5. Arbeitsblatt

Woche 12.01.1998 und Woche 19.01.1998

Lektüre

Literatur zu Typklassen in Haskell.

Leitfragen

1. Was ist eine (Typ-)Klasse in *Hugs*?
2. Was ist der Unterschied zwischen `class` und `instance`?
3. In welcher Beziehung stehen Typen und Klassen in *Hugs*?
4. Wie können *(Type-)constraints* in *Hugs* notiert werden?
5. Welche vordefinierten Typklassen sind Euch bekannt?
6. Wie lassen sich für algebraische Typen Instantiierungen für `Eq` ableiten?

Anwendungsaufgaben (die in der Gruppe entwickelt und analysiert werden sollen)

1. **Déjà-vu?**
   Definiert eine Funktion `numEqual`, die zählt, wie oft ein Objekt `x` in einer Liste `xs` von Objekten gleichen Typs vorkommt. Definiert eine weitere Funktion `numLess` die zählt, wieviele Objekte in `xs` kleiner als `x` sind. Welche *(Type-)constraints* sind dabei zu beachten?

2. **Class Visible**
   In Thompsons Haskell-Buch wird die Klasse `Visible` wie folgt definiert:

   ```
   class Visible t where
    toString :: t -> String
    size     :: t -> Int
   ```

   - Unter welchen Voraussetzungen können 2- oder 3-Tuple-Typen zu `Visible` Typen gemacht werden?
   - Schreibt eine Funktion, die einen `Int` in einen `String` wandelt und zeigt, wie `Int` dann zu einer Instanz von `Visible` gemacht werden kann.
   - Unter welchen Bedingungen ließe sich eine `Queue`, die Instanz von `Visible` ist, realisieren?