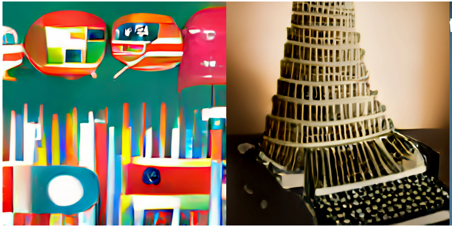
Types, Languages, and Compilers

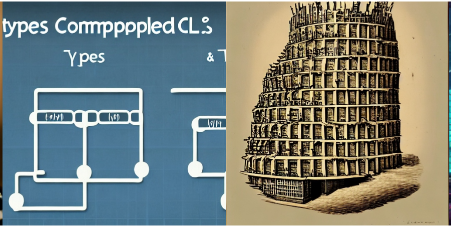
Stephen A. Edwards

Columbia University

Spring 2023



craiyon.com



deepai.org

hotpot.ai

Instructor

Prof. Stephen A. Edwards

sedwards@cs.columbia.edu

http://www.cs.columbia.edu/~sedwards/

462 Computer Science Building

Email me for appointments, or just come by

The Hindley-Milner Type System

$$
\begin{aligned}
e &::= x \mid e\ e \mid \lambda\ x\ .\ e \mid \mathbf{let}\ x = e\ \mathbf{in}\ e \\
\tau &::= \alpha \mid C\ \tau \ldots \tau \mid \tau \longrightarrow \tau \\
\sigma &::= \tau \mid \forall\ \alpha\ .\ \sigma
\end{aligned}
$$

$$\frac{x : \sigma \in \Gamma}{\Gamma \vdash x : \sigma}\,\text{Var} \qquad \frac{\Gamma \vdash e_0 : \tau \longrightarrow \tau' \qquad \Gamma \vdash e_1 : \tau}{\Gamma \vdash e_0\ e_1 : \tau'}\,\text{App}$$

$$\frac{\Gamma, x : \tau \vdash e : \tau'}{\Gamma \vdash \lambda\ x.e : \tau \longrightarrow \tau'}\,\text{Abs} \qquad \frac{\Gamma \vdash e_0 : \sigma \qquad \Gamma, x : \sigma \vdash e_1 : \tau}{\Gamma \vdash \mathbf{let}\ x = e_0\ \mathbf{in}\ e_1 : \tau}\,\text{Let}$$

$$\frac{\Gamma \vdash e : \sigma' \qquad \sigma' \sqsubseteq \sigma}{\Gamma \vdash e : \sigma}\,\text{Inst} \qquad \frac{\Gamma \vdash e : \sigma \qquad \alpha \notin \text{free}(\Gamma)}{\Gamma \vdash e : \forall \alpha.\sigma}\,\text{Gen}$$

Syllabus

Constructive Logic and Inductive Definitions

Regular Expressions

Context-Free Grammars

The Lambda Calculus

The Simply-Typed Lambda Calculus

The Hindley-Milner Type System

Operational Semantics

Dependent Types, Proof Assistants, and the Curry-Howard Correspondence

Prerequisites

- COMS 3203 Discrete Mathematics

  Sets Functions Relations Theorems Proofs Induction Logic

- COMS 3261 Computer Science Theory

  Alphabets, Strings, & Languages Regular Langauges Context-Free Grammars

- COMS 4115 Programming Languages and Translators

  ASTs Context-Free Grammars Parsing Algorithms Type Checking

- Experience with Haskell or another functional language

  OCaml SML Scala Scheme

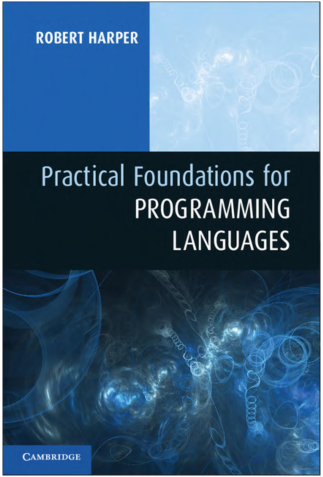
Assignments and Grading

30 %	Homework assignments
70 %	Final Project (in pairs)

Homework assignments will be a mix of Haskell and mathematics

Do the project in pairs

Recommended Texts

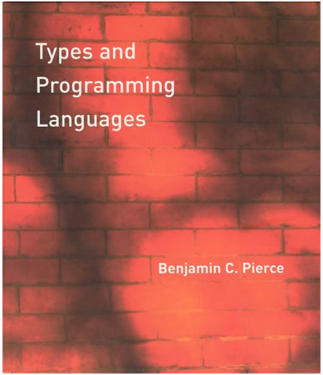


Robert Harper.
Practical Foundations for Programming Languages.
2nd ed. Cambridge University Press, 2016.

`http://www.cs.cmu.edu/~rwh/pfpl.html`

Harper is closest in scope to the class, but we will not follow it slavishly. In particular, it has a lot of detail that will not be discussed.

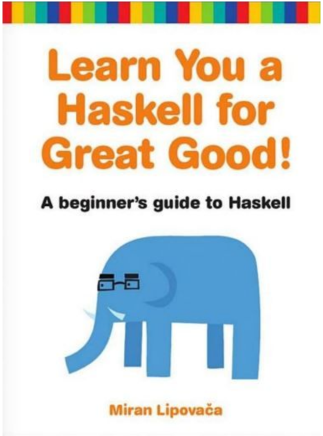
Recommended Texts



Benjamin C. Pierce.
Types and Programming Languages.
MIT Press, 2002.

Pierce is also an excellent reference for this course. As its name suggests, it focuses almost exclusively on types. Unlike Harper, it has extensive discussions of how to implement the discussed ideas in code.

Recommended Texts

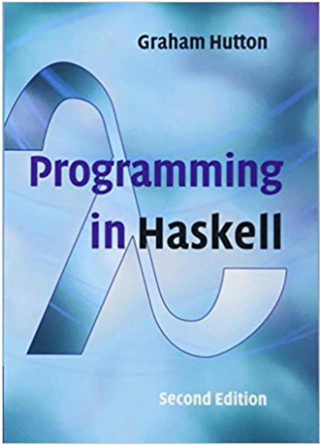

Learn You a Haskell for Great Good!
A beginner's guide to Haskell
Miran Lipovača

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

`http://learnyouahaskell.com/`

Excellent introductory text on Haskell

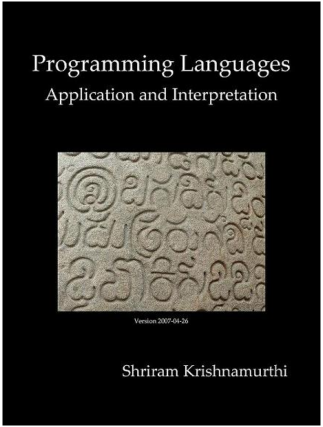
Recommended Texts



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

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

Recommended Texts

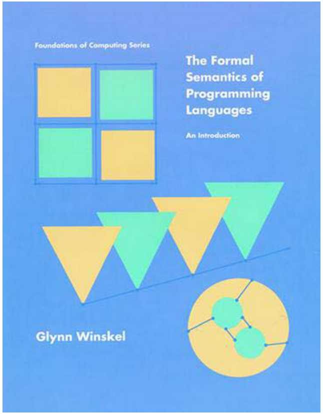


Shriram Krishnamurthi.
Programming Languages: Application and Interpretation.
2nd ed. Self-Published, 2017.

`https://cs.brown.edu/~sk/Publications/Books/ProgLangs/2007-04-26/`

Krishnamurthi uses Racket and is heavy on implementation details. It has some discussion of types and dynamic checking through contracts, but is closer to a compiler implementation text than the others listed here.

Recommended Texts


Foundations of Computing Series
The Formal Semantics of Programming Languages
An Introduction
Glynn Winskel

Glynn Winskel. *The Formal Semantics of Programming Languages: An Introduction.*
MIT Press, 1993.

Winskel is an older book that focuses, unsurprisingly, on programming language semantics (operational, denotational, and axiomatic), but also discusses types and the Lambda Calculus.