Übungsblatt 16 zu "Programmiersprachen"

Berthold Hoffmann, Studiengang Informatik (hof@informatik.uni-bremen.de)

Besprechung am 15. Juni 2010

Parametrische Polymorphe in objektorientierten Sprachen

Untersuchen Sie, wie weit parametrische Polymorphie auch in C++, Java oder C# mit *Templates* bzw. generischen Klassen realisiert werden kann.

1. Wie kann der parametrisierte Typ $\alpha\,\beta\,\mathrm{Pair} = \alpha * \beta$ realisiert werden?
2. Wie kann die Funktion second (x,y) = y realisiert werden?
3. Wie kann die Funktion id (x) = x realisiert werden?
4. Wie kann der parametrisierte Typ $\alpha\,\mathrm{List} = \mathrm{Nil}\ () \mid \mathrm{Cons}\ \alpha * (\beta\,\mathrm{List})$ realisiert werden?
5. Wie können die Funktionen head (x:y) = x, tail (x:y) = y realisiert werden?
6. Wie kann die Funktion length realisiert werden?

Typklassen in Haskell

Typklassen in Haskell schränken die klassische *universelle parametrische Polymorphie* funktionaler Sprachen ein.

Bei einem *beschränkt polymorphen Typ* **data** $C\alpha \Rightarrow T\alpha = \ldots$ darf für die Typvariable α ein Typ t nur dann eingesetzt werden, wenn t eine Instanz der Klasse C ist (in Haskell definiert mit "**instance** $C\ t$ **where** ...").

Klären Sie, wie dieses Konzept mit folgenden anderen Konzepten zusanmenhängt:

1. *ad-hoc*-Polymorphie (Überladen)
2. *Inklusions*-Polymorphie (Vererbung)

Geben Sie dafür Beispiele in C++, Java oder C#.