Parallel Functional Programming

Max Levatich & Stephen A. Edwards

Columbia University

Fall 2025

Instructor

Maxwell Levatich, 6th year PhD candidate

Advisor: Stephen A. Edwards

ml4553@columbia.edu

Office Hours: Tuesday 1:00 - 3:00pm, 468 CSB

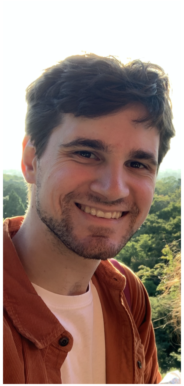
Instructor

Previously...

- **Took** PFP
- **TA’d** for PFP (+ 8 other CS classes)
- Lots of Haskell experience

Same syllabus (mostly), same assignments (mostly)

Consulting with Stephen on teaching strategy and course material



Haskell

```
primes = filterPrime [2..]
  where filterPrime (p:xs) =
          p : filterPrime [x | x <- xs, x `mod` p /= 0]
```

Sieve of Eratosthenes

Purely Functional · Declarative · Lazy · Statically Type-Inferred · Parallel

Sequential Haskell in the first half · Parallel in the second half

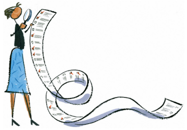
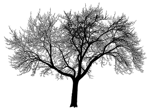
Why take PFP?

- FP in the classroom = on-ramp to all functional languages
  - Broader skillset = more job opportunities
- Practice *thinking functionally* about computation
  - You will write better, more correct programs
  - You will be a better person (?)
- Parallel concepts: synchronization, load balancing, determinism. . .
  - Very hard!
  - Yet modern machines are multi-core; single-threaded is retro!

Prerequisites

Data structures (COMS W3134, W3137, or equivalent)

- You must be fluent in at least one programming language
- You must dream about lists and trees

- You do not need prior experience in a *functional* programming language; that’s what this course is for

Assignments and Grading

60 %	Homework assignments
10 %	In-class quizzes
30 %	Final Project (alone or in pairs)

This is a coding† class

The homework must be your own code

The project may be done alone or in a group

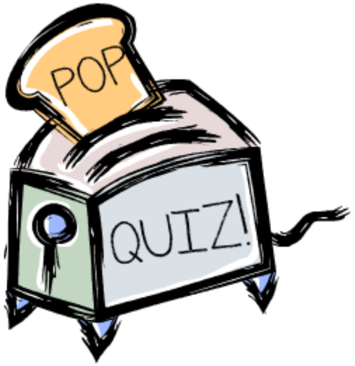
†More precisely, mostly debugging,
with a little bit of bugging

Quizzes

Wednesdays during last 10min of lecture, starting Sep 17

- Pencil-and-paper, no devices
- One question, with single-word or numeric answer
- Should be “easy” if you are attending lecture, I will adjust if not
- Lowest three dropped

Meant to be low-stress. Just a slight nudge to encourage you not to fall behind on material!



Your Resources

Course site: `www.cs.columbia.edu/~sedwards/classes/2025/4995-fall`

- Syllabus, schedule, assignments, course policies, office hours

Courseworks: `courseworks2.columbia.edu/courses/227047`

- Course announcements, lecture recordings, submissions, grades

Ed discussion: `edstem.org/us/courses/81304/discussion`

- Discussion forum, homework questions, TA announcements

Still can’t find the info you need? **Email me!**

Collaboration

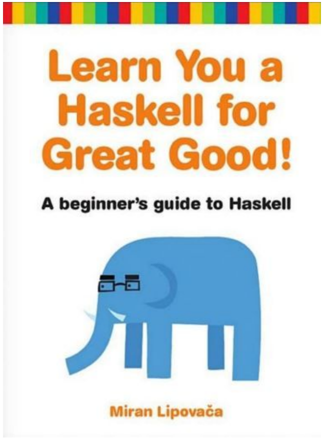
You may seek outside help, including from other students, on homework, but

- You must write all of your own code. No copying or copying-with-modification of any code. No looking at other student's code as reference as you write your own. No consulting AI solutions.
- You must cite all people and resources you consulted. For example, you might add a comment like

```
{- I collaborated with Haskell Curry, Jim Backus, Alonzo Church,
   and Grace Hopper on this assignment, and consulted
http://hackage.haskell.org/package/base-4.12.0.0/docs/Data-List.html
https://stackoverflow.com/questions/211216
http://www.cis.upenn.edu/~cis194/fall16/policies.html
-}
```

See also `https://www.cs.columbia.edu/academic/academic-honesty/`

Recommended Texts

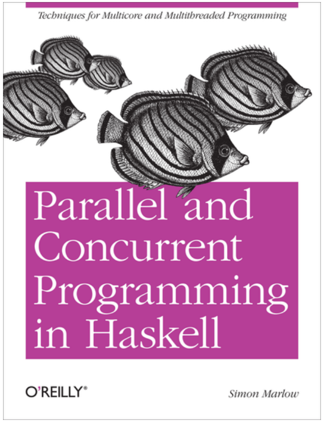

Learn You a Haskell for Great Good!
A beginner's guide to Haskell
Miran Lipovača

Miran Lipovača.
Learn You a Haskell for Great Good!
No Starch Press, 2001.

`https://learnyouahaskell.github.io/`

Excellent introductory text. We will be following it for roughly the first half of the class.

Recommended Texts

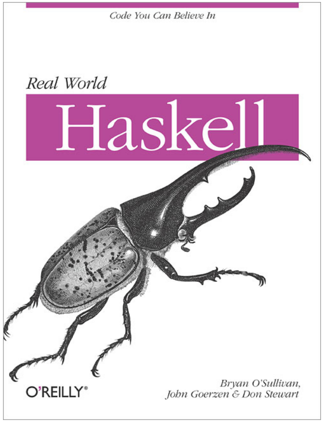

Techniques for Multicore and Multithreaded Programming
Parallel and Concurrent Programming in Haskell
O'REILLY
Simon Marlow

Simon Marlow.
Parallel and Concurrent Programing in Haskell.
O'Reilly, 2013.

`https://simonmar.github.io/pages/pcph.html`

Like its title says. Assumes a reasonable understanding of Haskell. We will be following it for the second half of the class.

Recommended Texts

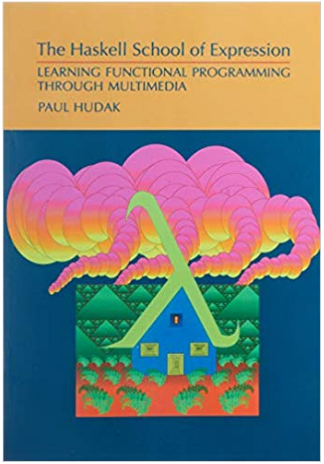


Bryan O'Sullivan, Don Stewart, and John Goerzen.
Real World Haskell.
O'Reilly, 2009.

`http://book.realworldhaskell.org/`

Also an introductory text on Haskell that starts at the beginning, it quickly focuses on practical, real-world aspects of writing Haskell programs, such as elaborate I/O, and interfacing with external libraries.

Recommended Texts

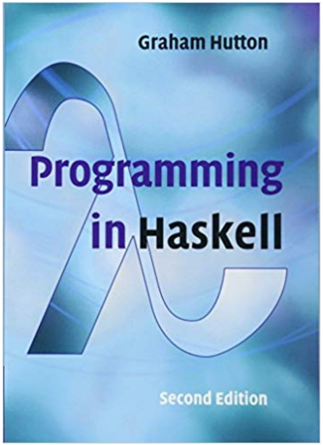

The Haskell School of Expression
LEARNING FUNCTIONAL PROGRAMMING THROUGH MULTIMEDIA
PAUL HUDAK

Paul Hudak.
The Haskell School of Expression.
Cambridge University Press, 2000.

`http://www.cs.yale.edu/homes/hudak/SOE/`

An idiosyncratic approach to learning Haskell based on multimedia (graphics, animation, and sound) ultimately leading to domain-specific langauges.

Recommended Texts

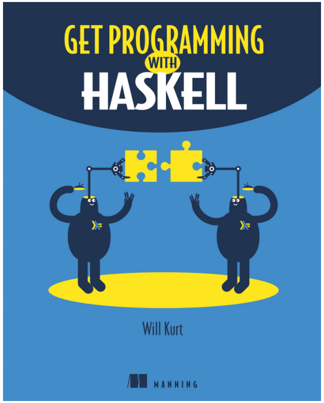


Graham Hutton.
Programming in Haskell.
Second Edition, Cambridge University Press, 2016.

`http://www.cs.nott.ac.uk/~pszgmh/pih.html`

Another introductory Haskell text, this one written by a professor from the University of Nottingham

Recommended Texts



Will Kurt.
Get Programming with Haskell.
Manning, 2018.

`https://www.manning.com/books/get-programming-with-haskell`

Another introductory Haskell text, written more like a textbook