

Computer Systems A Programmer S Perspective Globa

Programmer's Guide to PC Video Systems
 The Developer's Code
 User Interface Design for Programmers
 The Computer Boys Take Over
 The Healthy Programmer
 Designing Data-Intensive Applications
 The Programmer's Apprentice
 Computer Systems
 Internet Routing Architectures
 Computer Systems: A Programmer's Perspective Plus Masteringengineering with Pearson Etext -- Access Card Package
 A Programmer's View of Computer Architecture
 Code
 Programmers and Managers
 Computer Engineering for Babies
 Computer Systems: A Programmer's Perspective, Global Edition
 UNIX Systems for Modern Architectures
 Feedback Control for Computer Systems
 Programming Machine Learning
 The Elements of Computing Systems
 Coders at Work
 Embedded Systems Architecture
 Inside the Machine
 Essential Computer Science
 Write Great Code, Volume 1
 Algorithm Design
 New Programmers Start Here
 Computer Systems: Pearson New International Edition
 The Programmer's Guide to SCSI
 The Pattern On The Stone
 Programmers at Work
 Think Like a Programmer
 Who Says Women Can't Be Computer Programmers?
 Deep Learning for Coders with fastai and PyTorch
 Computer Systems
 A Small Matter of Programming
 Programming Language Pragmatics
 Classic Computer Science Problems in Python
 Computer Systems Design And Architecture, 2/E
 Computer Systems
 Kill It with Fire

Computer Systems A Programmer S Perspective Globa

Downloaded from intra.itu.edu by guest

OBRIEN ZAYDEN

[Programmer's Guide to PC Video Systems](#) No Starch Press

How can you take advantage of feedback control for enterprise programming? With this book, author Philipp K. Janert demonstrates how the same principles that govern cruise control in your car also apply to data center management and other enterprise systems. Through case studies and hands-on simulations, you'll learn methods to solve several control issues, including mechanisms to spin up more servers automatically when web traffic spikes. Feedback is ideal for controlling large, complex systems, but its use in software engineering raises unique issues. This book provides basic theory and lots of practical advice for programmers with no previous background in feedback control. Learn feedback concepts and controller design Get practical techniques for implementing and tuning controllers Use feedback "design patterns" for common control scenarios Maintain a cache's "hit rate" by automatically adjusting its size Respond to web

traffic by scaling server instances automatically Explore ways to use feedback principles with queueing systems Learn how to control memory consumption in a game engine Take a deep dive into feedback control theory

The Developer's Code Pragmatic Bookshelf

Printed in full color. To keep doing what you love, you need to maintain your own systems, not just the ones you write code for. Regular exercise and proper nutrition help you learn, remember, concentrate, and be creative--skills critical to doing your job well. Learn how to change your work habits, master exercises that make working at a computer more comfortable, and develop a plan to keep fit, healthy, and sharp for years to come. Small changes to your habits can improve your health--without getting in the way of your work. The Healthy Programmer gives you a daily plan of action that's incremental and iterative just like the software development processes you're used to. Every tip, trick, and best practice is backed up by the advice of doctors, scientists, therapists, nutritionists, and numerous fitness experts. We'll review the latest scientific research to understand how being healthy is good for your body and mind. You'll start by adding a small

amount of simple activity to your day--no trips to the gym needed. You'll learn how to mitigate back pain, carpal tunnel syndrome, headaches, and many other common sources of pain. You'll also learn how to refactor your diet to properly fuel your body without gaining weight or feeling hungry. Then, you'll turn the exercises and activities into a pragmatic workout methodology that doesn't interfere with the demands of your job and may actually improve your cognitive skills. You'll also learn the secrets of prominent figures in the software community who turned their health around by making diet and exercise changes. Throughout, you'll track your progress with a "companion iPhone app". Finally, you'll learn how to make your healthy lifestyle pragmatic, attainable, and fun. If you're going to live well, you should enjoy it. Disclaimer This book is intended only as an informative guide for those wishing to know more about health issues. In no way is this book intended to replace, countermand, or conflict with the advice given to you by your own healthcare provider including Physician, Nurse Practitioner, Physician Assistant, Registered Dietician, and other licensed professionals. Keep in mind that results vary from person to person. This book is not intended as a substitute for medical or nutritional advice from a healthcare

provider or dietician. Some people have a medical history and/or condition and/or nutritional requirements that warrant individualized recommendations and, in some cases, medications and healthcare surveillance. Do not start, stop, or change medication and dietary recommendations without professional medical and/or Registered Dietician advice. A healthcare provider should be consulted if you are on medication or if there are any symptoms that may require diagnosis or medical attention. Do not change your diet if you are ill, or on medication except under the supervision of a healthcare provider. Neither this, nor any other book or discussion forum is intended to take the place of personalized medical care of treatment provided by your healthcare provider. This book was current as of January, 2013 and as new information becomes available through research, experience, or changes to product contents, some of the data in this book may become invalid. You should seek the most up to date information on your medical care and treatment from your health care professional. The ultimate decision concerning care should be made between you and your healthcare provider. Information in this book is general and is offered with no guarantees on the part of the author, editor or The Pragmatic Programmers, LLC. The author, editors and publisher disclaim all liability in connection with the use of this book.

[User Interface Design for Programmers](#) Henry Holt and Company (BYR)

Explores the functions, attributes, and applications of BGP-4 (Border Gateway Protocol Version 4), the de facto interdomain routing protocol, through practical scenarios and configuration examples.

The Computer Boys Take Over Newnes

The contentious history of the computer programmers who developed the software that made the computer revolution possible. This is a book about the computer revolution of the mid-twentieth century and the people who made it possible. Unlike most histories of computing, it is not a book about machines, inventors, or entrepreneurs. Instead, it tells the story of the vast but largely anonymous legions of computer specialists—programmers, systems analysts, and other software developers—who transformed the electronic computer from a scientific curiosity into the defining technology of the modern era. As the systems that they built became increasingly powerful and ubiquitous, these specialists became the focus of a series of critiques of the social and organizational impact of electronic computing. To many of their contemporaries, it seemed the “computer boys” were taking over, not just in the corporate setting, but also in government, politics, and society in general. In *The Computer Boys Take Over*, Nathan Ensmenger traces the rise to power of the computer expert in modern American society. His rich and nuanced portrayal of the men and women (a surprising number of the “computer boys” were, in fact, female) who built their careers around the novel technology of electronic computing explores issues of power, identity, and expertise that have only become more significant in our increasingly computerized society. In his recasting of the drama of the computer revolution through the eyes of its principle revolutionaries, Ensmenger reminds us that the computerization of modern society was not an inevitable process driven by impersonal technological or economic imperatives, but was rather a creative, contentious, and above all, fundamentally human development.

The Healthy Programmer Addison-Wesley Professional

The classic guide to how computers work, updated with new chapters and interactive graphics "For me, Code was a revelation. It was the first book about programming that spoke to me. It started with a story, and it built up, layer by layer, analogy by analogy, until I understood not just the Code, but the System. Code is a book that is as much about Systems Thinking and abstractions as it is about code and programming. Code teaches us how many unseen layers there are between the computer systems that we as users look at every day and the magical silicon rocks that we infused with lightning and taught to think." - Scott Hanselman, Partner Program Director, Microsoft, and host of Hanselminutes Computers are everywhere, most obviously in our laptops and smartphones, but also our cars, televisions, microwave ovens, alarm clocks, robot vacuum cleaners, and other smart appliances. Have you ever wondered what goes on inside these devices to make our lives easier but occasionally more infuriating? For more than 20 years, readers have delighted in Charles Petzold's illuminating story of the secret inner life of computers, and now he has revised it for this new age of computing. Cleverly illustrated and easy to understand, this is the book that cracks the mystery. You'll discover what flashlights, black cats, seesaws, and the ride of Paul Revere can teach you about computing, and how human ingenuity and our compulsion to communicate have shaped every electronic device we use. This new expanded edition explores more deeply the bit-by-bit and gate-by-gate construction of the heart of every smart device, the central processing unit that combines the simplest of basic operations to perform the most complex of feats. Petzold's companion website, [CodeHiddenLanguage.com](#), uses animated

graphics of key circuits in the book to make computers even easier to comprehend. In addition to substantially revised and updated content, new chapters include: Chapter 18: Let's Build a Clock! Chapter 21: The Arithmetic Logic Unit Chapter 22: Registers and Busses Chapter 23: CPU Control Signals Chapter 24: Jumps, Loops, and Calls Chapter 28: The World Brain From the simple ticking of clocks to the worldwide hum of the internet, Code reveals the essence of the digital revolution.

Designing Data-Intensive Applications No Starch Press

Peter Seibel interviews 15 of the most interesting computer programmers alive today in *Coders at Work*, offering a companion volume to Apress's highly acclaimed best-seller *Founders at Work* by Jessica Livingston. As the words “at work” suggest, Peter Seibel focuses on how his interviewees tackle the day-to-day work of programming, while revealing much more, like how they became great programmers, how they recognize programming talent in others, and what kinds of problems they find most interesting. Hundreds of people have suggested names of programmers to interview on the *Coders at Work* web site: [www.codersatwork.com](#). The complete list was 284 names. Having digested everyone's feedback, we selected 15 folks who've been kind enough to agree to be interviewed: Frances Allen: Pioneer in optimizing compilers, first woman to win the Turing Award (2006) and first female IBM fellow Joe Armstrong: Inventor of Erlang Joshua Bloch: Author of the Java collections framework, now at Google Bernie Cosell: One of the main software guys behind the original ARPANET IMPs and a master debugger Douglas Crockford: JSON founder, JavaScript architect at Yahoo! L. Peter Deutsch: Author of Ghostscript, implementer of Smalltalk-80 at Xerox PARC and Lisp 1.5 on PDP-1 Brendan Eich: Inventor of JavaScript, CTO of the Mozilla Corporation Brad Fitzpatrick: Writer of LiveJournal, OpenID, memcached, and Perbal Dan Ingalls: Smalltalk implementor and designer Simon Peyton Jones: Coinventor of Haskell and lead designer of Glasgow Haskell Compiler Donald Knuth: Author of *The Art of Computer Programming* and creator of TeX Peter Norvig: Director of Research at Google and author of the standard text on AI Guy Steele: Coinventor of Scheme and part of the Common Lisp Gang of Five, currently working on Fortress Ken Thompson: Inventor of UNIX Jamie Zawinski: Author of XEmacs and early Netscape/Mozilla hacker

[The Programmer's Apprentice](#) Basic Books

Deep learning is often viewed as the exclusive domain of math PhDs and big tech companies. But as this hands-on guide demonstrates, programmers comfortable with Python can achieve impressive results in deep learning with little math background, small amounts of data, and minimal code. How? With *fastai*, the first library to provide a consistent interface to the most frequently used deep learning applications. Authors Jeremy Howard and Sylvain Gugger, the creators of *fastai*, show you how to train a model on a wide range of tasks using *fastai* and PyTorch. You'll also dive progressively further into deep learning theory to gain a complete understanding of the algorithms behind the scenes. Train models in computer vision, natural language processing, tabular data, and collaborative filtering Learn the latest deep learning techniques that matter most in practice Improve accuracy, speed, and reliability by understanding how deep learning models work Discover how to turn your models into web applications Implement deep learning algorithms from scratch Consider the ethical implications of your work Gain insight from the foreword by PyTorch cofounder, Soumith Chintala

Computer Systems No Starch Press

For courses in Computer Science and Programming *Computer Systems: A Programmer's Perspective* explains the underlying elements common among all computer systems and how they affect general application performance. Written from the programmer's perspective, this book strives to teach students how understanding basic elements of computer systems and executing real practice can lead them to create better programs. Spanning across computer science themes such as hardware architecture, the operating system, and systems software, the 3rd Edition serves as a comprehensive introduction to programming. This book strives to create programmers who understand all elements of computer systems and will be able to engage in any application of the field—from fixing faulty software, to writing more capable programs, to avoiding common flaws. It lays the groundwork for students to delve into more intensive topics such as computer architecture, embedded systems, and cybersecurity. This book focuses on systems that execute an x86-64 machine code, and recommends that students have access to a Linux system for this course. Students should have basic familiarity with C or C++. The full text downloaded to your computer With eBooks you can: search for key concepts, words and phrases make highlights and notes as you study share your notes with friends eBooks are downloaded to your computer and accessible either offline through the Bookshelf (available as a free download), available online and

also via the iPad and Android apps. Upon purchase, you'll gain instant access to this eBook. Time limit The eBooks products do not have an expiry date. You will continue to access your digital ebook products whilst you have your Bookshelf installed.

[Internet Routing Architectures](#) Simon and Schuster

"Computer systems: A Programmer's Perspective explains the underlying elements common among all computer systems and how they affect general application performance. Written from the programmer's perspective, this book strives to teach students how understanding basic elements of computer systems and executing real practice can lead them to create better programs."--Publisher's website.

[Computer Systems: A Programmer's Perspective Plus Masteringengineering with Pearson Etext -- Access Card Package](#) John Wiley & Sons

Norbert Wiener, perhaps better than anyone else, understood the intimate and delicate relationship between control and communication: that messages intended as commands do not necessarily differ from those intended simply as facts. Wiener noted the paradox when the modem computer was hardly more than a laboratory curiosity. Thirty years later, the same paradox is at the heart of a severe identity crisis which confronts computer programmers. Are they primarily members of "management" acting as foremen, whose task it is to ensure that orders emanating from executive suites are faithfully translated into comprehensible messages? Or are they perhaps simply engineers preoccupied with the technical difficulties of relating "software" to "hardware" and vice versa? Are they aware, furthermore, of the degree to which their work whether as manager or engineer-routinizes the work of others and thereby helps shape the structure of social class relationships? I doubt that many of us who lived through the first heady and frantic years of software development—at places like the RAND and System Development Corporations—ever took time to think about such questions. The science fiction-like setting of mysterious machines, blinking lights, and torrents of numbers served to awe outsiders who could only marvel at the complexity of it all. We were insiders who constituted a secret society into which only initiates were welcome. So today I marvel at the boundless audacity of a rank outsider in writing a book like *Programmers and Managers*.

A Programmer's View of Computer Architecture Apress

Computer Systems, Fifth Edition provides a clear, detailed, step-by-step introduction to the central concepts in computer organization, assembly language, and computer architecture. It urges students to explore the many dimensions of computer systems through a top-down approach to levels of abstraction. By examining how the different levels of abstraction relate to one another, the text helps students look at computer systems and their components as a unified concept.

[Code](#) Elsevier

For Computer Systems, Computer Organization and Architecture courses in CS, EE, and ECE departments. Few students studying computer science or computer engineering will ever have the opportunity to build a computer system. On the other hand, most students will be required to use and program computers on a near daily basis. *Computer Systems: A Programmer's Perspective* introduces the important and enduring concepts that underlie computer systems by showing how these ideas affect the correctness, performance, and utility of application programs. The text's hands-on approach (including a comprehensive set of labs) helps students understand the under-the-hood operation of a modern computer system and prepares them for future courses in systems topics such as compilers, computer architecture, operating systems, and networking.

Programmers and Managers Springer Science & Business Media

"Whether you're a novice or a seasoned professional, there's an Aha! moment in this book for everyone." - James Watson, Adaptive "Highly recommended to everyone interested in deepening their understanding of Python and practical computer science." —Daniel Kenney-Jung, MD, University of Minnesota Key Features • Master formal techniques taught in college computer science classes • Connect computer science theory to real-world applications, data, and performance • Prepare for programmer interviews • Recognize the core ideas behind most “new” challenges • Covers Python 3.7 Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About The Book Programming problems that seem new or unique are usually rooted in well-known engineering principles. *Classic Computer Science Problems in Python* guides you through time-tested scenarios, exercises, and algorithms that will prepare you for the “new” problems you’ll face when you start your next project. In this amazing book, you'll tackle dozens of coding challenges, ranging from simple tasks like binary search algorithms to clustering data using k-means. As you work through examples for web development,

machine learning, and more, you'll remember important things you've forgotten and discover classic solutions that will save you hours of time. What You Will Learn • Search algorithms • Common techniques for graphs • Neural networks • Genetic algorithms • Adversarial search • Uses type hints throughout This Book Is Written For For intermediate Python programmers. About The Author David Kopec is an assistant professor of Computer Science and Innovation at Champlain College in Burlington, Vermont. He is the author of *Dart for Absolute Beginners* (Apress, 2014), *Classic Computer Science Problems in Swift* (Manning, 2018), and *Classic Computer Science Problems in Java* (Manning, 2020) Table of Contents 1. Small problems 2. Search problems 3. Constraint-satisfaction problems 4. Graph problems 5. Genetic algorithms 6. K-means clustering 7. Fairly simple neural networks 8. Adversarial search 9. Miscellaneous problems

Computer Engineering for Babies Association for Computing Machinery (ACM)
NOTE: Before purchasing, check with your instructor to ensure you select the correct ISBN. Several versions of Pearson's MyLab & Mastering products exist for each title, and registrations are not transferable. To register for and use Pearson's MyLab & Mastering products, you may also need a Course ID, which your instructor will provide. Used books, rentals, and purchases made outside of Pearson If purchasing or renting from companies other than Pearson, the access codes for Pearson's MyLab & Mastering products may not be included, may be incorrect, or may be previously redeemed. Check with the seller before completing your purchase. For courses in Computer Organization and Architecture This package includes MasteringEngineering® Computer systems: A Programmer's Perspective explains the underlying elements common among all computer systems and how they affect general application performance. Written from the programmer's perspective, this book strives to teach readers how understanding basic elements of computer systems and executing real practice can lead them to create better programs. Spanning across computer science themes such as hardware architecture, the operating system, and systems software, the Third Edition serves as a comprehensive introduction to programming. This book strives to create programmers who understand all elements of computer systems and will be able to engage in any application of the field--from fixing faulty software, to writing more capable programs, to avoiding common flaws. It lays the groundwork for readers to delve into more intensive topics such as computer architecture, embedded systems, and cyber security. This book focuses on systems that execute an x86-64 machine code, and recommends that programmers have access to a Linux system for this course. Programmers should have basic familiarity with C or C++. Personalize Learning with MasteringEngineering MasteringEngineering is an online homework, tutorial, and assessment system, designed to improve results through personalized learning. This innovative online program emulates the instructor's office hour environment, engaging and guiding students through engineering concepts with self-paced individualized coaching With a wide range of activities available, students can actively learn, understand, and retain even the most difficult concepts. 0134123832/9780134123837 Computer Systems: A Programmer's Perspective plus MasteringEngineering with Pearson eText -- Access Card Package, 3/e Package consists of: * 013409266X/9780134092669 Computer Systems: A Programmer's Perspective, 3/e * 0134071921/9780134071923 MasteringEngineering with Pearson eText -- Standalone Access Card -- for Computer Systems: A Programmer's Perspective, 3/e *Computer Systems: A Programmer's Perspective, Global Edition* Jones & Bartlett Learning

The real challenge of programming isn't learning a language's syntax—it's learning to creatively solve problems so you can build something great. In this one-of-a-kind text, author V. Anton Spraul breaks down the ways that programmers solve problems and teaches you what other introductory books often ignore: how to Think Like a Programmer. Each chapter tackles a single programming concept, like classes, pointers, and recursion, and open-ended exercises throughout challenge you to apply your knowledge. You'll also learn how to: -Split problems into discrete components to make them easier to solve -Make the most of code reuse with functions, classes, and libraries -Pick the perfect data structure for a particular job -Master more advanced programming tools like recursion and dynamic memory -Organize your thoughts and develop strategies to tackle particular types of problems Although the book's examples are written in C++, the creative problem-solving concepts they illustrate go beyond any particular language; in fact, they often reach outside the realm of computer science. As the most skillful programmers know, writing great code is a creative art—and the first step in creating your masterpiece is learning to Think Like a Programmer.

UNIX Systems for Modern Architectures Apress

This introductory text offers a contemporary treatment of computer architecture using assembly and machine language with a focus on software. Students learn how computers work through a clear, generic presentation of a computer architecture, a departure from the traditional focus on a specific architecture. A computer's capabilities are introduced within the context of software, reinforcing the software focus of the text. Designed for computer science majors in an assembly language course, this text uses a top-down approach to the material that enables students to begin programming immediately and to understand the assembly language, the interface between hardware and software. The text includes examples from the MIPS RISC (reduced instruction set computer) architecture, and an accompanying software simulator package simulates a MIPS RISC processor (the software does not require a MIPS processor to run).

Feedback Control for Computer Systems Cisco Press

Data is at the center of many challenges in system design today. Difficult issues need to be figured out, such as scalability, consistency, reliability, efficiency, and maintainability. In addition, we have an overwhelming variety of tools, including relational databases, NoSQL datastores, stream or batch processors, and message brokers. What are the right choices for your application? How do you make sense of all these buzzwords? In this practical and comprehensive guide, author Martin Kleppmann helps you navigate this diverse landscape by examining the pros and cons of various technologies for processing and storing data. Software keeps changing, but the fundamental principles remain the same. With this book, software engineers and architects will learn how to apply those ideas in practice, and how to make full use of data in modern applications. Peer under the hood of the systems you already use, and learn how to use and operate them more effectively Make informed decisions by identifying the strengths and weaknesses of different tools Navigate the trade-offs around consistency, scalability, fault tolerance, and complexity Understand the distributed systems research upon which modern databases are built Peek behind the scenes of major online services, and learn from their architectures

Programming Machine Learning No Starch Press

You've decided to tackle machine learning - because you're job hunting, embarking on a new project, or just think self-driving cars are cool. But where to start? It's easy to be intimidated, even

as a software developer. The good news is that it doesn't have to be that hard. Master machine learning by writing code one line at a time, from simple learning programs all the way to a true deep learning system. Tackle the hard topics by breaking them down so they're easier to understand, and build your confidence by getting your hands dirty. Peel away the obscurities of machine learning, starting from scratch and going all the way to deep learning. Machine learning can be intimidating, with its reliance on math and algorithms that most programmers don't encounter in their regular work. Take a hands-on approach, writing the Python code yourself, without any libraries to obscure what's really going on. Iterate on your design, and add layers of complexity as you go. Build an image recognition application from scratch with supervised learning. Predict the future with linear regression. Dive into gradient descent, a fundamental algorithm that drives most of machine learning. Create perceptrons to classify data. Build neural networks to tackle more complex and sophisticated data sets. Train and refine those networks with backpropagation and batching. Layer the neural networks, eliminate overfitting, and add convolution to transform your neural network into a true deep learning system. Start from the beginning and code your way to machine learning mastery. What You Need: The examples in this book are written in Python, but don't worry if you don't know this language: you'll pick up all the Python you need very quickly. Apart from that, you'll only need your computer, and your code-adept brain.

The Elements of Computing Systems "O'Reilly Media, Inc."

A picture book biography of Ada Lovelace, the woman recognized today as history's first computer programmer—she imagined them 100 years before they existed! In the early nineteenth century lived Ada Byron: a young girl with a wild and wonderful imagination. The daughter of internationally acclaimed poet Lord Byron, Ada was tutored in science and mathematics from a very early age. But Ada's imagination was never meant to be tamed and, armed with the fundamentals of math and engineering, she came into her own as a woman of ideas—equal parts mathematician and philosopher. From her whimsical beginnings as a gifted child to her most sophisticated notes on Charles Babbage's Analytical Engine, this book celebrates the woman recognized today as the first computer programmer. This title has Common Core connections. Christy Ottaviano Books

Coders at Work Pearson Higher Ed

Most programmers' fear of user interface (UI) programming comes from their fear of doing UI design. They think that UI design is like graphic design—the mysterious process by which creative, latte-drinking, all-black-wearing people produce cool-looking, artistic pieces. Most programmers see themselves as analytic, logical thinkers instead—strong at reasoning, weak on artistic judgment, and incapable of doing UI design. In this brilliantly readable book, author Joel Spolsky proposes simple, logical rules that can be applied without any artistic talent to improve any user interface, from traditional GUI applications to websites to consumer electronics. Spolsky's primary axiom, the importance of bringing the program model in line with the user model, is both rational and simple. In a fun and entertaining way, Spolsky makes user interface design easy for programmers to grasp. After reading *User Interface Design for Programmers*, you'll know how to design interfaces with the user in mind. You'll learn the important principles that underlie all good UI design, and you'll learn how to perform usability testing that works.

Best Sellers - Books :

- [Why A Daughter Needs A Dad: Celebrate Your Father Daughter Bond This Father's Day With This Special Picture Book! \(always In My Heart\) By Gregory E. Lang](#)
- [How To Catch A Leprechaun By Adam Wallace](#)
- [Flash Cards: Sight Words](#)
- [Adult Children Of Emotionally Immature Parents: How To Heal From Distant, Rejecting, Or Self-involved Parents By Lindsay C. Gibson Psyd](#)
- [Hunting Adeline \(cat And Mouse Duet\) By H. D. Carlton](#)
- [Things We Never Got Over \(knockemout\)](#)
- [Daisy Jones & The Six: A Novel By Taylor Jenkins Reid](#)
- [Bluey And Bingo's Fancy Restaurant Cookbook: Yummy Recipes, For Real Life](#)
- [Rich Dad Poor Dad: What The Rich Teach Their Kids About Money That The Poor And Middle Class Do Not! By Robert T. Kiyosaki](#)
- [The Silent Patient](#)