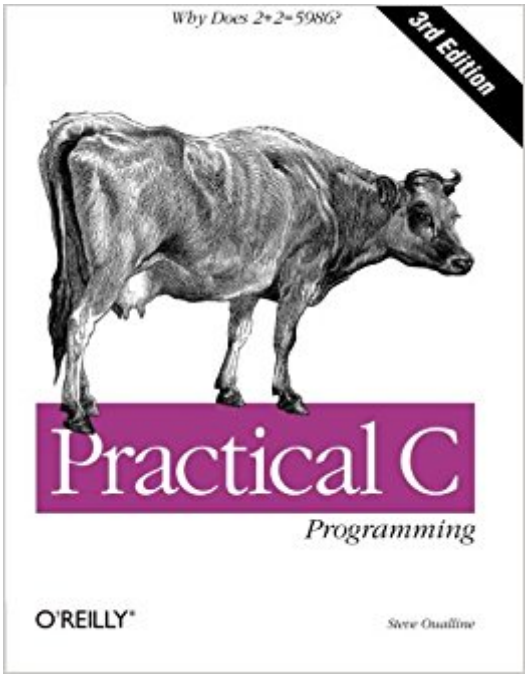
Practical C Programming: Why Does 2+2 = 5986? (Nutshell Handbooks)



Synopsis

There are lots of introductory C books, but this is the first one that has the no-nonsense, practical approach that has made Nutshell HandbooksÃ Â® famous.C programming is more than just getting the syntax right. Style and debugging also play a tremendous part in creating programs that run well and are easy to maintain. This book teaches you not only the mechanics of programming, but also describes how to create programs that are easy to read, debug, and update.Practical rules are stressed. For example, there are fifteen precedence rules in C (&& comes before || comes before ?:). The practical programmer reduces these to two:Multiplication and division come before addition and subtraction.Contrary to popular belief, most programmers do not spend most of their time creating code. Most of their time is spent modifying someone else's code. This books shows you how to avoid the all-too-common obfuscated uses of C (and also to recognize these uses when you encounter them in existing programs) and thereby to leave code that the programmer responsible for maintenance does not have to struggle with. Electronic Archaeology, the art of going through someone else's code, is described.This third edition introduces popular Integrated Development Environments on Windows systems, as well as UNIX programming utilities, and features a large statistics-generating program to pull together the concepts and features in the language.

Book Information

Series: Nutshell Handbooks

Paperback: 456 pages

Publisher: O'Reilly Media; 3rd edition (August 11, 1997)

Language: English

ISBN-10: 1565923065

ISBN-13: 978-1565923065

Product Dimensions: 7 x 1.1 x 9.2 inches

Shipping Weight: 2 pounds (View shipping rates and policies)

Average Customer Review: 4.3 out of 5 stars 660 customer reviews

Best Sellers Rank: #54,032 in Books (See Top 100 in Books) #16 inÃ Â Books > Computers & Technology > Operating Systems > Unix #20 inÃ Â Books > Computers & Technology > Programming > Microsoft Programming > C & C++ Windows Programming #26 inÃ Â Books > Computers & Technology > Programming > Languages & Tools > C & C++ > C

Customer Reviews

There are lots of introductory C books, but this is the one that has the no-nonsense, practical

approach that has made Nutshell Handbooks(R) famous. C programming is more than just getting the syntax right. Style and debugging also play a tremendous part in creating programs that run well and are easy to maintain. This new edition of Practical C Programming teaches you not only the mechanics of programming, but also how to create programs that are easy to read, debug, and maintain. It features more extensive examples, offers an introduction to graphical development environments, and describes Electronic Archaeology (the art of going through someone else's code). As in earlier editions, practical rules are still stressed. For example, there are fifteen precedence rules in C (&& comes before || comes before ?:). The practical programmer reduces these to two: multiplication and division come before addition and subtraction put parentheses around everything else. Topics covered: Good programming style C syntax: what to use and what not to use The programming environment, including integrated development kits The total programming process Floating point limitations Tricks and surprises Program examples conform to ANSI C. Covers several Windows compilers, as well as UNIX compilers.

Steve Oualline lives in Southern California, where he works as a software engineer for a major phone company. In his free time he is a real engineer on the Poway Midland Railroad. Steve has written almost a dozen books on programming and Linux software. His web site is http://www.oualline.com .

Only ~100 pages in, but I gotta say there really isn't a more extensive book on C++ out there. One would expect no less, considering it was written by its creator. But the organization of the material makes an astounding amount of sense. Most undergrad books will have you working with the standard library and creating programs that bear very little relevance to actual programming, or try to focus on one specific style (usually object-oriented). This book tries to approach many different styles in a pragmatic sense. Just like how there isn't one language built for every task, there isn't one style that is infinitely superior to another. Stroustrup attempts to showcase this by offering scenarios where operations are exceedingly resource-intensive, (such as when handling large data structures) and offers new ways of solving a problem, compared to conventional methods. Or rather, he insists conventional methods should involve deep thought into resource management and writing clean code. Just having gone through Part One of the book, I feel as though I've improved immensely.Side note: This book attempts to get users in the habit of using C++11 assignment operators and such. This isn't necessarily a bad thing, but it should be worth noting that some compilers will not work with these examples without specifying that it is written in C++11 format. (i.e.

affixing -std=c++11 during compilation. Some professors who teach out of other C++ books may mark you wrong for using ++11 or even ++14 conventions.)I would recommend this book to anyone wishing to really delve deep into C++ and produce efficient, clean code. Anyone looking to get "closer to the machine", so to speak, would most likely find this book to tickle their fancy.

(This is for the kindle edition) would give this book 4 or 5 stars if not for the horribly misaligned typography. The text has keywords, variables, etc. in a different font (which is a very good idea), but the text in this font is terribly misaligned. Where the author wants to have "x++", the first + may or may not be next to the x, and the second + winds up two words to the right on top of another character, sometimes rendering the text unreadable. This is not an occasional occurrence; it happens more often than not. I don’t know if the paper edition has the same problem, since I haven’t been able to find it (which was why I bought the kindle version)

This is a nice reference book on C++.If I were new to C++, I would not start learning C++ with this book. Rather, I would use these below three steps and the relevant books in this order:1. Accelerated C++ by Andrew Koenig & Barbara Moo -- Read and practice example code and exercises from this book first.2. Programming: Principles and Practice Using C++ (2nd Edition) by Bjarne Stroustrup -- Chapters 5,6, and 7 are gems in this book. You can in fact use this book in parallel with the Accelerated C++ book and3. (a) The C++ Programming Language (4th Edition) by Bjarne Stroustrup, -- Definite reference book to have.3. (b) The C++ Standard Library: A Tutorial and Reference (2nd Edition) by Nicolai M. Josuttis, -- Clear examples and very methodical3. (c) C++ Templates: The Complete Guide by David Vandevoorde -- What can I say! This is simply a classic.C++ is not a race. It is a marathon. So, enjoy learning and also make use of many many C++ resources online.

This book is simply a must-read for C++ programmers.I really liked it’s structure: there’s a brief "Tour of C++" before the more detailed chapters. In this tour, you can see in a glance what C++11 offers for many programming tasks that’s not present in earlier standards: variadic templates, static assertions, many concurrency primitives, a new uniform initialization syntax, initializer lists, range-for loop, new STL containers, etc.After that, there are detailed chapters intended to cover all the details of all the language features and the STL. After seeing a lot of cool stuff in the tour, you are motivated enough to go through the detailed descriptions of everything written by the C++ creator himself.But pay attention to the title: the book is about "The C++ Programming Language". It’s not

intended to instruct you about:- How to program;- How to write efficient, readable and/or modularized code using C++;- How to use concurrency to enhance the performance of algorithms;- How to design APIs (although the STL is a good example in many situations);- What are the best tools (compiler, VCSs, build systems, IDEs, libraries) to develop C++ programs.It's rather a hitchhiker's guide to C++.

Practical C Programming: Why Does 2+2 = 5986? (Nutshell Handbooks) Python Programming: Python Programming for Beginners, Python Programming for Intermediates, Python Programming for Advanced C++: The Ultimate Crash Course to Learning the Basics of C++ (C programming, C++ in easy steps, C++ programming, Start coding today) (CSS,C Programming, ... Programming,PHP, Coding, Java Book 1) Spiritual Activation: Why Each of Us Does Make the Difference (Why Each of Us Does Makes the Difference) Family Law in a Nutshell, 5th (In a Nutshell (West Publishing)) (Nutshell Series) C++ and Python Programming: 2 Manuscript Bundle: Introductory Beginners Guide to Learn C++ Programming and Python Programming C++ and Python Programming 2 Bundle Manuscript. Introductory Beginners Guide to Learn C++ Programming and Python Programming Python Programming: The Complete Step By Step Guide to Master Python Programming and Start Coding Today! (Computer Programming Book 4) Why Does E=mc2? (And Why Should We Care?) Admiralty in a Nutshell, 6th (In a Nutshell (West Publishing)) (Nutshells) Land Use in a Nutshell (Nutshell Series) Government Contracts in a Nutshell, 5th (West Nutshell Series) Government Contracts in a Nutshell (Nutshell Series) Regulated Industries in a Nutshell (Nutshell Series) Animal Law in a Nutshell (In a Nutshell (West Publishing)) (Nutshells) Burr's Entertainment Law in a Nutshell, 2d (In a Nutshell (West Publishing)) Employment Law in a Nutshell, Third Edition (West Nutshell) Consumer Law in a Nutshell (NUTSHELL SERIES) Securities Regulation in a Nutshell, 10th (Nutshell Series) Toxic Torts in a Nutshell (In a Nutshell (West Publishing))