Exploring Languages with Interpreters and Functional Programming 2018 Version

H. Conrad Cunningham

14 February 2019

Professor of Computer and Information Science
University of Mississippi
211 Weir Hall
P.O. Box 1848
University, MS 38677
(662) 915-5358

Browser Advisory: The HTML version of this document may require use of a browser that supports the display of MathML. A good choice as of January 2019 is a recent version of Firefox from Mozilla.

Exploring Languages with Interpreters and Functional Programming

Chapter 0: Preface

- Chapter: [HTML] [PDF]
- Slides: NONE

Chapter 1: Evolution of Programming Languages

- Chapter: [HTML] [PDF]
- Slides:
  - (HTML) Evolving Computer Hardware Affects Programming Languages
  - (HTML) History of Programming Languages

Chapter 2: Programming Paradigms

- Chapter: [HTML] [PDF]
- Slides: [(HTML) Programming Paradigms]
  [(HTML) Programming Paradigms Scala Version]

Chapter 3: Object-Based Paradigms

- Chapter: [HTML] [PDF]
- Slides: [(HTML) Object-Based Paradigms]

Chapter 4: First Haskell Programs

- Chapter: [HTML] [PDF]
- Slides: [(HTML) First Haskell Functions]

Chapter 5: Types

- Chapter: [HTML] [PDF]
- Slides:
  - [(HTML) Type System Concepts]
  - No slides yet for 5.3

Chapter 6: Procedural Abstraction

- Chapter: [HTML] [PDF]
- Slides:
  - [(HTML) Top-Down Stepwise Refinement]
  - [(HTML) Modular Design and Programming]
- Other: [(Haskell) Quick Overview of Basic Haskell]

Chapter 7: Data Abstraction

- Chapter: [HTML] [PDF]
- Slides: [(HTML) Using Data Abstraction]

Chapter 8: Evaluation Model

- Chapter: [HTML] [PDF]
- Slides: [(HTML) Evaluation of Functional Programs]

Chapter 9: Recursion Styles and Efficiency

- Chapter: [HTML] [PDF]
- Slides: [(HTML) Recursion Styles]

Chapter 10: Simple Input and Output (FUTURE)

- [HTML] [PDF]
- Slides: NONE YET

Chapter 11: Software Testing Concepts

- Chapter: [HTML] [PDF]
- Slides: NONE YET

Chapter 12: Testing Haskell Programs

- Chapter: [HTML] [PDF]
- Slides: NONE YET

Chapter 13: List Programming

- Chapter: [HTML] [PDF]
- Slides: (HTML) List Programming

Chapter 14: Infix Operators and List Examples

- Chapter: [HTML] [PDF]
- Slides: (HTML) Infix Operations and List Examples

Chapter 15: Higher-Order Functions

- Chapter: [HTML] [PDF]
- Slides: (HTML) Higher-Order List Programming

Chapter 16: Haskell Function Concepts

- Chapter: [HTML] [PDF]
- Slides: (HTML) Haskell Function Concepts

Chapter 17: Higher-Order Function Examples

- Chapter: [HTML] [PDF]
- Slides: (HTML) Higher Order Function Examples
- Wally World POP Project: [HTML] [PDF]
  [WWMPOP_skeleton.hs]

Chapter 18: More List Processing

- Chapter: [HTML] [PDF]
- Slides: (HTML) More List Processing

Chapter 19: Systematic Generalization

- Chapter: [HTML] [PDF]
- Slides: NONE YET

Chapter 20: Problem Solving

- Chapter: [HTML] [PDF]
- Slides: NONE YET

Chapter 21: Algebraic Data Types

- Chapter: [HTML] [PDF]
- Slides: (HTML) Algebraic Data Types
- Carrie's Candy Bowl Project: [HTML] [PDF]
  [CandyBowl_skeleton.hs]
  [[Candy Bowl ADT Semantics (Spring 2017) }(<../notes/CandyBowl/Lua/candybowl_semantics.html>)]
- Exam DSL Project: [HTML] [PDF]
  [ExamDSL_base.hs]
  [SimpleHTML.hs]
- SandwichDSL Project: [HTML] [PDF]
  [SandwichDSL_base.hs]
- Domain Specific Languages

Chapter 22: Overloading and Type Classes

- Chapter: [HTML] [PDF]
- Slides: (HTML) Overloading and Type Classes

- Movable Objects case study:
  [MovableObjects module]
  [NamedObjects module]
  [NamedMovableObjects module]
  [NamedMovableTest module skeleton]

Chapter 23: Data Abstraction Revisited

- Chapter: [HTML] [PDF]
- Slides: NONE YET

Chapter 24: Type Inference

- Chapter: [HTML] [PDF]
- Slides: NONE YET

Chapter 25: Proving Haskell Laws

- Chapter: [HTML] [PDF]
- Slides: NONE YET

Chapter 26: Program Synthesis

- Chapter: [HTML] [PDF]
- Slides: NONE YET

Chapter 27: Text Processing

- Chapter: [HTML] [PDF]
- Slides: NONE YET

Chapter 28: FUTURE

- Chapter: [HTML] [PDF]
- Slides: NONE YET
- Likely drawn from: Chapter 13, Models of Reduction *Notes on Functional Programming with Haskell*

Chapter 29: Divide and Conquer Algorithms

- Chapter: [HTML] [PDF]
- Slides: NONE YET

Chapter 30: Infinite Data Structures

- Chapter: [HTML] [PDF]
- Slides: NONE YET
- Strictness and Laziness (Scala)

Chapter 40: Language Processing (FUTURE)

- Chapter: [HTML] [PDF]
- Slides: NONE YET

Chapter 41: Calculator: Concrete Syntax

- Chapter: [HTML] [PDF]
- Slides: (HTML) Calculator Concrete Syntax

Chapter 42: Calculator: Abstract Syntax & Evaluation

- Chapter: [HTML] [PDF]
- Slides: (HTML) Calculator: Abstract Syntax & Evaluation

Chapter 43: Calculator: Modular Structure

- Chapter: [HTML] [PDF]
- Slides: (HTML) Calculator: Modular Structure

Chapter 44: Calculator: Parsing

- Chapter: [HTML] [PDF]
- Slides: (HTML) Calculator: Parsing

Chapter 45: Parser Combinators

- Chapter: [HTML] [PDF]
- Slides: None Yet

Chapter 46: Calculator: Compilation

- Chapter: [HTML] [PDF]
- Slides: None Yet

Chapter 80: (Appendix) Review of Relevant Mathematics

- Chapter: [HTML] [PDF]
- Slides: NONE YET

OLD Chapter 13: Imperative Core Language

UNDER DEVELOPMENT

- Chapter: Imperative Core Language Case Study (does not exist yet as of 14 Juy 2018)
- Interpreter modules (prefix syntax) code mostly works but needs a bit of update to match recent changes to Expression Language
  - REPL module
  - Recursive descent parser module
  - Lexical analyzer module
  - Abstract Syntax module
  - Evaluator module
  - Environments module
  - Values module
  - Test Imp Core(in work, not current)

OLD Chapter 4: List Programming Supplements

Most of the content of this old chapter went into new chapters 13 and 14, but some was moved to earlier chapters.

- Possible future material on modular programming based on:
  - Lecture Notes on Modular Design
  - Lecture Notes on Data Abstraction
  - Cookie Jar ADT Problem Description (Scala)

Future Chapter? Using Algebraic Data Types

- TBD: Regular Expressions using algebraic data types
- Framework Design Using Function Generalization: A Binary Tree Traversal Case Study

Future Chapter? Domain Specific Languages

- Domain Specific Languages
- Sandwich DSL Case Study
  - Haskell version
  - Scala version
  - old Lua version

Future Chapters? Games

- Wizard's Adventure game (Elixir)
- Dice of Doom (Elixir)

Acknowledgements

I began this effort in Summer 2016 by adapting previous materials from my courses on Functional Programming (primarily), Multiparadigm Programming, Object-Oriented Programming, Software Architecture, Software Families, and Software Language Engineering.

I added new materials in Spring and Summer 2017 to draft the 2017 version of the textbook titled *Introduction to Functional Programming Using Haskell*.

In Spring and Summer 2018, I began work on an updated 2018 version of the textbook, now titled *Exploring Languages with Interpreters and Functional Programming*. I broke several of the longer chapters into 2-4 new chapters or appendices. I incorporated new material from my Spring 2018 Software Language Engineering class (e.g. Type Concepts). I also wrote new chapters including the two new chapters on Software Testing.

I maintain this textbook as text files in Pandoc's dialect of Markdown using embedded LaTeX markup for the mathematical formulas and then translate the documents to HTML, PDF, and other formats as needed.