Schedule

A week-by-week breakdown of the material.

Overview

- Introduction to Haskell and Fuctional Programming (1.1-1.14)
- Compiler and interpreter (2.1-2.7)
- Basic types (3.1-3.7)
- Designing and writing programs (4.1-4.8)
- Tuples and lists (5.1-5.7)
- More programming with lists (6.1-6.8)
- Defining functions over lists (7.1-7.6)
- Input and output in Haskell (8.1-8.6)
- Patterns of computation (10.1-10.5)
- Higher-order functions (11.1-11.5)
- Developing higher-order programs (12.1-12.7)
- Overloading and type-classes (13.1-13.8)
- Algebraic types (14.1-14.6)
- Case study: Huffman codes (15.1-15.7)
- Abstract data types (16.1-16.9)
- Lazy evaluation (17.1-17.8)
- I/O programming and Monads (18.1-18.6)

Week 1

Mon
- Reading: 1.1-1.11. Optional: 1.12-1.14
- Setting up[1]
- Introduction to Haskell and Fuctional Programming[2]

Wed
- Reading: 2.1-2.7

[1] notes/setup.html
[2] notes/intro.html

- Commands for GHCi interactive mode[3]
- Practice with GHCi[4]

Fri
- Reading: 3.1-3.7
- Standard Haskell values and types.[5]
- Conditionals. Guarded Expressions.[6]
- Assignment 0. Due Wed 01/15[7]

Week 2

Mon
- Reading: 4.1-4.3, 4.8, 5.1-5.3
- Compound Types[8]
- Type Aliases and Custom Types.[9]

Wed
- Reading: 5.4-5.7
- Lists, list comprehensions
- Assignment 1. Due Mon 01/20[10]

Fri TBD

Week 3

Mon TBD

Wed TBD

Fri TBD

Week 4

Mon TBD

Wed TBD

Fri TBD

[3] notes/ghci_commands.html
[4] notes/ghci_practice.html
[5] notes/standard.html
[6] notes/functions_conditionals.html
[7] assignments/assignment0.html
[8] notes/compoundTypes.html
[9] notes/types_custom.html
[10] assignments/assignment1.html

Week 5

Mon TBD

Wed TBD

Fri TBD

Week 6

Mon TBD

Wed TBD

Fri TBD

Week 7

Mon TBD

Wed TBD

Fri TBD

Week 8

Mon TBD

Wed TBD

Fri TBD

Week 9

Mon TBD

Wed TBD

Fri TBD

Week 10

Mon TBD

Wed TBD

Fri TBD

Week 11

Mon TBD

Wed TBD

Fri TBD

Week 12

Mon TBD

Wed TBD

Fri TBD

Week 13

Mon TBD

Wed TBD

Fri TBD

Old links

- Working with the GHC compiler and interpreter. Lists.[11] (2.1-2.5)
- More advanced typing: Curried Functions. Polymorphism, Type classes.[12] (3.6-3.9)
- More advanced typing: Curried Functions. Polymorphism, Type classes. (cont)[13] (3.6-3.9)
- Pattern Matching.[14] (4.4)
- More practice with Pattern Matching.[15]
- Version Control[16]
- Assignment 2. Due 09/29[17]

[11] notes/lists.html
[12] notes/types_advanced.html
[13] notes/types_advanced.html
[14] notes/pattern_matching.html
[15] notes/more_pattern_matching.html
[16] notes/version_control.html
[17] assignments/assignment2.html

- Recursion[18] (6.1-6.6)
- Anonymous Functions. Sections.[19] (4.5-4.6)
- Assignment 3. Due 10/13[20]
- The Maybe (Option) Type.[21]
- List Comprehensions.[22] (5.1-5.4)
- Functions as Values: Difference Lists, Composition[23] (7.5)
- Functions as Values: Difference Lists, Composition (cont)[24] (7.5)
- MIDTERM (study guide[25])
- Interactive Programming[26] (10.1-10.5)
- Practice with Interactive Programming[27] (10.6)
- BREAK
- Recursive Types[28] (8.4)
- Assignment 4. Due 11/03[29]
- Folding[30] (7.3-7.4)
- Overview of Software Development Practices[31]
- Information hiding and abstraction with modules[32]
- Testing[33]
- The State Monad[34]
- Functors, Applicatives, Monads[35]
- Specification Testing with HSpec[36]

[18] notes/recursion.html
[19] notes/anonymous_functions.html
[20] assignments/assignment3.html
[21] notes/maybe.html
[22] notes/list_comprehensions.html
[23] notes/difference_lists.html
[24] notes/difference_lists.html
[25] notes/midterm_study_guide.html
[26] notes/interactive.html
[27] notes/interactive_hangman.html
[28] notes/recursive_types.html
[29] assignments/assignment4.html
[30] notes/folding.html
[31] notes/dev_overview.html
[32] notes/modules.html
[33] notes/testing.html
[34] notes/functors_monads.html
[35] notes/functors_monads.html
[36] notes/testing_hspec.html

- Final Study Guide[37]

[37] notes/final_study_guide.html