<<编程的本质>>

图书基本信息

书名：<<编程的本质>>

13位ISBN编号：9787111300274

10位ISBN编号：7111300270

出版时间：2010

出版时间：机械工业出版社

作者：Alexander Stepanov,Paul McJones

页数：262

前言

This book applies the deductive method to programming by affiliating programs with the abstract mathematical theories that enable them to work. Specification of these theories, algorithms written in terms of these theories, and theorems and lemmas describing their properties are presented together. The implementation of the algorithms in a real programming language is central to the book. While the specifications, which are addressed to human beings, should, and even must, combine rigor with appropriate informality, the code, which is addressed to the computer, must be absolutely precise even while being general. As with other areas of science and engineering, the appropriate foundation of programming is the deductive method. It facilitates the decomposition of complex systems into components with mathematically specified behavior. That, in turn, is a necessary precondition for designing efficient, reliable, secure, and economical software. The book is addressed to those who want a deeper understanding of program- ming, whether they are full-time software developers, or scientists and engineers for whom programming is an important part of their professional activity. The book is intended to be read from beginning to end. Only by reading the code, proving the lemmas, and doing the exercises can readers gain understanding of the material. In addition, we suggest several projects, some open-ended. While the book is terse, a careful reader will eventually see the connections between its parts and the reasons for our choice of material. Discovering the architectural principles of the book should be the reader's goal. We assume an ability to do elementary algebraic manipulations, l We also assume familiarity with the basic vocabulary of logic and set theory at the level of undergrad- uate courses on discrete mathematics; Appendix A summarizes the notation that we use. We provide definitions of a few concepts of abstract algebra when they are needed to specify algorithms. We assume programming maturity and understanding of computer architecture2 and fundamental algorithms and data structures) We chose C++ because it combines powerful abstraction facilities with faithful representation of the underlying machine) We use a small subset of the language and write requirements as structured comments. We hope that readers not already familiar with C++ are able to follow the book. Appendix B specifies the subset of the language used in the book? Wherever there is a difference between mathematical notation and C++, the typesetting and the context determine whether the mathe- matical or C++ meaning applies. While many concepts and programs in the book have parallels in STL (the C++ Standard Template Library), the book departs from some of the STL design decisions. The book also ignores issues that a real library, such as STL, has to address: namespaces, visibility, inline directives, and so on. Chapter 1 describes values, objects, types, procedures, and concepts. Chapters 2-5 describe algorithms on algebraic structures, such as semigroups and totally ordered sets. Chapters 6-11 describe algorithms on abstractions of memory. Chapter 12 describes objects containing other objects. The Afterword presents our reflections on the approach presented by the book.

<<编程的本质>>

内容概要

　　本书提供了有关编程的一种与众不同的理解。
其主旨是，实际的编程也应像其他科学和工程领域一样基于坚实的数学基础。
本书展示了在实际编程语言(如C++)中实现的算法如何在最一般的数学背景中操作。
例如，如何定义快速求幂算法，使之能使用任何可交换运算。
使用抽象算法将能得到更高效、可靠、安全和经济的软件。
　　这不是一本很容易读的书，它也不是能提升你的编程技能的秘诀和技巧汇编。
本书的价值是更根本性的，其终极目标是提升你对编程的洞察力。
要想从中大获裨益，你需要从头到尾认真学习：阅读代码，证明引理，完成练习。
到结束之时，你将看到如何把这里讨论的演绎式方法应用到你的程序中，保证你做出的软件部件能一起工作，并表现出它们所应该表现的行为。
　　书中给出的算法和需求针对某些被操作的类型。
有关这些描述的代码(也可以通过Web得到)采用C++的一个小子集书写，这样做是为了让所有有经验的程序员都能理解。
这个小子集可以看做一种特殊语言，是由Sean Parent和Bjarne Stroustrup一起设计的。
　　无论你是一位软件开发者，还是其他以编程作为一项重要活动的专业人员，或者是一名在校的学生，你都会逐渐理解本书的经验丰富的作者多年来一直在教授和阐释的道理：数学对于编程是绝好的东西，理论对于实际是绝好的东西。

作者简介

Alexander Stepanov于1967～1972年间在莫斯科国立大学学习数学，从1972年开始在苏联，1977年移民美国后在美国从事编程工作。
他编写过操作系统、编程工具、编译器和各种库。
他在程序设计基础方面的工作先后得到GE、Polytechnic、AT&T、惠普、Silicon Graphics的支持，2002年后是Adobe的支持。
1995年因C++标准模板库的设计获Dr.Dobb的程序设计杰出贡献奖。
Paul McJones于1967～1971年间在加州大学伯克利分校学习工程数学。
从1967年开始介入程序设计，涉足的领域包括操作系统、程序设计环境、事务处理系统以及企业和客户应用系统等。
他先后在加州大学、IBM、Xerox、Tandem、DEC工作，2003年至今在Adobe公司。
1982年他与合作者一起因其论文“ The Recovery Manager of the System R Database Manager ”获得ACM程序设计系统和语言论文奖。

书籍目录

章节摘录

插图：An object is passed directly flit is passed as an argument or returned as the resultand is passed indirectly if it is passed via a pointer or pointerlike object. An object isan input to a procedure if it is read, but not modified, by the procedure. An object isan output from a procedure if it is written, created, or destroyed by the procedure,but its initial state is not read by the procedure. An object is an input~output of aprocedure if it is modified as well as read by the procedure.A computational basis for a type is a finite set of procedures that enable theconstruction of any other procedure on the type. A basis is efficient if and onlyif any procedure implemented using it is as efficient as an equivalent procedurewritten in terms of an alternative basis. For example, a basis for unsigned k-bitintegers providing only zero, equality, and the successor function is not efficient,since the complexity of addition in terms of successor is exponential in k.A basis is expressive if and only if it allows compact and convenient definitionsof procedures on the type. In particular, all the common mathematical operationsneed to be provided when they are appropriate. For example, subtraction could beimplemented using negation and addition but should be included in an expressivebasis. Similarly, negation could be implemented using subtraction and zero butshould be included in an expressive basis.

媒体关注与评论

“ 要是问一位机械、建筑或电子工程师，如果不依靠坚实的数学基础，他们能走多远。
他们会告诉你‘ 走不了多远 ’。
而所谓的软件工程师在实践其技能时，却常常对他们所做工作的数学基础知之甚少，甚至一无所知。
同时我们也很奇怪为什么软件由于不能按时发布并充斥错误而声名狼藉，而其他工程师却能按时完成其桥梁、汽车、各种电子装置等，而且有很少的缺陷。
本书就是想纠正这种不平衡现象。
我在Adobe的高级开发团队的成员们，但凡参加了基于同样材料的课程，都觉得付出的时间获益匪浅。
初看可能觉得这种高度技术性的文字只是为计算机科学家写的，其实所有从事实际工作的软件工程师都应该来读。
”　　——Martin Newell，Adobe 院士 “ 本书包含一些我所见过的最美的代码。
”　　——Bjarne Stroustrup，C++ 设计者 “ 我很高兴看到Alex课程的内容。
作为Silicon Graphics的CTO时，我曾大力支持这一课程的开发和教授，现在这本书已经能被所有程序员阅读了。
”　　——Forest Baskett，合伙人，New Enterprise Associates “ Paul的耐心和在体系结构方面的经验帮助把Alex的数学方法组织成为一套高度结构化的大厦——功德无量！
”　　——Robert W. Taylor，Xerox PARC CSL和DEC系统研究中心创始人

编辑推荐

《编程的本质(英文版)》：经典原版书库

<<编程的本质>>