Deklarative Programmierung
WS 24/25

Michael Hanus

20. November 2024

Detaillierter Vorlesungsverlauf

28.10. Organsiatorisches;
Einführung in die funktionale Programmierung: Variablenbegriff, Programm, Funktionsdefinitionen, Ausdrücke, Beispiel `square`, Beispiele `min`/`fac`, streng getypt vs. implizite Typkonversionen Auswertungsmöglichkeiten

30.10.: Fibonacci-Zahlen (rekursiv), Fibonacci-Zahlen (iterativ), lokale Definitionen, Layout-Regel, Vorteile lokaler Definitionen; Basisdatentypen, Typannotationen, algebraische Datentypen (Aufzählungstypen, Verbundtypen, gemischte Typen, Listen), Ausgabe von Daten (`deriving Show`), Operatoren

4.11. polymorphe Funktionen (`length`, `(++)`, `last`), Definition polymorpher algebraischer Datentypen, `Maybe` und `maybeHead`, Binärbäume, `String`, Vereinigungstypen (selbst definiert)

6.11. Vereinigungstypen (`Either`), Tupel (`fst`, `snd`), Funktionen `zip`, `unzip` Testen mit QuickCheck: Eigenschaften, `==>`, Referenzimplementierung, Regressionstests, automatisierte Ausführung aller Tests, Tests mit polymorphen Funktionen, Pattern Matching (Patternaufbau, case-Ausdrücke)

11.11. Guards; Funktionen höherer Ordnung, anynome Funktionen, partielle Applikation, Currying, Sections, Funktion `flip`, generische Programmierung (`map`, `foldr`, `filter`), Anwendung (`nub`, Quicksort)

13.11. `foldl`, Kontrollstrukturen als Funktionen höherer Ordnung (`while`), Funktionen als Datenstrukturen (Implementierung von Feldern), wichtige Funktionen höherer Ordnung (Komposition, `curry`/`uncurry`, `const`), Funktionen höherer Ordnung in imperativen Sprachen (Python, Java 8, JavaScript)

18.11. Motivation und Struktur von Typklassen, Instanzen, vordefinierte Funktionen in Typklassen, Typklassen für polymorphe Datentypen, Standardklassen, `deriving`, Klasse `Read` und Funktionen `read` und `reads`; Lazy Evaluation, Rechnen mit unendlichen Datenstrukturen (`from`, `primes`)

20.11. Rechnen mit unendlichen Datenstrukturen (`fibs`, `repeat`, `iterate`), Lazy Evaluation, Sharing, Graphreduktion, Vorteile von Lazy Evaluation, arithmetische und andere Sequenzen, Typklassen `Enum` und `Bounded`, List comprehensions