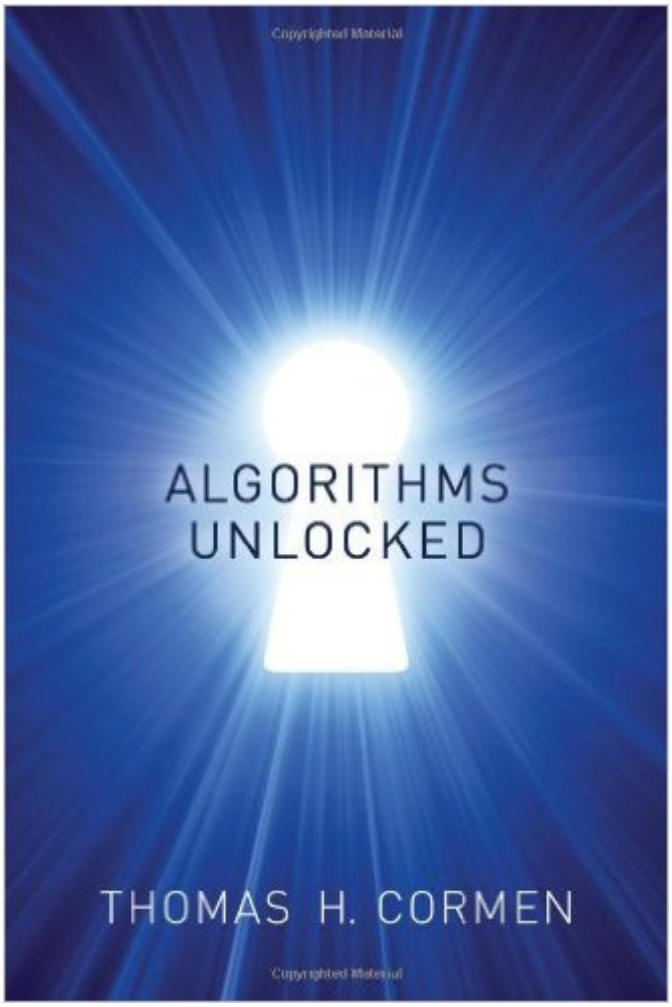
Algorithms Unlocked (MIT Press)



Synopsis

Have you ever wondered how your GPS can find the fastest way to your destination, selecting one route from seemingly countless possibilities in mere seconds? How your credit card account number is protected when you make a purchase over the Internet? The answer is algorithms. And how do these mathematical formulations translate themselves into your GPS, your laptop, or your smart phone? This book offers an engagingly written guide to the basics of computer algorithms. In Algorithms Unlocked, Thomas Cormen -- coauthor of the leading college textbook on the subject -- provides a general explanation, with limited mathematics, of how algorithms enable computers to solve problems. Readers will learn what computer algorithms are, how to describe them, and how to evaluate them. They will discover simple ways to search for information in a computer; methods for rearranging information in a computer into a prescribed order ("sorting"); how to solve basic problems that can be modeled in a computer with a mathematical structure called a "graph" (useful for modeling road networks, dependencies among tasks, and financial relationships); how to solve problems that ask questions about strings of characters such as DNA structures; the basic principles behind cryptography; fundamentals of data compression; and even that there are some problems that no one has figured out how to solve on a computer in a reasonable amount of time.

Book Information

Series: MIT Press

Paperback: 240 pages

Publisher: The MIT Press (March 1, 2013)

Language: English

ISBN-10: 0262518805

ISBN-13: 978-0262518802

Product Dimensions: 6 x 0.5 x 9 inches

Shipping Weight: 12 ounces (View shipping rates and policies)

Average Customer Review: 4.3 out of 5 starsÂ Â See all reviewsÂ (23 customer reviews)

Best Sellers Rank: #48,645 in Books (See Top 100 in Books) #25 inÂ Books > Computers & Technology > Programming > Algorithms #14928 inÂ Books > Reference

Customer Reviews

This is an interesting book. It’s halfway between the typical science-book-for-non-scientists and a textbook. Unlike most science books for non-scientists, the author doesn’t dumb down the material for the reader; he assumes you’re intelligent and are willing to work a bit to understand what he says

(including following mathematical arguments with actual equations). In fact, I only bought the book because the author was Cormen; I was familiar with his algorithms textbook (Introduction to Algorithms, better known as CLRS after the initials of the authors), which is absolutely authoritative, so I knew the material would be correct. (Having Julie Sussman, P.P.A., as the proofreader was also a big plus.) Unlike CLRS, this is not a textbook, but rather kind of a "pre-textbook". Its purpose is to whet your appetite, to teach you enough about the material so that you can decide if you want to study it more formally or not. If so, you can go straight to CLRS. If not, at least you'll have learned something. The people who will get the most out of this book are self-taught programmers who have never taken a course in algorithms but who nevertheless need to know this material. (And believe me, if you're a programmer, you _do_ need to know this material, whether you realize it or not.) Non-programmers may find that the book is too technical (especially the last few chapters, which get fairly heavily into topics like number theory and reductions of NP-complete problems). Despite this, I can think of no other book that provides such a clear introduction to the field of algorithms with so few prerequisites. Cormen's explanations are lucid and interesting throughout, and the topics are motivated by real-world applications which often don't find their way into textbook descriptions.

Upon un-boxing, I looked at the back cover photo and thought, "uh-oh, another Ivy League professor who thinks everything is intuitively obvious, and disrespects you if it's not." One of the reasons more Americans don't go into computer science IMHO is that the basics are NOT intuitive, and turning your class into a boot camp - with programming assignments made purposely too time consuming - does nothing but hinder....oh, wait, this is a book review, I digress.Anyways, this book is NOT that, it's a good book. For example, chapter 2 goes from assuming no knowledge of computer programming to asymptotic notation, and it's actually understandable! I've taken an algorithms class, and although able to muddle through it, I only understood about 1/4 the theoretical part. This book helps fill that void and I have renewed confidence that I will be able to understand the more theoretical aspect to algorithms the next time I encounter them (I'm back in school studying CS at night). I've read this book once, from cover to cover - retaining about 1/2 of it. Now it's a second time with highlighter and notes so that I can retain the book in its entirety.It's not really a book for the "we're all going to be cyborgs, algorithms rule the world" pop-computing crowd, and I suspect it would take a gifted reader to understand this book if he or she truly had no clue about programming. A computer program algorithm is a different way of looking at the world than most folks are used to. But in the first chapter - presumably the one somebody would thumb through at the book store - the author actually points the reader to a less technical book if this one seems too

technical.

Admittedly, I’m not finished with it yet & may revise this review later on... So far I’ve read up through the Sorting chapters.As stated in other reviews & in the book’s intro, the author assumes no prior programming knowledge.It’s confusing to consider Arrays as starting a 1 & having to add/subtract everywhere, but ok I can deal with it...What wasn’t clear at purchase time was the assumption of familiarity with math at let’s say "Algebra II" or pre-calc levels.If you don’t understand or remember how to deal with factorials, polynomials, exponents & logarithms, you’ll need to look those up.These are not particularly difficult or high-level, but my last prolonged exposure to these concepts was more than a decade ago.I do appreciate the need to keep the concepts abstract enough to apply across various applications, but it would be nice to see some concrete implementation. I’d have a much easier time following code examples in a language/syntax I’ve seen more recently.I’m considering the "Algorithms 4th Ed" book by Sedgewick & Wayne for that reason: practical application w/examples in Java.Modern programming best practices described in books like "The Pragmatic Programmer" or "Code Complete" also clearly emphasize the need to use unique/meaningful names for variables and functions so that people can easily follow your work. It also helps YOU to follow your own work. This book instead takes you back to math class where everything is a meaningless jumble of alphabet soup - "x,y,z,q,p,r,a" - and you have to constantly bounce back & forth (sometimes over several pages) to keep track of what’s what.Which example gives you a better idea what’s going on?

Place: Urban Landscapes as Public History (MIT Press) The New Science of Cities (MIT Press) Design Meets Disability (MIT Press) Speculative Everything: Design, Fiction, and Social Dreaming (MIT Press)