ELSEVIER

Broadview®
www.broadview.com.cn

Programming Language Pragmatics (Third Edition)

程序设计语言——实践之路 第3版

[美] Michael L.Scott 著　韩江　陈玉 译

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

Programming Language Pragmatics (Third Edition)

程序设计语言——实践之路 第3版

[美] Michael L.Scott 著　韩江　陈玉 译

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内容简介

这是一本很有特色的计算机教材，其核心是讨论程序设计语言的基本原理和技术。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与探索。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3版新增了关于运行时程序管理的讨论，对关于并发的一章做了重大的改写，并更新了大量的实例。

这本教材在美国大学已使用了二十余年，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。

版权贸易合同登记号　图字：01-2010-0228

图书在版编目（CIP）数据

程序设计语言：实践之路：第3版 /（美）斯科特（Scott,M.L.）著；韩江，陈玉译. —北京：电子工业出版社，2012.7

书名原文：Programming Language Pragmatics, Third Edition

ISBN 978-7-121-17067-6

Ⅰ.①程… Ⅱ.①斯… ②韩… ③陈… Ⅲ.①程序语言—教材 Ⅳ.①TP312

中国版本图书馆CIP数据核字（2012）第099275号

策划编辑：刘　皎
责任编辑：许　艳
特约编辑：顾慧芳
印　　刷：北京丰源印刷厂
装　　订：三河市鹏成印业有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路173信箱　　邮编：100036
开　　本：787×980　1/16　　印张：55　　字数：1135千字
印　　次：2012年7月第1次印刷
定　　价：128.00元（含光盘1张）

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至zlts@phei.com.cn，盗版侵权举报请发邮件至dbqq@phei.com.cn。

服务热线：（010）88258888。

结合配套光盘使用本书

《程序设计语言——实践之路》包括一张配套光盘，其中包含了一些高级的或者可选的主题，以及完整的第5章——目标机体系结构和第16章——代码改进。这个配套光盘的内容在正文中以几种不同的方式引用或标识：

- **“深度探索”段落**

  在“深度探索”段落对可选读内容进行了简要的概述。

- **目录表**

  光盘内容包含在目录表中，使读者可以很方便地识别出位于光盘中的章节内容。目录条目的形式如下所示：

  2.4 理论基础 ..... ◎13 100
  2.4.1 有穷自动机 ..... ◎13
  2.4.2 下推自动机 ..... ◎18

  这里你可以看到第2.4节在正文98页有简单介绍（在该页还可以找到“深度探索”简介段落），而该节及其小节的全部内容则在配套光盘中的第13至18页。

- **引用点**

  在正文中的标记◎表示对配套光盘中“深度探索”内容的引用。

- **练习**

  用于个人自学、高阶的挑战性问题或研究项目，在正文中标为“深度探索”作为练习。

- **第5章和第16章的全部内容**

  整个第5章和第16章的内容都以单独的PDF文件的形式存放在配套光盘中。这两章在正文中有简介，也包含在目录中。

译者序

Michael Scott的《程序设计语言——实践之路》（*Programming Language Pragmatics*）是一本优秀的教科书。在网络时代计算技术飞速发展的背景下，各种创新的软、硬件设计理念及实践如雨后春笋般层出不穷，如何将大量的信息组织起来，并突出其核心内容，是类似教材的作者们所面临的现实挑战。而Scott这本教材最大的特点就在于，它紧紧抓住了处于计算机科学技术领域中心位置的主题——程序设计语言，为读者深入地探讨了程序设计语言及其实现的关键概念，同时将笔触延伸到编译技术、软件系统，甚至软、硬件体系结构等诸多领域。

本书系统地介绍了与程序设计语言相关的各种基本概念，内容涉及语言处理方面的具体细节，以及各种语言范式。作者在讨论各个主题时，将对语言概念的描述与如何实现这些概念的具体说明从整体上结合在一起，讲解清晰，并且给出了大量的实例。这样，就使得读者通过学习本书，不仅能“知其然”——了解各种程序设计语言的实现中所做出的具体选择，还能够“知其所以然”——了解这些语言的设计者做出这些选择背后的逻辑和取舍。

本书的编排方式也非常灵活，各章的内容相对独立，每章都有大量随堂练习和课后习题，以帮助读者巩固学到的知识，并启发他们做进一步的思考。随书附带的光盘上还包含了许多较深入的内容。根据需要，这本教科书可以通过不同的方式组织起来，用于侧重点不同的教学课程，读者自学时也可以根据自己的情况自由选读。

作者在本书（第3版）中对内容做了大量的更新，最明显的就是增加了关于运行时程序管理的全新的第15章，以及关于并发的第12章的很多改写，增加了一些较新的主题。这一版本更新了大量的实例（例如用X86上的C代码代替了Pascal），并且更多地采用了C#、Java 5、Python和Eiffel等现代语言的例子。作者对第3版的改进还体现在许多细节方面，足以反映出作者之用心。

本书在翻译的过程中，得到了博文视点总经理郭立老师的支持与指导，她对IT专业书籍翻译的质量非常重视，使我们对本书的翻译不敢有丝毫的懈怠，力求达到精益求精。感谢本书的策划编辑刘皎，是她持续的鼓励给予我们信心让我们继续下去。还要感谢那些为本书的翻译做出贡献的所有人，他们都是默默无闻的后台工作者。

参加本书翻译的人员除封面署名外，还有郭大权、丁素菊、王鹏、谭丹丹、韩澎、王嫣、唐立刚、郝强、刘晋东。诚然，限于译者水平，书中难免含有一些错误和理解不到位的地方，敬请读者朋友批评指正。我们的联系方式是Changchuan@gmail.com。

译者

2011年中秋节于北京

本书所获赞誉

计算机在21世纪的日常生活中已经无处不在，这说明程序设计语言处于计算机科学教育的中心是合理的。程序设计语言将计算机科学的理论基础（各种问题解决算法的来源）和现代计算机体系结构（相应的程序在其上生成解决方案）联系在一起。在后Internet时代，计算技术的发展日新月异，关于计算的教科书的结构必须组织起关于某个主题的信息，而不只是讨论这个主题。在这本书中，Michael Scott为读者广泛并全面地呈现了程序设计语言及其实现的关键概念，其方式非常适合计算机科学这一主题。

——摘自Barbara Ryder，Virginia Tech所写的序言

《程序设计语言——实践之路》是一本关于语言设计和实现的出色的入门书。它不仅展示了支撑我们所使用的语言的理论，还展示了计算机体系结构的发展对这些理论的指引，以及这些理论继续发展以面对利用多核硬件的挑战的方式。

——Tim Harris，微软研究院

Michael Scott为我们呈现了一本名副其实的著作——《程序设计语言——实践之路》。除了覆盖传统的语言论题之外，这本书还深入到有时比较模糊，但总是必需的各领域的程序设计细节。本书的这一新版本依然覆盖了现代语言的基础，并且包含关于现代运行时环境（包括虚拟机）的新材料和更新的内容。对于任何希望为现实世界的应用开发语言的读者来说，这本书都不啻为一本绝佳的入门读物。

——Perry Alexander，Kansas大学

Micheal Scott对《程序设计语言——实践之路》一书的新版本从整体和局部两个方面做了改进。其中的变化包括增加了更多有见地的例子，将Pascal和MIPS的例子换成C和Intel 86的，以及关于运行时系统的新的一章。这一章中更加深入讨论了关于现代语言的设计和实现问题。

——Eileen Head，Binghamton大学

这个新版本为这一动态领域带来了黄金般的标准，同时在一本教科书的三个关键方面保持了绝好的平衡，这三个方面是：广度、深度和清晰。

——Christopher Vickery，CUNY的Queens学院

《程序设计语言——实践之路》一书全面覆盖了程序设计语言的理论和实现。Michael Scott使用关于最流行的和最有影响力的程序设计语言的几百个例子，良好地解释了各种概念及其实际的意义。在第3版新增的关于运行时系统的一整章中，包含了虚拟机、即时编译和符号调试等新主题。

——William Calhoun，Bloomsburg大学

关于作者

Michael L. Scott是Rochester大学计算机科学系的教授和原系主任。1985年他由麦迪逊的Wisconsin大学[1]获得计算机科学博士学位。他的研究兴趣在于程序设计语言、操作系统以及高级计算机体系结构的交叉领域，重点关注并行和分布式计算。他是Lynx分布式程序设计语言的设计者，与他人合作设计了Charlotte和Psyche并行操作系统、Bridge并行文件系统、Cashmere和InterWeave共享存储系统，以及事务性存储实现的RSTM套件。他与John Mellor-Crummey合作设计的MCS互斥锁被用在许多商业性和学术性的系统中。与Maged Michael、Bill Scherer和Doug Lea设计的一些其他算法出现在java.util.concurrent标准库中。2006年他和Mellor-Crummey博士共同分享了ACM SIGACT/SIGOPS Edsger W. Dijkstra的分布式计算奖。

Scott博士是美国计算机协会（Association for Computing Machinery，ACM）的会员，IEEE的高级会员，Concerned Scientists联合会和社会责任计算机专业工作者协会的会员，曾为各种程序委员会和资助评审委员会服务，作为负责人或协作研究者从事过来自NSF、ONR、DARPA、NASA、国防部、福特基金会、数字设备公司（现为HP）、Sun、IBM、Intel和微软资助的许多项目。作为超过100篇论文出版物的作者，他曾担任过2003年ACM操作系统原理研讨会以及2008年ACM SIGPLAN并行程序设计的原理和实践研讨会的主席。2001年他获得了Rochester大学的“Robert和Pamela Georgen本科生教学杰出贡献奖”。

[1] 即美国威斯康星州州立大学麦迪逊分校。

献给我的父母，

Dorothy D.Scott和Peter Lee Scott，

他们是子女的楷模，

在人文价值方面是我们杰出的表率。

序

21世纪的日常生活中计算机已经无处不在，这说明程序设计语言处于计算机科学教育的中心是合理的。程序设计语言将计算机科学的理论基础（各种问题解决算法的来源）和现代计算机体系结构（相应的程序在其上生成解决方案）联系在一起。在后Internet时代，计算技术的发展日新月异，计算的教科书的结构必须组织起关于某个主题的信息，而不只是讨论这个主题。在这本书中，Michael Scott为读者广泛并全面地呈现了程序设计语言及其实现的关键概念，其方式非常适合计算机科学这一主题。

Scott这本书的最大的优点在于，它将对语言概念的描述与如何实现这些概念的具体说明从整体上结合在一起。这些讨论非常深入，第3版中更新了反映最新研究和实践的内容，除基本的信息之外，还为对特定主题感兴趣的读者提供了补充材料。即使有选择地去掉一些内容，教授本书的讲师还是可以整理出一份本书各主题的一个内在统一的子集。此外，Scott使用了来自真实语言的大量实例来说明关键问题。对于感兴趣或有目标的读者，可以在本书的配套光盘中找到其他一些深入的和高级的讨论与练习，从而使兴趣和能力不同的学生可以根据自己在程序设计语言及编译方面的基础情况进行进一步的探索。

在过去的几年中，我使用Scott的教科书讲授一门时长一学期的比较程序设计语言课程。我向学生们强调，我给他们的目标是学习如何来学习程序设计语言，而不是掌握关于任何一门程序设计语言的细节。这门课的目的是教会学生一种在其职业生涯中学习新语言（在计算机科学领域中必然会遇到这种情况）的有组织的架构。到目前为止，我特别喜欢Scott关于程序设计语言范式（即函数式、逻辑式、面向对象语言和脚本语言）的内容，我的课堂资料也是按这种方式组织的。不过，我还在其中包含了一些基础性的知识，如存储组织、名字和位置、作用域、类型以及废料收集，这些都受益于将语言概念连接到其实现细节的呈现方式。Scott的讲解直奔主题并且非常直观，具有清晰的展示和良好的实例。讨论往往与前面展现的资料相互独立，从而使得从课程大纲中选择主题更加容易。此外，网上还有一些补充的教学材料。

对于我来说，本书的这一版本使我最感兴趣的部分，是关于运行时环境和虚拟机（VM）的全新的第15章，以及对关于并发的第12章的大量更新。鉴于当前对虚拟化的关注，在书中包含关于VM（例如Java的JVM和CLI）的一章，有助于学生们理解这个重要

的主题，同时说明了现代语言如何获得跨许多平台的可移植性。对动态编译和二元翻译的讨论，提供了对书中前面展示的传统编译模型的一个对照。Scott在此包含的这种较新的编译技术的确很重要，有助于学生们可以更好地理解，为了支持书中所描述的较新的动态语言功能我们需要些什么。此外，对符号调试和性能分析的讨论，展示了软件开发周期中遍布的各种程序设计语言和编译器技术。

类似地，Scott对第12章增加了一些讨论，其内容是近期研究所关注的较新的主题（例如存储一致模型、软件事务性存储）。将并发作为一种程序设计范式的讨论，可以放在程序设计语言的课程中，而不仅是放在操作系统的课程中。在这种课程中，可以很容易地比较和对比语言设计的各种选择，以及它们所隐含的实现方面的考虑。这种对语言设计、编译、操作系统和体系结构之间界限的混合，反映了当前实践中的软件开发。这种事实也在Scott的这本第3版中得到了体现。

除了这些重要的变化之外，这一版本还更新了大量的实例（例如用X86上的C代码代替了Pascal），并且用C#、Java 5、Python和Eiffel等现代语言增强了讨论。用多种程序设计语言来展现例子，可以帮助学生理解那些重要的基础通用概念，而不是它们语法上的区别。

总之，Michael Scott的这本书是一本关于程序设计语言及其实现的优秀教科书。这本第3版为学生提供了很好的参考，可作为大学课程的补充资料。这本教科书可以通过不同的方式组织起几种不同“风格”的课程，每一种都覆盖大多数但不是全部的章节。本书的内容清晰且全面，同时讨论了彼此支持的语言设计和实现问题。

恭喜Michael为这本优秀的教科书创作了出色的第3版！

Barbara G. Ryder
美国弗吉尼亚理工大学计算机科学系教授

前　言

计算机程序设计的课程，让普通学生第一次接触到计算机科学领域。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，用来发送电子邮件、玩计算机游戏、浏览网页、做文字处理、参加社交网络，以及从事大量其他事情，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。它的目标很简单，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：如果我们真的能很好地解释一件事情究竟是怎样的，学生总是会希望理解更多东西，并乐于去接触更多材料的。

在常规的有关“系统”的教学计划中，数据结构（或者再加上计算机组织结构）之后的内容被分别归入不同领域的一批课程中，如程序设计语言、编译器构造、计算机体系结构、操作系统、网络、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，就是这一课程表在不断增长，但是本科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进展，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25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、编译器以及语言的运行时系统之间的分界线变得很模糊了。如今许多程序被常规地嵌在网页、电子报表和用户界面中。随着多核处理器的出现，原来只有系统程序员才会用到的并发技术，现在已经开始影响日常的程序设计了。

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中的趋势。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，再提供一些更专业的后续课程。这一趋势在很大程度上是ACM/IEEE-CS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，第11.1节，有修改]。

《程序设计语言——实践之路》的第1版和第2版很幸运地跟上了这种趋势。第3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。其中并没有列举许多不同语言的细节，而是重点关注学生可能遇到的有关所有语言之基础的那些概念，通过各种各样的具体例子来阐释这些概念，并努力探索解释不同语言的设计之所以采纳不同方式背后的那些利弊权衡。类似地，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完整参与的工作），它将重点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3版中的变化

《程序设计语言——实践之路》的第3版与第2版相比，有以下不同。

1. 关于虚拟机和运行时程序管理的一章新内容。
2. 对介绍并发的一章进行了大量修订。
3. 在许多地方反映了该领域中近期出现的变化。
4. 根据来自讲授该课程教师的反馈，以及对广为人知的主题进行重新思考，对本书进行了一些改进。

第1项也许是最明显的变化。它反映了托管代码和脚本语言日益广泛的使用。第15章首先从整体上介绍了虚拟机，然后详细讨论了两个使用最广泛的实例：JVM和CLI。这章还介绍了动态编译、二进制翻译、自反、调试器、分析器，以及现代语言系统中出现的越来越复杂的运行系统机制的其他方面。

第2项也反映了这一领域不断发展的事实。随着多核处理器的传播，并发语言对于主流计算机来说已经越来越重要了，而且这一领域还在不断地变化发展。对第12章（并发）的改动包括关于非阻塞同步、存储一致性模型和软件事务性存储的各小节，以及关于OpenMP、Erlang、Java 5和用于.NET的Parallel FX的更多内容。

其他新内容（第3项）遍布在本书中的各个地方。©第5.4.4节从体系结构的角度介绍了多核的创新。第8.7节讨论了串行和并发语言中的事件处理。第14.2节中，关于gcc的讨

论不仅包括RTL，也包括更新的GENERIC和Gimple中间形式。全书的引用内容都做了更新，以反映最新的发展情况，如Java 6、C++'0X、C# 3.0、F#、Fortran 2003、Perl 6和Scheme R6RS。

最后，第4项包含了对本书中几乎所有小节的改进。其中，更新最多的主题包括贯穿第1章的例子（从Pascal/MIPS改为C/x86），自举（第1.4节），扫描（第2.2节），表驱动语法分析（第2.3.2节和第2.3.3节），闭包（第3.6.2、3.6.3、8.3.1、◎第8.4.4、8.7.2和9.2.3节），宏（第3.7节），求值顺序和严格性（第6.6.2节和第10.4节），小数类型（第7.1.4节），数组形状和分配（第7.4.2节），参数传递（第8.3节），内部（嵌套）类（第9.2.3节），单体（第10.4.2节）和第11章的Prolog实例（现在遵循ISO）。

为了容纳新的内容，关于某些主题的讨论被压缩了。这样的情况包括模块（第3章和第9章），循环控制（第6章），补偿类型（第7章），Smalltalk类继承关系（第9章），元循环解释（第10章），互联网络（第12章）和线程创建语法（第12章）。其他一些内容被放到了本书的配套光盘上。这些内容包括第5章（目标机体系结构），联合（◎第7.3.4节），悬空引用（◎第7.7.2节），消息传递（◎第12.5节）和XSLT（◎第13.3.5节）。在全书中，原来用现在已不流行的语言编写的例子，都在恰当的情况下替换为更新的等价语言。

整体上看，本书的印刷本只比原来多了30页，不过配套光盘上多了将近100页新内容。此外还新增了14个“设计和实现”旁白，70多个新例子，数量相当的新“检查你的理解”问题，以及超过60个新的章节后练习和探索。在创建一个一致且全面的索引时付出了相当的努力。正如前面两个版本一样，Morgan Kaufmann继续承诺以合理的成本来提供最佳的图书：本书的第3版比同类书籍更便宜，但是内容更多，也更全面。

配套光盘

为了减小本书的物理体积，为新材料让出位置，使学生可以在浏览本书时重点关注最基本的材料，大约350页更高级的或者更外围的材料被放入随书配套的光盘中。光盘中的各节在书的正文中都用一段关于其主题的简介，以及一个总结省略部分的“深度探索”段落来表示。

需要注意的是，将一些材料安排在光盘上并不构成对这些材料技术重要性的判断，而只是反映了一个事实：所有值得包含的材料已经超过了一卷书或者一门课程的容量。由于侧重点和教学进度各不相同，大多数教师可能会安排学生自己阅读配套光盘上的材料，并且会略过印刷出的教科书中的某些特定章节。我的意图只是将那些大部分课程可

能涵盖的材料放在印刷形式的教科书中。

光盘上还包含了正文中所有主要代码片段的可编译副本（包括二十几种语言），以及对在线资源的指引。

关于设计和实现的旁白

与前两版一样，本书第3版重点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3版用超过135个“设计和实现”旁白，突出了这方面的联系和相互影响。有关这些旁白的更多细节将在第9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

第3版中的实例紧密地编织在讨论中。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，我们给每个实例都显示一个页边注，标明其编号和标题。全文和配套光盘里大约有1000个这样的实例，附录C是有关它们的详细目录。

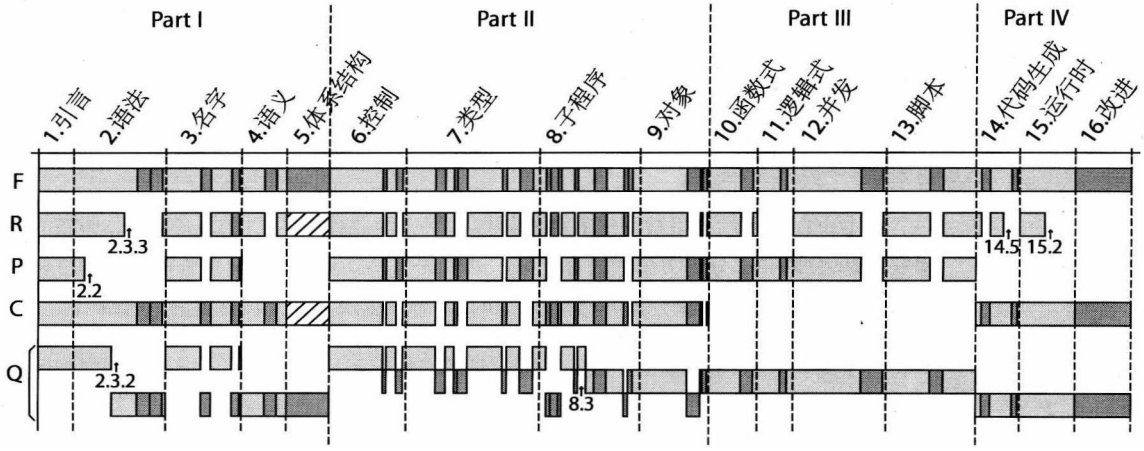
练习的安排

复习题放在本书中各主要小节的后面，大约10页就出现一处。它们直接基于前面的正文内容，并且有着较短和直截了当的答案。

更详细的问题放在各章的最后。它们分为练习和探索两类。前者一般比各节后面的复习题更有挑战性，它们适合作为课下作业或简单的课题。后者中的问题更加开放，需要做一些基于网络或图书馆的研究工作，需要明显更多一些的时间，或者是形成一种客观的认识。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textbooks.elsevier.com/web/9780123745149。

如何使用本书

《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（程序设计语言的设计），也可以被用于CS 340（编译器构造）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）中的主要内容。图0.1给出的是使用本教程的几条可能路径。



F: 一年/自学计划

R: 罗彻斯特大学一学期计划

P: 传统程序设计语言计划，在所示章节中还可以减少一些强调实现的内容

C: 编译器课程计划，在所示章节也可以减少有关设计的材料

Q: 1+2学季计划。一个学季概览，另外两个独立的可选学季，一个强调语言，一个强调编译

■ 补充材料（光盘）

▨ 略过，根据学生需要浏览

图0.1 课程路线图。较暗的区块表示配套光盘中的“深度探索”补充内容。节序号表示没有与补充内容对应的断开的部分。

作为自学或一学年的课程（图0.1中的轨迹F），我建议从头到尾通读本书，当遇到每个“深度探索”部分时再转到相应的配套光盘上的内容。在罗彻斯特大学作为一学期的课程时（轨迹R. 本书最初就是为它开发的），我们也覆盖了本书的大部分内容，但不包括大部分光盘上的章节，自下而上的语法分析（第2.3.3节）和第14章（构造可运行程序）的第二部分内容，以及第15章（运行时程序管理）。

一些章（第2、4、5、14、15和16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的章之间的顺序可以在一定限度内调整。许多学生可能已经熟悉了第5章的一些材料，多半是来自一门有关计算机组织结构的课程，因此我们将第5章放在了配套光盘上。有些学生可能也熟悉第2章的一些材料，可能是来自一门有关自动机理论的课程。在这种情况下，该章的大部分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（图0.1中的轨迹P），那么可以排除有关扫描器和语法分析器的内容，加上第4章的所有内容，还可以减少对其他章节中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（第7.2.4节），多重继承（第9.5节），Smalltalk（第9.6.1节），lambda演算（第10.6节）和谓词演算（第11.3节）。

本书也在一些学校里被用于有关编译器的引论性课程（图0.1的轨迹C）。典型的课程安排是去掉第3部分的大多数内容（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到16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，一种常见的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0.1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和第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到第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到第5章和第14到第15章，以及第3、第8和第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。具体是哪种语言都没有关系。书中例子取自各种不同的语言，但总是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包含了超过50种不同语言的简单介绍。这里的算法都是采用自解释的非形式的伪代码。真正的程序设计语言代码使用的字体是“typewriter”字体，伪代码用的字体是“sans-serif”字体。

辅助材料

除了作为正文章节的补充外，配套光盘中还包括其他一些资源：

- 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；
- 对文本内容和配套光盘内容的搜索引擎。