Math. Daniel Seidel

Wintersemester 2009/10

Aufgabe 21
Beweisen Sie, dass wenn Funktionen *get*, *compl* und *inv* erfüllen dass:

1. $inv\ (get\ s, compl\ s) = s$
2. wenn $inv\ (v, c)$ definiert, dann $get\ (inv\ (v, c)) = v$
3. wenn $inv\ (v, c)$ definiert, dann $compl\ (inv\ (v, c)) = c$

dann mit der Definition

$$put\ s\ v' = inv\ (v', compl\ s)$$

auch folgende Eigenschaften erfüllt sind:

4. $put\ s\ (get\ s) = s$
5. wenn $put\ s\ v$ definiert, dann $get\ (put\ s\ v) = v$
6. wenn $put\ s\ v$ definiert, dann $put\ (put\ s\ v)\ (get\ s) = s$
7. wenn $put\ s\ v$ und $put\ (put\ s\ v)\ v'$ definiert, dann $put\ (put\ s\ v)\ v' = put\ s\ v'$ ◇

Aufgabe 22
Gegeben sei die Typdefinition

$$\textbf{data}\ Lense\ a\ b = Lense\ \{get :: a \to b, put :: a \to b \to a\}$$

Definieren Sie eine Funktion

$$lense\_comp :: Lense\ a\ b \to Lense\ b\ c \to Lense\ a\ c$$

so dass für $l3 = lense\_comp\ l1\ l2$ stets gilt, dass wenn $(get\ l1)/(put\ l1)$ und $(get\ l2)/(put\ l2)$ jeweils die Eigenschaften 4.–7. aus Aufgabe 21 erfüllen, dies auch für $(get\ l3)/(put\ l3)$ gilt, und darüber hinaus $(get\ l3) = (get\ l2) \circ (get\ l1)$. ◇

Aufgabe 23
Gegeben seien $get :: [a] \to [a]$ und folgende Definitionen:

$$
\begin{array}{l}
compl :: [a] \to (Int, [(Int, a)]) \\
compl\ s = \\
\quad \textbf{let} \\
\quad\quad n = (length\ s) - 1 \\
\quad\quad t = [0\,..\,n] \\
\quad\quad g = zip\ t\ s \\
\quad\quad g' = filter\ (\lambda(i, \_) \to notElem\ i\ (get\ t))\ g \\
\quad \textbf{in}\ (n + 1, g') \\
inv :: ([a], (Int, [(Int, a)])) \to [a] \\
inv\ (v', (n + 1, g')) = \\
\quad \textbf{let} \\
\quad\quad t = [0\,..\,n] \\
\quad\quad h = zip\ (get\ t)\ v' \\
\quad\quad h' = h +\!\!+ g' \\
\quad \textbf{in}\ map\ (\lambda i \to fromJust\ (lookup\ i\ h'))\ t
\end{array}
$$

Versuchen Sie zu argumentieren, dass *get*, *compl* und *inv* die Bedingungen 1.–3. aus Aufgabe 21 erfüllen. ◇