The Reasoned Schemer

A Patriot's History of the United States
Essentials of Programming Languages, third edition
Software Design for Flexibility
The Reasoned Schemer, second edition
The Craft of Prolog
Let Over Lambda
Logic for Applications
Slam Book Fever
The Practice of Prolog
Alices Adventures
Programming with Higher-Order Logic
The Formal Semantics of Programming Languages
Simply Scheme
Thinking Forth
A Book of Scoundrels
Scheme and the Art of Programming
The Reasoned Schemer, second edition
Adventure in Prolog
The Little Typer
Live Wires
The Reasoned Schemer
Semantics Engineering with PLT Redex
Adventures of Sherlock Holmes
The Investment Answer
Investigation of Failure of the SEC to Uncover

Bernard Madoff's Ponzi Scheme
Lisp in Small Pieces
Seven More Languages in Seven Weeks
The Seasoned Schemer, second edition
The Little LISPer
The Little MLer
Truth and Consequences
Father James Page
The Joy of Clojure
Types in Logic Programming
Realm of Racket
The Little Prover
Essentials of Programming Languages
Structure and Interpretation of Classical Mechanics, second edition
A Little Java, a Few Patterns
Sinking in the Swamp

ARIANA MCKEE

A Patriot's History of the United States
MIT Press
An introduction to dependent types, demonstrating the most beautiful aspects, one step at a time. A program's type describes its behavior. Dependent types are a first-class part of a language, and are much more powerful than other kinds of types; using just one language for types and programs allows program descriptions to be as powerful as the programs they describe. The Little Typer explains

dependent types, beginning with a very small language that looks very much like Scheme and extending it to cover both programming with dependent types and using dependent types for mathematical reasoning. Readers should be familiar with the basics of a Lisp-like programming language, as presented in the first four chapters of The Little Schemer. The first five chapters of The Little Typer provide the needed tools to understand dependent types; the remaining chapters use these tools to build a bridge between mathematics and programming. Readers will learn that tools they know from programming —pairs, lists, functions, and recursion—ca n also capture patterns of reasoning. The Little Typer does not attempt to teach either practical programming skills or a fully rigorous approach to types. Instead, it demonstrates the most beautiful aspects as simply as possible, one step at a time.

Essentials of Programming Languages, third edition
MIT Press
Racket is a descendant of Lisp, a programming language renowned for its elegance, power, and challenging learning curve. But while Racket retains the functional

goodness of Lisp, it was designed with beginning programmers in mind. Realm of Racket is your introduction to the Racket language. In Realm of Racket, you'll learn to program by creating increasingly complex games. Your journey begins with the Guess My Number game and coverage of some basic Racket etiquette. Next you'll dig into syntax and semantics, lists, structures, and conditionals, and learn to work with recursion and the GUI as you build the Robot Snake game. After that it's on to lambda and mutant structs (and an Orc Battle), and fancy loops and the Dice of Doom. Finally, you'll explore laziness, AI, distributed games, and the Hungry Henry game. As you progress through the games, chapter checkpoints and challenges help reinforce what you've learned. Offbeat comics keep things fun along the way. As you travel through the Racket realm, you'll: –Master the quirks of Racket's syntax and semantics –Learn to write concise and elegant functional programs –Create a graphical user interface using the 2htdp/image library –Create a server to handle true multiplayer games Realm of Racket is a

lighthearted guide to some serious programming. Read it to see why Racketeers have so much fun!

Software Design for Flexibility
Springer Science & Business Media
Showing off scheme - Functions - Expressions - Defining your own procedures - Words and sentences - True and false - Variables - Higher-order functions - Lambda - Introduction to recursion - The leap of faith - How recursion works - Common patterns in recursive procedures - Advanced recursion - Example : the functions program - Files - Vectors - Example : a spreadsheet program - Implementing the spreadsheet program - What's next?

The Reasoned Schemer, second edition
MIT Press
A new edition of a book, written in a humorous question-and-answer style, that shows how to implement and use an elegant little programming language for logic programming. The goal of this book is to show the beauty and elegance of relational programming, which captures the essence of logic programming. The book shows how to implement a relational programming language in Scheme, or in any other functional language, and demonstrates

the remarkable flexibility of the resulting relational programs. As in the first edition, the pedagogical method is a series of questions and answers, which proceed with the characteristic humor that marked The Little Schemer and The Seasoned Schemer. Familiarity with a functional language or with the first five chapters of The Little Schemer is assumed. For this second edition, the authors have greatly simplified the programming language used in the book, as well as the implementation of the language. In addition to revising the text extensively, and simplifying and revising the “Laws” and “Commandments,” they have added explicit “Translation” rules to ease translation of Scheme functions into relations.

The Craft of Prolog MIT Press

Let Over Lambda is one of the most hardcore computer programming books out there. Starting with the fundamentals, it describes the most advanced features of the most advanced language: Common Lisp. Only the top percentile of programmers use lisp and if you can understand this book you are in the top percentile of lisp programmers. If you are looking for a

dry coding manual that re-hashes common-sense techniques in whatever langue du jour, this book is not for you. This book is about pushing the boundaries of what we know about programming. While this book teaches useful skills that can help solve your programming problems today and now, it has also been designed to be entertaining and inspiring. If you have ever wondered what lisp or even programming itself is really about, this is the book you have been looking for.

Let Over Lambda MIT Press

For the past three decades, many history professors have allowed their biases to distort the way America's past is taught. These intellectuals have searched for instances of racism, sexism, and bigotry in our history while downplaying the greatness of America's patriots and the achievements of "dead white men." As a result, more emphasis is placed on Harriet Tubman than on George Washington; more about the internment of Japanese Americans during World War II than about D-Day or Iwo Jima; more on the dangers we faced from Joseph McCarthy than those we faced from Josef Stalin. A Patriot's

History of the United States corrects those doctrinaire biases. In this groundbreaking book, America's discovery, founding, and development are reexamined with an appreciation for the elements of public virtue, personal liberty, and private property that make this nation uniquely successful. This book offers a long-overdue acknowledgment of America's true and proud history.

Logic for Applications

MIT Press

In December 2008, the world watched as master financier Bernard L. Madoff was taken away from his posh Manhattan apartment in handcuffs, accused of swindling thousands of innocent victims-including friends and family-out of billions of dollars in the world's largest Ponzi scheme. Madoff went to jail; he will spend the rest of his life there. But what happened to his devoted wife and sons? The people closest to him, the public reasoned, must have known the truth behind his astounding success. Had they been tricked, too? With unprecedented access to the surviving family members -- wife Ruth, son Andrew and his fiancéee Catherine Hooper -- journalist Laurie Sandell reveals the personal

details behind the headlines. How did Andrew and Mark, the sons who'd spent their lives believing in and building their own families around their father's business first learn of the massive deception? How does a wife, who adored her husband since they were teenagers, begin to understand the ramifications of his actions? The Madoffs were a tight-knit and even claustrophobic clan, sticking together through marriages, divorces, and illnesses. But the pressures of enduring the massive scandal push them to their breaking points, most of all son Mark, whose suicide is one of the many tragedies that grew in the wake of the scandal. Muzzled by lawyers, vilified by the media and roundly condemned by the public, the Madoffs have chosen to keep their silence -- until now. Ultimately, theirs is one of the most riveting stories of our time: a modern-day Greek tragedy about money, power, lies, family, truth and consequences .

Slam Book Fever MIT Press

The new edition of a classic text that concentrates on developing general methods for studying the behavior of classical systems, with extensive use of

computation. We now know that there is much more to classical mechanics than previously suspected. Derivations of the equations of motion, the focus of traditional presentations of mechanics, are just the beginning. This innovative textbook, now in its second edition, concentrates on developing general methods for studying the behavior of classical systems, whether or not they have a symbolic solution. It focuses on the phenomenon of motion and makes extensive use of computer simulation in its explorations of the topic. It weaves recent discoveries in nonlinear dynamics throughout the text, rather than presenting them as an afterthought. Explorations of phenomena such as the transition to chaos, nonlinear resonances, and resonance overlap to help the student develop appropriate analytic tools for understanding . The book uses computation to constrain notation, to capture and formalize methods, and for simulation and symbolic analysis. The requirement that the computer be able to interpret any expression provides the student with strict and immediate feedback about whether an expression is correctly

formulated. This second edition has been updated throughout, with revisions that reflect insights gained by the authors from using the text every year at MIT. In addition, because of substantial software improvements, this edition provides algebraic proofs of more generality than those in the previous edition; this improvement permeates the new edition.

The Practice of Prolog

Рипол Классик

The first comprehensive presentation of reduction semantics in one volume, and the first tool set for such forms of semantics. This text is the first comprehensive presentation of reduction semantics in one volume; it also introduces the first reliable and easy-to-use tool set for such forms of semantics. Software engineers have long known that automatic tool support is critical for rapid prototyping and modeling, and this book is addressed to the working semantics engineer (graduate student or professional language designer). The book comes with a prototyping tool suite to develop, explore, test, debug, and publish semantic models of programming languages. With PLT Redex, semanticists can formulate models as grammars and reduction

models on their computers with the ease of paper and pencil. The text first presents a framework for the formulation of language models, focusing on equational calculi and abstract machines, then introduces PLT Redex, a suite of software tools for expressing these models as PLT Redex models. Finally, experts describe a range of models formulated in Redex. PLT Redex comes with the PLT Scheme implementation, available free at http://www.plt-scheme.org/. Readers can download the software and experiment with Redex as they work their way through the book.

Alices Adventures

MIT Press Predictions ... Slam books are the newest craze at Sweet Valley High. They're do-it-yourself books of lists and predictions about everyone in school. They start out as fun but soon stir up big trouble. First, Jeffrey French, Elizabeth Wakefield's boyfriend, gets paired up with another girl under the category, "Couple of the Future." Then Elizabeth gets matched with the new boy at school, A.J. Morgan--and her twin, Jessica, is furious, because she's the one who's fallen hard for A.J. Will the mysterious slam-book entries spell

the end of happiness for both Elizabeth and Jessica Wakefield?

Programming with Higher-Order Logic MIT Press

What if there were a way to cut through all the financial mumbo-jumbo? Wouldn't it be great if someone could really explain to us-in plain and simple English-the basics we must know about investing in order to insure our financial freedom? At last, here's good news. Jargon-free and written for all investors-experienced, beginner, and everyone in between-The Investment Answer distills the process into just five decisions-five straightforward choices that can lead to safe and sound ways to manage your money. When Wall Street veteran Gordon Murray told his good friend and financial advisor, Dan Goldie, that he had only six months to live, Dan responded, "Do you want to write that book you've always wanted to do?" The result is this eminently valuable primer which can be read and understood in one sitting, and has advice that benefits you, not Wall Street and the rest of the traditional financial services industry. The Investment Answer asks readers to make five basic but key decisions to stack the

investment odds in their favor. The advice is simple, easy-to-follow, and effective, and can lead to a more profitable portfolio for every investor. Specifically: Should I invest on my own or seek help from an investment professional? How should I allocate my investments among stocks, bonds, and cash? Which specific asset classes within these broad categories should I include in my portfolio? Should I take an actively managed approach to investing, or follow a passive alternative? When should I sell assets and when should I buy more? In a world of fast-talking traders who believe that they can game the system and a market characterized by instability, this extraordinary and timely book offers guidance every investor should have.

The Formal Semantics of Programming Languages Johns Hopkins University Press

The emphasis in The Craft of Prolog is on using Prolog effectively. It presents a loose collection of topics that build on and elaborate concepts learned in a first course. Hacking your program is no substitute for understanding your problem. Prolog is different, but not that different. Elegance is not optional. These are the themes that

unify Richard O'Keefe's very personal statement on how Prolog programs should be written. The emphasis in The Craft of Prolog is on using Prolog effectively. It presents a loose collection of topics that build on and elaborate concepts learned in a first course. These may be read in any order following the first chapter, "Basic Topics in Prolog," which provides a basis for the rest of the material in the book. Richard A. O'Keefe is Lecturer in the Department of Computer Science at the Royal Melbourne Institute of Technology. He is also a consultant to Quintus Computer Systems, Inc.Contents: Basic Topics in Prolog. Searching. Where Does the Space Go? Methods of Programming. Data Structure Design. Sequences. Writing Interpreters. Some Notes on Grammar Rules. Prolog Macros. Writing Tokenisers in Prolog. All Solutions.

Simply Scheme MIT Press

Contents: (1) Results of the Invest.; (2) SEC Review of 2000 and 2001 Markopolos Complaints: (3) SEC 2004 OCIE Cause Exam. of Madoff; (4) SEC 2005 NERO Exam. of Madoff; (5) SEC 2006 Invest. of Markopolos Complaint; (6) Effect of Madoff¿s Stature and Reputation on

SEC Exam.; (7) Allegations of Conflict of Interest from the Relationship between Eric Swanson and Shana Madoff; (8) Private Entities¿ Due Diligence Efforts Revealed Suspicious Activity about Madoff¿s Operations; (9) Potential Investors Relied upon the Fact That the SEC had Examined and Investigated Madoff in Making Decisions to Invest with Him; (10) Additional Complaints Received by the SEC re: Madoff; (11) Additional Exam. and Inspect. of Madoff¿s Firms by the SEC.

Thinking Forth MIT Press

A new edition of a book, written in a humorous question-and-answer style, that shows how to implement and use an elegant little programming language for logic programming. The goal of this book is to show the beauty and elegance of relational programming, which captures the essence of logic programming. The book shows how to implement a relational programming language in Scheme, or in any other functional language, and demonstrates the remarkable flexibility of the resulting relational programs. As in the first edition, the pedagogical method is a series of questions and answers, which proceed

with the characteristic humor that marked The Little Schemer and The Seasoned Schemer. Familiarity with a functional language or with the first five chapters of The Little Schemer is assumed. For this second edition, the authors have greatly simplified the programming language used in the book, as well as the implementation of the language. In addition to revising the text extensively, and simplifying and revising the “Laws” and “Commandments,” they have added explicit “Translation” rules to ease translation of Scheme functions into relations.

A Book of Scoundrels

Penguin Thinking Forth applies a philosophy of problem solving and programming style to the unique programming language Forth. Published first in 1984, it could be among the timeless classics of computer books, such as Fred Brooks' The Mythical Man-Month and Donald Knuth's The Art of Computer Programming. Many software engineering principles discussed here have been rediscovered in eXtreme Programming, including (re)factoring, modularity, bottom-up and incremental design. Here you'll find all of those and more, such as

the value of analysis and design, described in Leo Brodie's down-to-earth, humorous style, with illustrations, code examples, practical real life applications, illustrative cartoons, and interviews with Forth's inventor, Charles H. Moore as well as other Forth thinkers.

Scheme and the Art of Programming No Starch Press

This is a comprehensive account of the semantics and the implementation of the whole Lisp family of languages, namely Lisp, Scheme and related dialects. It describes 11 interpreters and 2 compilers, including very recent techniques of interpretation and compilation. The book is in two parts. The first starts from a simple evaluation function and enriches it with multiple name spaces, continuations and side-effects with commented variants, while at the same time the language used to define these features is reduced to a simple lambda-calculus. Denotational semantics is then naturally introduced. The second part focuses more on implementation techniques and discusses precompilation for fast interpretation: threaded code or bytecode; compilation towards C. Some extensions are also described such as dynamic

evaluation, reflection, macros and objects. This will become the new standard reference for people wanting to know more about the Lisp family of languages: how they work, how they are implemented, what their variants are and why such variants exist. The full code is supplied (and also available over the Net). A large bibliography is given as well as a considerable number of exercises. Thus it may also be used by students to accompany second courses on Lisp or Scheme.

The Reasoned Schemer, second edition

Pragmatic Bookshelf

Great programmers aren't born--they're made. The industry is moving from object-oriented languages to functional languages, and you need to commit to radical improvement. New programming languages arm you with the tools and idioms you need to refine your craft. While other language primers take you through basic installation and "Hello, World," we aim higher. Each language in Seven More Languages in Seven Weeks will take you on a step-by-step journey through the most important paradigms of our time. You'll learn seven exciting languages: Lua, Factor, Elixir, Elm,

Julia, MiniKanren, and Idris. Learn from the award-winning programming series that inspired the Elixir language. Hear how other programmers across broadly different communities solve problems important enough to compel language development. Expand your perspective, and learn to solve multicore and distribution problems. In each language, you'll solve a non-trivial problem, using the techniques that make that language special. Write a fully functional game in Elm, without a single callback, that compiles to JavaScript so you can deploy it in any browser. Write a logic program in Clojure using a programming model, MiniKanren, that is as powerful as Prolog but much better at interacting with the outside world. Build a distributed program in Elixir with Lisp-style macros, rich Ruby-like syntax, and the richness of the Erlang virtual machine. Build your own object layer in Lua, a statistical program in Julia, a proof in code with Idris, and a quiz game in Factor. When you're done, you'll have written programs in five different programming paradigms that were

written on three different continents. You'll have explored four languages on the leading edge, invented in the past five years, and three more radically different languages, each with something significant to teach you.

Adventure in Prolog MIT Press

The Formal Semantics of Programming Languages provides the basic mathematical techniques necessary for those who are beginning a study of the semantics and logics of programming languages. These techniques will allow students to invent, formalize, and justify rules with which to reason about a variety of programming languages. Although the treatment is elementary, several of the topics covered are drawn from recent research, including the vital area of concurency. The book contains many exercises ranging from simple to miniprojects.Starting with basic set theory, structural operational semantics is introduced as a way to define the meaning of programming languages along with associated proof techniques. Denotational and axiomatic semantics are illustrated on a simple language of while-programs, and fall proofs are given of the equivalence of the operational

and denotational semantics and soundness and relative completeness of the axiomatic semantics. A proof of Godel's incompletenes s theorem, which emphasizes the impossibility of achieving a fully complete axiomatic semantics, is included. It is supported by an appendix providing an introduction to the theory of computability based on while-programs. Following a presentation of domain theory, the semantics and methods of proof for several functional languages are treated. The simplest language is that of recursion equations with both call-by-value and call-by-name evaluation. This work is extended to lan guages with higher and recursive types, including a treatment of the eager and lazy lambda-calculi. Throughout, the relationship between denotational and operational semantics is stressed, and the proofs of the correspondenc e between the operation and denotational semantics are provided. The treatment of recursive types - one of the more advanced parts of the book - relies on the use of information systems to represent domains. The book concludes with a chapter on parallel programming

languages, accompanied by a discussion of methods for specifying and verifying nondeterministic and parallel programs. The Little Typer Sweet Valley Addressed to readers at different levels of programming expertise, The Practice ofProlog offers a departure from current books that focus on small programming examples requiringadditional instruction in order to extend them to full programming projects. It shows how to designand organize moderate to large Prolog programs, providing a collection of eight programmingprojects, each with a particular application, and illustrating how a Prolog program was written tosolve the application. These range from a simple learning program to designing a database formolecular biology to natural language generation from plans and stream data analysis.Leon Sterlingis Associate Professor in the Department of Computer Engineering and Science at Case Western ReserveUniversity. He is the coauthor, along with Ehud Shapiro, of The Art of Prolog.Contents: A SimpleLearning Program, Richard O'Keefe. Designing a Prolog Database for

Molecular Biology, Ewing Lusk,Robert Olson, Ross Overbeek, Steve Tuecke. Parallelizing a Pascal Compiler, Eran Gabber. PREDITOR: AProlog-Based VLSI Editor, Peter B. Reintjes. Assisting Register Transfer Level Hardware Design, PaulDrongowski. Design and Implementation of aPartial Evaluation System, Arun Lakhotia, Leon Sterling.Natural Language Generation from Plans, Chris Mellish. Stream Data Analysis in Prolog, Stott Parker. *Live Wires* Simon and Schuster This textbook offers an understanding of the essential concepts of programming languages. The text uses interpreters, written in Scheme, to express the semantics of many essential language elements in a way that is both clear and directly executable.

Related with The Reasoned Schemer:
• Logical Fallacies Worksheet With Answers : click here