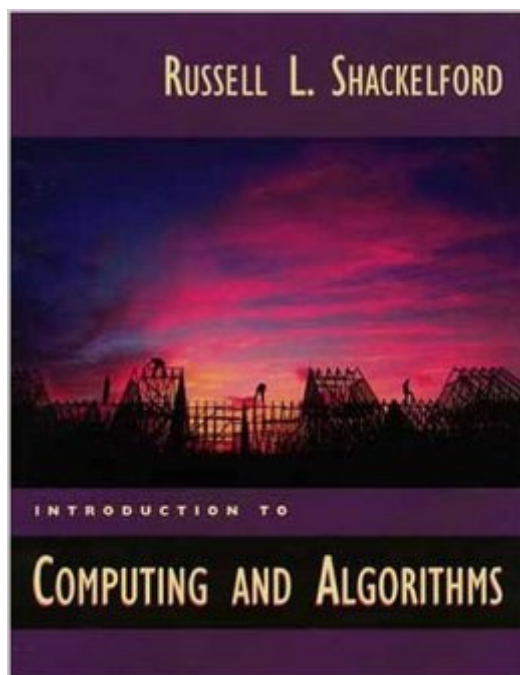# Introduction To Computing And Algorithms

Synopsis

This work prepares students for the world of computing by giving them a solid foundation in the science of computer science, algorithms. By taking an algorithm-based approach to the subject, this introductory text seeks to help students grasp overall concepts, rather than bogging them down in specific syntax detail of a programming language that can become obsolete quickly.

Book Information

Paperback: 399 pages

Publisher: Pearson; 1 edition (October 26, 1997)

Language: English

ISBN-10: 0201314517

ISBN-13: 978-0201314519

Product Dimensions: 7.4 x 1 x 9 inches

Shipping Weight: 1.6 pounds (View shipping rates and policies)

Average Customer Review: 4.5 out of 5 stars See all reviews (13 customer reviews)

Best Sellers Rank: #96,601 in Books (See Top 100 in Books)   #29 in Books > Textbooks > Computer Science > Algorithms   #62 in Books > Computers & Technology > Databases & Big Data > Data Processing   #65 in Books > Computers & Technology > Programming > Algorithms

Customer Reviews

As a teaching assistant for CS1501, Introduction to Computing at Georgia Tech, a successful brain-child of Dr. Shackelford's, I have used this book from both a student's and a teacher's standpoint, and thus feel adept at highlighting its successes.This book provides a concise, clear review of the basic concepts of algorithmic thought and its subsequent expression and application. After a brief review of the history of computing, the reader is launched into an ever-deepening understanding of basic CS tools from a problem-solving point of view.Topics of interest include dynamic data structures (BST's, linked lists), array storage, stacks + queues, object-oriented programming, precedence and dependence, and a brief sojourn into higher-level theory concepts.The highlight, and in my opinion success, of this book is the fact that no "real" programming language is used to notate any examples. Rather, Dr. Shackelford, along with other TA's of recent past, devised a pseudo-language which encapsulates many elements of previous educational languages (such as Pascal); naturally, this language is not vulnerable to obsolescence. Also, since this language can not be practically compiled, the reader is forced to trace through examples on his or her own, building his or her skills in mentally evaluating algorithms.This book,

once limited to the Georgia Tech CS curriculum, has now expanded to many other colleges and universities, and more institutions are becoming interested as the months progress. This text is a must-have for any individual interested in getting a substantial taste in algorithms and computing.

I can only copy the other two reviewers. This book makes for an excellent....no, an awesome introduction into computing, algorithms and data structures (even OOP fundamentals are covered).While this book is a textbook for computer science students it's neither dry to read nor hard to understand. There are several amusing or eye-opening anecdotes plus an excellent history section of how the alphabet developed (and with it book printing, etc.) and how programming influences our way of thinking.On the technical side this book is the first I read that explained the inner workings of recursive functions in such a simple way that I wondered why there's so much mystery around recursion. Not to mention all the other things like ADTs, BSTs, debugging and so on.Although this book uses pseudo-code for its explanations and exercises don't think of this as a disadvantage. In fact, it makes things easier to follow and makes for good exercises to convert the pseudo-code to real code to whatever language you're using.Anyone who's programming should have read this book. Anyone who's going to learn programming MUST read this book (well, to be honest, you do not need to, but don't come crying if, after years, you read it and find out how much easier it would have been....).So I give this book 5 Million stars, a hug and a big smile :)))

I've been a professional computer programmer for the last seven years, with no formal training. I bought this book to begin improving my grasp of computer theory as well as practice. I've found this book to be eminantly readable, an actual page turner. Parts of it are quite elementary, but it's very good at explaining "why" as well as "what".I consider the pseudo-code an advantage, since it forces me to look at unfamiliar terms and structures and equate them with the equivalent items in my "real" language. It teaches *theory*, which I can apply to any language, not implementation in a single language that will be obsolete in a year or two.

If the title of this book was "Introduction to Programming", then I would wholeheartedly agree with Mr. Black below. Indeed, Dr. Shackelford does not teach programming in his book. Rather, he presents algorithms in a general sense, and algorithmic thought over programming practice is presented to the novice.An adequate programming foundation begins with the trials and errors of learning one's first programming knowledge. Dr. Shackelford's book better preparesone for their first programming language experience, but is by no means a substitute for a programming language

course if one's goal is to learn to program.There are many that wish to gain a better knowledge of computing from a higher-level, theoretical standpoint. Also as an upper-level computer science theory TA at Georgia Tech, I realize the importance of being introduced to a concept in its high-level foundations, rather than in its gory details.The CS curriculum at Tech is moving to Scheme as an introduction to computing and computer science. While I adamantly feel this is an excellent move on the part of the College of Computing, given their goal to output programmers from the school, I also feel some generalized theoretical foundations are lacking from the course that were present when it was taught with Dr. Shackelford's book.Overall, if one is interested in grasping high-level algorithms and receiving a basic understanding of how computers process data and execute programs, then this book is wonderful. If, however, one is interested in coding VB applications in a couple of weeks, this book is not appropriate.

Introduction to Computing and Algorithms Student Solutions Manual for Differential Equations: Computing and Modeling and Differential Equations and Boundary Value Problems: Computing and Modeling GPU Computing Gems Emerald Edition (Applications of GPU Computing Series) Applied Cryptography: Protocols, Algorithms, and Source Code in C [APPLIED CRYPTOGRAPHY: PROTOCOLS, ALGORITHMS, AND SOURCE CODE IN C BY Schneier, Bruce (Author) Nov-01-1995 Combinatorial Optimization: Theory and Algorithms (Algorithms and Combinatorics) Geometric Algorithms and Combinatorial Optimization (Algorithms and Combinatorics) Algorithms in C, Parts 1-5 (Bundle): Fundamentals, Data Structures, Sorting, Searching, and Graph Algorithms (3rd Edition) Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms Practical Algorithms in Pediatric Hematology and Oncology: (Practical Algorithms in Pediatrics. Series Editor: Z. Hochberg) Introduction to Computing Systems: From Bits and Gates to C and Beyond Introduction to Computing and Programming with Java: A Multimedia Approach Quantum Computing: A Gentle Introduction (Scientific and Engineering Computation) An Introduction to Statistics with Python: With Applications in the Life Sciences (Statistics and Computing) Introduction to Computing and Programming in Python (4th Edition) Quantum Nanoelectronics: An introduction to electronic nanotechnology and quantum computing Cloud Computing - An Introduction Introduction to Parallel Computing (2nd Edition) Introduction to Computing Using Python: An Application Development Focus, 2nd Edition A Web-Based Introduction to Programming: Essential Algorithms, Syntax, and Control Structures Using PHP, HTML, and MySQL, Third Edition Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra (Undergraduate Texts in

Mathematics)