

# Deklarative Programmierung

## WS 23/24

Michael Hanus

10. Januar 2024

### Detaillierter Vorlesungsverlauf

**25.10.** Organisatorisches;

**Einführung in die funktionale Programmierung:** Variablenbegriff, Programm, Funktionsdefinitionen, Ausdrücke, Beispiel `square`, Beispiele `min/fac`, Auswertungsmöglichkeiten

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

**1.11.** Operatoren, polymorphe Funktionen (`length`, `++`), `last`), Definition polymorpher algebraischer Datentypen, `Maybe` und `maybeHead`, Binärbäume, `String`, Vereinigungstypen (`Either`), Tupel (`fst`, `snd`), Funktionen `zip`

**6.11.** Funktion `unzip`, Testen mit QuickCheck: Eigenschaften, `==>`, Referenzimplementierung, Regressionstests, automatisierte Ausführung aller Tests, Tests mit polymorphen Funktionen, Pattern Matching (Patternaufbau, case-Ausdrücke), Guards

**8.11.** Funktionen höherer Ordnung, anonyme Funktionen, partielle Applikation, Currying, Sections, Funktion `flip`, generische Programmierung (`map`, `foldr`, `filter`, `foldl`)

**13.11.** 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)

**15.11.** 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

**22.11.** Idee der Ein-/Ausgabe, I/O-Aktionen, Aktionen zur Ein- und Ausgabe, do-Notation, Beispiel Ausgabe von Zwischenergebnissen, I/O-Aktionen zum Lesen und Schreiben von Dateien, Zeilen einer Datei numeriert ausgeben

- 27.11. Module, Exportdeklarationen, Importdeklarationen, Erzeugung von ausführbaren Programmen mittels `ghc`; Transformationen auf Containerstrukturen: `Functor`, `fmap`, `Functor`-Gesetze, Transformationen mit beliebigen Funktionen: `pure`, `<*>`, `Applicative`
- 29.11. Arithmetische Ausdrücke und deren Auswertung, `Applicative`-Gesetze, `Applicative`-Instanzen für Listen und `IO`, effektvolle Berechnungen, Verbesserung der Auswertung arithmetischer Ausdrücke durch monadische Struktur, Klasse `Monad`, `Monad`-Instanz für Listen, Monadengesetze
- 4.12. QuickCheck: Eingabeklassifikation mit `classify` und `collect`, eigene Definitionen von Testdaten: Klasse `Arbitrary`, `elements`, `choose`, `oneof`, `sized`, `vector`, Fallstudien Peano-Arithmetik, `frequency`
- 6.12. **Einführung in die Logikprogrammierung:** Motivation, Verwandtschaftsbeispiel, Prolog-Programme, Fakten, Regeln, Anfragen, Prolog-Syntax (Zahlen, Atome, Strukturen, Listen), Variablen, Rechnen mit Listenstrukturen Operatoren, Gleichheit von Termen
- 11.12. Programmiertechnik Aufzählung des Suchraumes (Färben einer Landkarte, Sortieren von Zahlenlisten), Programmiertechnik Musterorientierte Wissensrepräsentation (`append`), Verwendung von Relationen, Peano-Zahlen (Definition, Addition, Subtraktion, Multiplikation)
- 13.12. Rechnen in der Logikprogrammierung: einfaches Resolutionsprinzip, Substitution, Unifikator, `mgu`, `disagreement sets`, Unifikationsalgorithmus, `occur check`, Komplexität, allgemeines Resolutionsprinzip (SLD-Resolution)
- 18.12. Auswertungsstrategie und SLD-Baum, Beweisstrategie von Prolog, Endlosschleifen, Negation als Fehlschlag, Probleme der Negation, verzögerte Negation mit Coroutining; `Cut`-Operator, Fallunterscheidung, Arithmetik in Prolog (`is`, Fakultätsfunktion)
- 20.12. arithmetische Constraints, Beispiel Schaltkreisanalyse, Beispiel Hypothekenberechnung, Constraint-Programmierung über endlichen Bereichen, allgemeines Vorgehen, `send-more-money`-Beispiel, 8-Damen-Problem, verbesserte Labeling-Strategien
- 8.1. Meta-Programmierung: Prädikate höherer Ordnung (`call`, `maplist`), Kapselung des Nicht-determinismus (`findall`, `bagof`, `setof`), Veränderung der Wissensbasis (`assert`, `retract`), Meta-Interpreter zur Beweislängenberechnung, Prädikate zur Ein- und Ausgabe von Daten
- 10.1. Tracing von Prolog-Programmen, Differenzlisten, Definite Clause Grammars, Kurzüberblick zu Multiparadigmen-Sprachen, Einführung in Curry (Verwandtschaftsbeispiel), funktionale Muster (`last`, `perm`), bedarfsgesteuerte Suche in Curry