<<程序设计语言>>

图书基本信息

书名：<<程序设计语言>>

13位ISBN编号：9787121042980

10位ISBN编号：7121042983

出版时间：2007-6

出版时间：电子工业出版社

作者：斯科特

页数：899

译者：裘宗燕

<<程序设计语言>>

内容概要

★列为全球上百所大学标准教材和首席参考书！

★图书馆必备典藏,作者Michael L.Scott 是计算机领域的著名学者，译者是北京大学的裘宗燕教授，他熟悉专业，译笔流畅，因此，这是一本难得的著、译双馨的佳作。

这是一本很有特色的教材，其核心是讨论程序设计语言的工作原理和技术。

本书融合了传统的程序设计语言教科书和编译教科书的有关知识，并增加了一些有关汇编层体系结构的材料，以满足没学过计算机组织的学生们的需要。

书中通过各种语言的例子，阐释了程序设计语言的重要基础概念，讨论了各种概念之间的关系，解释了语言中许多结构的形成和发展过程，以及它们演化为今天这种形式的根源。

书中还详细讨论了编译器的工作方式和工作过程，说明它们对源程序做了什么，以及为什么要那样做。

书的每章最后附有复习题和一些更具挑战性的练习。

这些练习的特别价值在于引导学生进一步深入理解各种语言和技术。

本书第2版新增了脚本语言问题的讨论，涵盖Perl、Python、Ruby、Tcl、PHP、JavaScript、XSLT等和其他语言。

本书在美国大学已使用了十余年，目前被欧美许多重要大学用于“程序设计语言”或者“软件系统”课程。

本书适合高年级本科生或者一年级研究生使用，许多内容对专业程序员也很有价值。

作者简介

Michael L. Scott是罗切斯特大学计算机科学系的教授，前任系主任。
他于1985年获得麦迪逊的威斯康星大学博士学位。
他是Lynx分布式程序设计语言、Charlotte和Psyche并行操作系统、Bddge并行文件系统、Cashmere和InterWeave分布式共享存储系统和许多广泛使用的同步算法和并发数据结构的设计者或共同设计者。
2001年获得该大学的”Rohert和Pamela Goergen本科生教学突出贡献奖”。
. 裘宗燕，北京大学数学学院信息科学系教授。
长期从事计算机软件与理论、程序设计语言方面的研究和教学工作。
先后翻译了多本国外计算机科学技术领域的经典名著，包括《程序设计语言——实践之路》、《C++程序设计语言（特别版）》、《计算机程序的构造和解释（第二版）》，《程序设计实践》等，负责了《代码大全（第2版）》的审校工作，深得国内读者好评。
...

<<程序设计语言>>

书籍目录

媒体关注与评论

《程序设计语言——实践之路》不仅用极清晰的笔触解释了语言的各种概念和实现细节，还仔细解释了计算机体系结构和编译器如何影响语言的设计和实现……。
本书展示了程序设计语言如何居于计算机科学真正的中心，是跨越程序员和机器之间的深渊的桥梁。
——摘自Microsoft Research的James Larus写的前言　　新版本的《程序设计语言——实践之路》是平衡教科书所需的三个质量要素（广度、深度和清晰性）的典范，它必然成为这一领域里的经典。
——Christopher Vickery, Queens College of CUNY　　Michael Scott的Programming Language Pragmatics是一本很有价值的教科书，其内容涵盖程序设计语言、编译技术、软件系统的许多方面，甚至延伸到硬件体系结构等许多领域。
出现这一情况的根源很明显：程序语言在计算机科学技术领域居于一种中心地位。
程序是计算机科学技术里最核心的概念，而作为描述程序的语言，集中体现了程序设计和软件开发实践中形成的最有价值、最具普遍性的认识和技术。
程序语言下接硬件体系结构，上承丰富多彩的计算机应用需求，既反映了开发者的专业发展和局限性，又受到实现的制约。
这样，程序语言里很自然地浓缩了相关领域的大量知识和技术精华，要理解语言发展和演化的现状和趋势，也必然涉及与之相关的众多领域。
本书作者熟悉这些相关领域，因此能在其中纵横驰骋，为我们展现了一幅有关程序语言的生动、全面，而又非常深刻的画卷。
.　　本书系统地介绍了程序语言领域的各种基本概念，介绍了语言处理方面的许多知识，不同的语言范型以及相关的理论和实践。
在讨论各种语言特征时，特别仔细地考察了人们的评价和反思，阐释了各种特征的设计变化，以及理论和技术发展对语言形态和细节的影响。
与此同时，本书还深入介绍了本领域的许多新发展、新问题和新技术。
例如，作者用一章的篇幅深入探讨了面向对象语言的问题，不仅介绍了这类语言的外在形式特征及其价值，还特别仔细地讨论了这类语言中各种新的重要机制的实现技术，如动态方法约束、多重继承等等。
用很长的一章深入讨论了并行性的历史和发展，以及与并行性有关的各种重要问题。
第2版新增了有关脚本语言的一章，其中讨论了脚本语言的特点、应用和许多深入问题。
书中还详细讨论了高级语言的加工过程，程序的静态连接和动态连接，以帮助读者理解这方面的情况。
作者在书中既强调了重要的概念和理论，也特别重视各种特征的实现技术，并深入探讨了实现技术及其发展进步对程序语言设计的影响。
应该看到，语言实现方面的许多技术都是最重要的程序技术，作者的这些想法也使本书成为一部很有价值的软件技术书籍。
..　　本书新的第2版增加了许多内容，特别反映了程序设计语言领域最近几年的新发展。
例如书中特别讨论了C99的许多新特征，讨论了Java和C# 的泛型机制，Java等语言最新的并行库的重要特征，Python和Ruby语言的面向对象模型和高级结构。
本书的内容也大大增加了，配套光盘里包含一些问题的深入讨论，还有许多很有价值的参考材料。
总而言之，本书在许多方面有鲜明特色，是程序设计语言领域教科书的最新代表。
本书出版之后受到人们的广泛关注，很快被世界各国的许多重要大学选为“程序设计语言”或相关课程的教科书或重要参考书。
我认为，这本书不仅值得计算机专业的本科生或研究生学习，也值得计算机领域的实践工作者们阅读。
对本书的学习能帮助我们理解现有的各种程序设计语言，提高学习和掌握新语言的能力，还能帮助我们看清隐藏在高级语言的各种机制背后的许多深入的东西，理解各种重要语言特征的价值、缺陷和使用它们的代价。

对程序设计语言的深入理解，对于计算机专业工作者深刻理解相关的理论和实践，灵活高效地运用程序语言和相关工具，都可能起到非常重要的作用。
　　我非常赞赏电子工业出版社引进这本很有价值的著作，也很高兴能在把本书介绍给国内读者方面做些工作。
我感谢周筠女士的远见卓识和陈元玉编辑的辛勤工作，还要感谢夏萍和丛欣对我的一贯支持和帮助。
本书涉及领域广泛，其中许多是我不熟悉的。
虽然我已尽可能查阅了一些材料，但书中难免还会留下许多缺陷，希望得到同行和读者的指教。
　　裘宗燕　　2007年5月，于北京大学　　序言　　计算机科学惯于在抽象之上建立抽象。
在我们的领域里，把细节隐藏到简化界面之下的功能既是一种利器，也是不得已而为之。
操作系统、数据库和编译器都是非常复杂的程序，历经40年理论和开发的磨砺。
在绝大多数情况下，为使用一个软件提供的功能，程序员很少或根本不必理解其内部逻辑或结构。
在大部分情况下，不知这是祸是福。
.　　然而，模糊的抽象也可能变成阻碍发展和进步的壁垒，而不是新生事物的可靠基础。
请看一看本书考察的主题：程序和程序设计语言。
为什么一个程序运行得慢如蜗牛，而从执行剖析却看不出其中有任何明显的瓶颈，或者无法给其瓶颈找到一种算法解释？
有些潜在问题根源于从语言结构到机器指令的翻译，或是所生成代码与处理器体系结构的交互作用。
要解决这类问题，就需要理解连接不同抽象层次的桥梁。
　　抽象也同样出现在学习的道路上。
简单的问题如：用英语的一个小小的呆板子集写出的程序如何控制只懂得二进制语言的机器？
或者，虽然有种类和数目繁多的程序设计语言，但为什么它们看起来都相当类似？
如果不潜入细节，不理解计算机、编译器和语言，这类问题也无法回答。
　　作为一个整体，计算机科学的教育确实能回答这些问题。
大多数本科生教学计划都提供了关于诸如计算机体系结构、操作系统、程序设计语言设计和编译器的课程。
所有这些美妙课程都很值得去学，但是却很难融入一个本科生的计算机科学教学计划里，因为它还要提供其他的丰富内容。
进一步说，许多课程教授的都是一些独立的主题，却很少解释一个主题与其他主题之间的联系。
..　　本书采用另一种视角，超越了分隔上述种种论题的抽象，从而能很好地回答上面这类问题。
Michael Scott是一位卓越的研究者，在语言实现、运行时系统和计算机体系结构方面都有杰出建树。
他特别有资格描绘所有这些领域，提供有关现代程序设计语言的一种具有内在统一性的理解。
本书不仅用极其清晰的笔调解释了语言的各种概念和实现细节，也阐释了计算机体系结构和编译器如何影响语言的设计和实现。
进一步说，它清晰地展现了各种不同语言在实际中如何使用，用一些很实际的例子说明问题领域如何打磨着这些语言。
　　在阅读本书所感兴趣的问题的同时，我必须承认当第一次读这本书时，我也曾感到担忧。
那时我担心Michael的方式会削弱教学计划中程序设计语言和编译器的地位，使学生对这一领域的理解表面化。
但现在，通过重读这本书，我已经认识到事实恰恰与我的担心相反。
通过把相关的论题放在合适的位置上展示，本书很好地说明了程序设计语言如何居于计算机科学真正的中心，成为跨越程序员和机器之间的深渊的桥梁。
　　James Larus，Microsoft Research　　计算机程序设计的课程给了普通学生有关计算机领域的第一个印象。
在上这种课程之前，大多数学生已经在自己的生活中使用着计算机，用于诸如电子邮件、计算机游戏、浏览网页、做文字处理、即时消息（聊天），以及大量其他事项，而且在他们还没有写出自己的程序之前，就已经开始关注这些应用系统的工作方式了。
在获得了作为程序员的一定能力之后（假定已经学过很好的有关数据结构和算法的课程），很自然地

，下一步就是想知道程序设计语言是如何工作的。
本书就是对此提供一个解释。
本书的目标很简单，就是采用尽可能容易理解和最精确的语言，采用普通本科生愿意阅读并易于接受的风格。
这一目标反应了我的一种信念：如果我们真能很好地解释一件事情究竟是怎样的，学生总是希望想理解更多东西，并乐于去接触更多的材料。
. 在常规的有关“系统”的教学计划里，数据结构（或者再加上计算机组织）之后的内容被分门别类归入属于一些子领域的一批课程，如程序设计语言、编译构造、计算机体系结构、操作系统、并行和分布式计算、数据库管理系统，可能还有软件工程、面向对象的设计、图形学，或者用户界面系统。
这种安排方式存在一个缺点：这一科目列表在不断地增长，而学士课程教学计划中的学期数却保持不变。
或许更重要的是，有关计算机科学最有趣的许多东西都处在这些科目之间的边界上。
例如，RISC革命造成计算机体系结构和编译器构造之间持续20年的联盟。
最近几年，重新热起来的对虚拟机的兴趣使操作系统内核与语言的运行时系统之间的分界线变得很模糊了。
Java和 .NET也以类似的形式模糊了编译器和运行库之间的分野。
今天许多程序被常规地嵌在万维网页、电子报表和用户界面里。
与此同时，教育工作者和研究者也越来越认识到必须关注这些相互关系。
特别是在高等教育的核心教学计划中出现一种集成的趋势。
许多学校不是给普通学生有关两三个狭窄科目的深入探讨，同时又在其他方面留下很大的空缺，而是重整了有关程序设计语言和操作系统的课程，使之涵盖范围更广泛的科目，再提供一些更专门的后续课程。
这一趋势在很大程度上是ACM/IEEE计算教程2001所提出的认识的发展。
在这一教程中强调了本领域的成长，对于广度日益增长的需要，教学计划设计灵活性的重要意义，以及对于毕业生的总体目标：“必须有一种系统层面的认识，该认识应适于帮助理解理论和实践之间的相互作用，熟悉常见的研究课题，并能随着本领域的发展而更新”[CR01，11.1节，有修改]。
《程序设计语言——实践之路》的第1版很幸运地跟上了这种趋势。
第2版继续并加强了对集成学习的重视程度，并继续以关于程序设计语言设计的讨论为中心。
本书的核心就是讨论程序设计语言是如何工作的问题。
其中并没有去列举许多不同语言的细节，而是集中关注学生们可能遇到的有关所有语言之基础的一批概念，通过各种各样的实际例子阐释这些概念，并努力探索解释为什么不同语言的设计采纳了不同方式背后的那些利弊权衡。
同样，本书也不去讨论如何构造一个编译器或者解释器（那只是极少数的程序员最终需要整个地参与的一种工作），它将集中关注编译器对输入的源程序做了些什么事情，以及为什么要那样做。
语言的设计和实现在这里被放在一起考察，其中特别强调它们之间相互作用的各种方式。
第2版的变化 Changes in the Second Edition 第2版有4个目标： 1. 引进一些新材料，最主要的是有关脚本语言的材料。
2. 更新内容，使之反映过去六年中发生的各方面情况。
3. 减少提高书价格的压力。
4. 从教育学的观点加强这本书，使之更容易使用，更容易理解。
项（1）是内容上最重要的变动。
由于万维网的迅猛扩张，我们可以看到如Perl、PHP、Tcl/Tk、Python、Ruby、JavaScript和XSLT等语言的巨大发展，不仅在其商业重要性方面，也在其设计创新方面。
今天的许多毕业生将把他们更多的时间花到脚本语言的工作上，可能比C++、Java和C# 更多。
有关脚本语言的一个新章（第13章）首先是按应用领域组织的（外壳语言，文字处理和报表生成，数学和统计，“粘接”语言和通用脚本语言，扩充语言，以及万维网脚本），而后再按新特征组织的（

名字和作用域，字符串和模式匹配，高级数据结构，面向对象）。
对于脚本语言的讨论也被加入正文中其他所有合适的地方。
项（2）反映了一些关键性的发展，如C99标准的最终完成，Java 5和C#（版本2.0）的出现。
第6章（控制流）现在包括了装盒和开盒，以及最新的迭代器结构。
第8章（子程序）讨论了Java和C# 的泛型。
第12章（并发性）涵盖了Java 5的并行库（JSR 166）。
对C# 的讨论也加入了通篇中各个合适的地方。
为了反映微处理器市场的变化，在第5章（体系结构）和第8章（子程序）的实例研究中，无处不在的Intel/AMD x86取代了Motorola 68000。
第8章里有关MIPS的实例研究也提升到64位模式。
有关技术指标和趋势的讨论也都做了更新。
在一些地方，我重写了使用语言的例子，采用了一些学生可能更熟悉的实例。
这一过程无疑还将在今后的版本中继续进行。
第2版里许多章节做了大量修改，以使之更清晰并更准确。
有关章节包括有穷自动机的创建（2.2.1节），声明顺序（3.3.3节），模块（3.3.4节），别名和重载（3.6.1节和3.6.2节），多态性和泛型（3.6.3节、7.1.2节、8.4节和9.4.4节），分别编译（3.7节），延续、异常和多层返回（6.2.1节、6.2.2节和8.5节），调用序列（8.2节）以及第5章的几乎全部内容。
项（3）反映了Morgan Kaufmann关于使教科书具有学生容易接受的价格的承诺。
第1版比与之可比的教科书更大而且内容更全面，但售价却更低。
第2版用高质量的平装形式保持了可接受的价格（同时减小了体积）。
.. 最后是项（4），体现在大量表现方式的改变上。
其中一些改变很细微，例如给出了更多的小节标题，以及更多历史事实。
更重要的是将本书组织为四个主要部分： 第1部分 包含最基础的材料：（1）有关语言设计和实现的简介；（2）程序设计语言的语法；（3）名字、作用域和约束；（4）语义分析；（5）目标机体系结构。
这里的（2）和（5）以合理的方式集中关注实现方面的问题，（1）和（4）涉及面较广，（3）介绍了语言设计中的一些核心问题。
第2部分 继续涵盖其他的核心问题：（6）控制流；（7）数据类型；（8）子程序和控制抽象；（9）数据抽象和面向对象。
最后一章从其在第1版里的位置前移了，反映了面向对象程序设计在现代计算中的中心地位。
第3部分 转向其他程序设计模型：（10）函数式语言；（11）逻辑式语言；（12）并发；（13）脚本语言。
在第1版里函数式和逻辑式语言共处一章。
第4部分 转向语言的实现：（14）构造可执行程序（代码生成、汇编和连接）；（15）代码改进（优化）。
配套光盘 The PLP CD 为了减小本教科书的物理体积，为新材料让出位置，使学生可以在浏览本书时集中关注最基本的材料，大约250页更高级的或者更外围的材料被放入随书配套的光盘里。其中的大部分章节（不是全部）是在第1版中被标为高级的或可选的内容。
其中最重要的变动是将有关代码改进的整个第15章移到光盘里，移入的其他材料是一些散布各处的节或小节。
这样的每节都在教科书正文中有所表现，有一段关于相关论题的简单介绍，以及一个“深度探索”段落，其中综述了该处省略的材料。
请注意，把这些材料放在光盘并不意味着对它们技术重要性的评价，而只是反映了一个事实：最值得包含的材料已经超过了一卷书或者一门课程的容量。
我的意图是把那些大部分课程可能涵盖的材料放在印刷形式的教科书中。
关于设计和实现的旁白 Design & Implementation Sidebars 本书第1版特别强调了语言设计对于实现选择的影响，以及预期的实现方式对于语言设计的影响。

第2版用超过120个旁白把这方面的联系表现得更清楚了。
有关这种旁白的更多细节将在第7页（第1章）说明。
附录B给出了所有旁白的目录。
编号并加标题的实例 Numberde and Titled Examples 第2版中的实例被更明确地编织在讨论中。
为使读者更容易找到特定的实例，记住它们的内容，并更容易在其他地方引用这些实例，现在每个实例都有一个页边注，标明其编号和标题。
全文和配套光盘里大约有900个这样的实例，附录C是有关它们的详细目录。
练习的安排 Exercise Plan 第1版包含总共385个复习题和312个练习，都放在各章最后。
在第2版里，复习题被移到各节的最后，更接近它们所涉及的内容，使读者更容易弄清自己是不是抓住了最重要的概念。
复习题的数目大约增加了一倍。
习题还是放在各章最后，现在分为练习和探索两类。
前者倾向于或多或少更加直截了当一些，当然比每节最后的复习题更有挑战性。
它们适合作为课下作业或简单课题。
探索中的问题是更开放的，需要做一些基于网络或图书馆的研究工作，花费明显更多的时间；或者是去形成一种客观的认识。
习题总数从第1版的略多于300道扩充为这一版的大约500道。
对于注册的教师，可以从密码保护的网址得到习题解答（不包括探索题），需要者请访问www.mkp.com/companions/ 0126339511/。
如何使用本书 How to Use the Book 《程序设计语言——实践之路》涵盖了计算教程2001报告 [CR01] 中PL“知识单元”的几乎所有材料。
本书特别适合CS 341模块课程（程序设计语言设计），也被用于CS 340（编译构造）或CS 343（程序设计范型）。
它包括了CS 344（函数式程序设计）和CS 346（脚本语言）中相当大一部分内容。
图1给出的是使用本教程的几条可能路径。
作为自学或者一学年的课程（图1里的轨迹F），我建议从头到尾通读本书，当遇到每个“深度探索”段落时再转到相应的光盘内容。
在罗切斯特大学作为一学期的课程时（轨迹R，本书就是为它开发的），我们也覆盖了本书的大部分内容，但不包括大部分光盘章节。
再去掉自下而上的语法分析（2.3.3节）、消息传递（12.4节）、万维网脚本（13.3节）以及第14章（构造可运行程序）的大部分内容。
有些章（第2、4、5、14和15章）比其他的章更着重强调实现问题。
这些章与其他更多关注语言设计的章之间的顺序可以在一定限度内调整，但一定要保证第5章或者与之相当的内容出现在第6到9章之前。
许多学生可能已经熟悉了第5章的一些材料，多半是来自一个有关计算机组织的课程。
在这种情况下就可以简单地跳过这一章，将其作为参考材料。
有些学生可能已经熟悉了第2章的一些材料，可能是来自一门有关自动机的课程。
在这种情况下，该章的内容可以很快读过去，只是可能要在某些实际问题上（例如从语法错误中恢复，或者实际扫描器与经典的有穷自动机的差异方面）多停留一下。
如果用于更传统的程序设计语言课程（轨迹P），那么可以排除有关扫描器和语法分析器的内容，加上完整的第4、5两章，还可以减少对其他章节里与实现有关的材料的重视程度。
在此基础上可以增加一些光盘内容，如ML类型系统（7.2.4节），多重继承（9.5节），Smalltalk（9.6.1节），lambda演算（10.6节）和谓词演算（11.3节）。
图1 课程路线图。
较暗的区块表示配套光盘中的“In More Depth”补充内容。
节序号表示没有与补充内容对应的断开的部分。

本书也在一些学校里被用于有关编译器的引论性课程（图1的轨迹C）。
典型的安排是基本去掉第3部分的内容（第10到13章），以及通篇中更多强调设计的材料。
包含所有有关扫描和语法分析的材料，第14和15章，以及其他光盘材料的某种混合。
对于那些采用4学季制的学校，一种常见选择方案是开一学期的引论性课程和两个随后的选修课程（图1的轨迹Q）。
引论课程可以覆盖第1、3、6、7章的主要内容（不包括光盘材料），再加上第2和8章的前一半。
后面一学季的面向语言的课程可以覆盖第8章的其余部分，整个第3部分，以及第6到8章的光盘材料。
或许再加上一些有关形式语义学、类型系统或其他相关论题的补充材料。
后面一学季的面向编译器的课程可以包含第2章的其余部分，第4—5章和14—15 章，以及第3章和第8~9章的光盘材料，或许再加上一些有关自动代码生成、更富进取性的代码优化、程序设计工具等等方面的材料。
无论采用哪条路径学习这本教科书，我都假定大多数读者已经对至少一种高级命令式程序设计语言有了相当的经验。
具体是哪种语言在这里并没有关系。
书中例子取自各种不同的语言，但总是带有足够的注释和其他讨论，使不熟悉该语言的读者也能比较容易地理解它们。
附录A包含了有关大约50种语言的简单介绍。
这里的算法都是采用自明的非形式的伪代码。
真正的程序设计语言代码用的字体是"typewriter"字体，伪代码用的字体是sans-selif字体。
辅助材料 Supplemental Materials 除了作为正文章节的补充外，随书光盘里还包括一些其他资源： · 对万维网上一些语言手册和教程的链接； · 对一些开源编译器和解释器的链接； · 书中所有不那么简单的实例的完整源代码（多于300个源文件）； · 对文本内容和随书光盘内容的搜索引擎。
进一步的资源在www.mkp.com/companions/0126339511/（你可能需要随时过去检查，以保证链接正确）。
对于采用本书的教师，还通过一个密码保护的页面提供了： · 全书中所有图形的可编辑的PDF源文件； · 可编辑的PowerPoint幻灯片； · 大部分练习的解答； · 对于大项目的建议。
第2版致谢 Acknowledgments for the Second Edition 在准备第2版的过程中，我有幸得到了很多人的真诚帮助。
许多同行提供了有关第1版中的勘误和反馈信息，包括Manuel E. Bermudez，John Boyland，Brian Cumming，Stephen A. Edward，Michael J. Eulenstein，Tayssir John Gabbour，Tommaso Galleri，Eileen Head，David Hoffman，Paul Ilardi，Lucian Ilie，Rahul Jain，Eric Joanis，Alan Kaplan，Les Lander，Jim Larus，Hui Li，Jingke Li，Evangelos Milios，Eduardo Pinheiro，Barbara Ryder，Nick Stuifbergen，Raymond Toal，Andrew Tolmach，Jens Troeger和Robert van Renesse。
裘宗燕做了本书的中文翻译并在此过程中发现了一些错误。
Simon Fillat维护了Morgan Kaufmann的网页。
无疑地我还要感谢列在第1版致谢中的另外许多人，他们作为评阅人、采用者和读者，为前一版的工作提供了许多帮助并使其成功。
他们的贡献延续到这一版中。
对于第2版的工作开始于SIGCSE02的“关注小组”的认真讨论。
我特别要感谢Denise Penrose、Emilia Thiuri和Morgan Kaufmann团队的其他人组织了这次活动，感谢那二十多位出席这次活动并使我能共享他们有关内容和教学法的意见的同行，以及其他评阅了两个版本的计划书的人们。
第2版的初稿于2004年秋季在8所大学做了课堂试验。
我非常感谢Gerald Baumgartner（路易斯安那州立大学）、William Calhoun（Bloomsburg大学）、Betty Cheng（密执安州立大学）、Jingke Li（Portland州立大学）、Beverly Sanders（佛罗里达大学）、Darko

Stefanovic（新墨西哥大学）、Raymond Toal（Loyola Marymount大学）、Robert van Engelen（佛罗里达州立大学）和他们的学生提供的大量建议、反映、勘误和其他反馈。
van Engelen教授提供了若干绝好的章末练习。
　　第2版的外部评阅人提供了大量有益的建议。
我要感谢Richard J. Botting（加利福尼亚州立大学San Bernardino分校）、Kamal Dahbur（DePaul大学）、Stephen A. Edwards（哥伦比亚大学）、Eileen Head（Binghamton大学）、Li Liao（特拉华大学）、Christopher Vickery（纽约城市大学皇后学院）、Garrett Wollman（MIT）、Neng-Fa Zhou（纽约城市大学布鲁克林学院）和Cynthia Brown Zickos（密西西比大学）。
Garrett Wollman有关第13章的评论特别有帮助，就像他对前一版本的各方面问题提出的评注一样。
我已将其中尽可能多的东西融入这一版本，并认真地保存起剩下的意见，留作第3版的写作指引。
在本版里剩下的问题都应该由我本人负责。
　　这个版本也于2004年秋季在罗切斯特大学做了教学试验。
我要感谢我的学生，特别是John Heidkamp、David Lu和Dan Mullowney的热情和建议。
Mike Spear对第13章的Web技术提供了若干有益的建议。
在这些年里，我的同事Chen Ding和Sandhya Dwarkadas用第1版授课多次，提出了许多有益的建议。
Chen对第15章的建议（也有Yutao Zhong的协助）特别有价值。
我还要感谢我的其他同事，以及系主任Mitsunori Ogihara。
感谢系里的管理员、秘书和技术人员为我提供了这么好的工作环境。
　　正如在第1版工作中一样，Morgan Kaufman出版社的工作人员对此工作真诚投入，包括专业上的和个人的。
我特别感谢出版人Denise Penrose，编辑Nate McFadden，产品编辑Carl Soares，光盘设计师Peter Ashenden，市场经理Brian Grimm和编辑助理Valerie Witte。
　　最重要的是很感激我的妻子Kelly和我们的女儿Erin和Shannon，感谢她们在这看不到尽头的写作和修改中的耐心和支持。
计算是一种美好的专业，而家庭却是最要紧的。
　　Micheal L. Scott　　Rochester, NY　　2005年4月

<<程序设计语言>>

编辑推荐

这是一本很有特色的教材，其核心是讨论程序设计语言的工作原理和技术。
本书融合了传统的程序设计语言教科书和编译教科书的有关知识，并增加了一些有关汇编层体系结构的材料，以满足没学过计算机组织的学生们的需要。
书中通过各种语言的例子，阐释了程序设计语言的重要基础概念，讨论了各种概念之间的关系，解释了语言中许多结构的形成和发展过程，以及它们演化为今天这种形式的根源。
书中还详细讨论了编译器的工作方式和工作过程，说明它们对源程序做了什么，以及为什么要那样做。
书的每章最后附有复习题和一些更具挑战性的练习。
这些练习的特别价值在于引导学生进一步深入理解各种语言和技术。
本书第2版新增了脚本语言问题的讨论，涵盖Perl、Python、Ruby、Tcl、PHP、JavaScript、XSLT等和其他语言。
本书在美国大学已使用了十余年，目前被欧美许多重要大学用于“程序设计语言”或者“软件系统”课程。
本书适合高年级本科生或者一年级研究生使用，许多内容对专业程序员也很有价值。
本书作者Michael L.Scott 是计算机领域的著名学者，译者是北京大学的裘宗燕教授，他熟悉专业，译笔流畅，因此，这是一本难得的著、译双馨的佳作。

版权说明