大学计算机教育丛书（影印版）

THIRD EDITION

PROGRAMMING LANGUAGES

DESIGN AND IMPLEMENTATION

程序设计语言设计与实现

第3版

TERRENCE W. PRATT
MARVIN V. ZELKOWITZ

清华大学出版社 · PRENTICE HALL
http://www.tup.tsinghua.edu.cn

PROGRAMMING LANGUAGES

Design and Implementation

THIRD EDITION

程序设计语言

设计与实现

第3版

Terrence W. Pratt

Center of Excellence in Space Data and Information Sciences
NASA Goddard Space Flight Center, Greenbelt, MD

Marvin V. Zelkowitz

Department of Computer Science and
Institute for Advanced Computer Studies
University of Maryland, College Park, MD

清华大学出版社
Prentice-Hall International, Inc.

(京)新登字158号

Programming languages: design and implementation 3rd ed./Terrence W. Pratt and Marvin V. Zelkowitz.

图书在版编目(CIP)数据

程序设计语言:设计与实现:第3版:英文/普拉特(Pratt, T.W.),泽尔柯维支(Zelkowitz, M.V.)著. -影印版. -北京:清华大学出版社,1998.5
(大学计算机教育丛书)
ISBN 7-302-02833-8

Ⅰ.编… Ⅱ.①普… ②泽… Ⅲ.程序语言,英文 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(98)第01690号

出版者:清华大学出版社(北京清华大学校内,邮编 100084)
因特网地址:www.tup.tsinghua.edu.cn
印刷者:清华大学印刷厂
发行者:新华书店总店北京科技发行所
开　本:850×1168 1/32　**印张**:21.125
版　次:1998年5月第1版　1998年5月第1次印刷
书　号:ISBN 7-302-02833-8/TP·1490
印　数:0001～5000
定　价:32.00元

出版前言

我们的大学生、研究生毕业后，面临的将是一个国际化的信息时代。他们将需要随时查阅大量的外文资料；会有更多的机会参加国际性学术交流活动；接待外国学者；走上国际会议的讲坛。作为科技工作者，他们不仅应有与国外同行进行口头和书面交流的能力，更为重要的是，他们必须具备极强的查阅外文资料获取信息的能力。有鉴于此，在国家教委所颁布的“大学英语教学大纲”中有一条规定：专业阅读应作为必修课程开设。同时，在大纲中还规定了这门课程的学时和教学要求。有些高校除开设“专业阅读”课之外，还在某些专业课拟进行英语授课。但教、学双方都苦于没有一定数量的合适的英文原版教材作为教学参考书。为满足这方面的需要，我们挑选了 7 本计算机科学方面最新版本的教材，进行影印出版。首批影印出版的 6 本书受到广大读者的热情欢迎，我们深受鼓舞，今后还将陆续推出新书。希望读者继续给予大力支持。Prentice Hall 公司和清华大学出版社这次合作将国际先进水平的教材引入我国高等学校，为师生们提供了教学用书，相信会对高校教材改革产生积极的影响。

清华大学出版社

Prentice Hall 公司

1997.11

Preface

This third edition of *Programming Languages: Design and Implementation* continues the tradition developed in the first two editions to describe programming language design by means of the underlying software and hardware architecture that is required for execution of programs written in those languages. This provides the programmer with the ability to develop software that is both correct and efficient in execution. In this new edition, we continue this approach, as well as improve upon the presentation of the underlying theory and formal models that form the basis for the decisions made in creating those languages.

Programming language design is still a very active pursuit in the computer science community as languages are "born," "age," and eventually "die." This third edition represents the vital languages of the mid-1990s. Chapters on COBOL, PL/I, SNOBOL4, and APL have been dropped. Discussions on C, C++, ML, Prolog, and Smalltalk have been added to reflect the evolution of programming language design and the emergence of new paradigms within the community. Pascal is starting to age, and Ada and FORTRAN have been renewed with new standards, Ada 95 and FORTRAN 90, respectively. It is interesting to speculate as to whether any of these languages will be in future editions of books such as this one. For both of us, the deletion of SNOBOL4 was a considerable loss. It is one of the most interesting and powerful languages ever developed, although it still lives on as "shareware" in PCs.

At the University of Maryland, a course has been taught for the past 20 years that conforms to the structure of this book. For our junior-level course, we assume the student already knows Pascal and C from earlier courses. We then emphasize ML, Prolog, C++, and LISP, as well as include further discussions of the implementation aspects of C and Pascal. The study of C++ furthers the students' knowledge of procedural languages with the addition of object oriented classes, and the inclusion of LISP, Prolog, and ML provide for discussions of different programming paradigms. Replacement of one or two of these by FORTRAN, Ada, or Smalltalk would also be appropriate.

It is assumed that the reader is familiar with at least one procedural language, generally C, FORTRAN, or Pascal. For those institutions using this book at a lower level, or for others wishing to review prerequisite material to provide a framework for

discussing programming language design issues, Chapters 1 and 2 provide a review of material needed to understand later chapters. Chapter 1 is a general introduction to programming languages, while Chapter 2 is a brief overview of requirements for programming languages.

The theme of this book is language design and implementation issues. Part I forms the core of an undergraduate course in programming languages. Chapters 3 through 8 are the basis for this course by describing the underlying grammatical model for programming languages and their compilers (Chapter 3), elementary data types (Chapter 4), encapsulation (Chapter 5), statements (Chapter 6), procedure invocation (Chapter 7), and inheritance (Chapter 8), which are the central concerns in language design. Examples of these features are described in a variety of languages and typical implementation strategies are discussed.

The topics in this book cover the 12 knowledge units recommended by the 1991 ACM/IEEE Computer Society Joint Curriculum Task Force for the programming languages subject area [TUCKER et al. 1991]. For institutions using this book at a higher level or those wishing to address more advanced topics, Chapter 9 continues the discussion of parsing that is first introduced in Chapter 3 and brings in the concept of programming language semantics with discussions of program verification, denotational semantics, and the lambda calculus with an introduction to undecidability and NP completeness. This provides the reader with an overview of more advanced courses in the programming language, software engineering, and computational theory areas of computer science. For this material, prior experience with the predicate calculus and mathematical logic would help. In addition, Chapter 9 addresses current issues in parallel programming, provides an introduction to current research in hardware and software, and suggests what are likely to be the programming language design issues in the future.

While compiler writing was at one time a central course in the computer science curriculum, there is increasing belief that not every computer science student needs to be able to develop a compiler; such technology should be left to the compiler specialist, and the "hole" in the schedule produced by deleting such a course might be better utilized with courses such as software engineering, database engineering, or other practical use of computer science technology. However, we believe that aspects of compiler design should be part of the background for all good programmers. Therefore, a focus of this book is how various language structures are compiled, and Chapter 3 provides a fairly complete summary of parsing issues.

The nine chapters of Part I emphasize programming language examples in FORTRAN, Ada, C, Pascal, ML, LISP, Prolog, C++, and Smalltalk. Additional examples are given in PL/I, SNOBOL4, APL, BASIC, and COBOL, as the need arises. The sections of Part II, however, are organized around individual languages. Each section describes a different language and shows how that language provides the features described in the first nine chapters of Part I. The goal is to present each language as a consistent implementation of the software architecture given in the first half of the book. While certainly not a reference manual for each language, each section should provide enough information for the student to solve interesting

class problems in each of those languages without the need to purchase separate language reference manuals. (However, having a few of those around for your local implementation is certainly a big help.)

While discussing all of the languages briefly during the semester is appropriate, we do not suggest that the programming parts of this course consist of problems in each of these languages. We think that would be too superficial in one course. Nine programs in nine different languages would be quite a chore and provide the student with little in-depth knowledge of any of these languages. We assume that each instructor will choose three or four of the Part II languages and emphasize those.

All examples in this book, except for the most trivial, were tested on an appropriate translator; however, as we clearly point out in Section 1.3.3, correct execution on our local system is no guarantee that the translator is processing programs according to the language standard. We are sure that Mr. Murphy is at work here, and some of the "trivial" examples may have errors. If so, we apologize for any problems that may cause.

To summarize, our goal in producing this third edition was to:

- Provide an overview of the key paradigms used in developing modern programming languages;
- Highlight several languages, which provide those features, in sufficient detail to permit programs to be written in each language demonstrating those features;
- Explore the implementation of each language in sufficient detail to provide the programmer an understanding of the relationship between a source program and its execution behavior;
- Provide sufficient formal theory to show where programming language design fits within the general computer science research agenda; and
- Provide a sufficient set of problems and alternative references to allow students the opportunity to extend their knowledge of this important topic.

We gratefully acknowledge the valuable comments received from Henry Bauer, Hikyoo Koh, John Mauney, and Andrew Oldroyd on earlier drafts of this manuscript and from the 118 students of CMSC 330 at the University of Maryland during the Spring, 1995 semester who provided valuable feedback on improving the presentation contained in this book.

Perhaps 70% of the text has been rewritten between edition 2 and edition 3. We believe the new edition is a considerable improvement over the previous version of this book. We hope that you agree.

Terry Pratt
Greenbelt, Maryland

Marv Zelkowitz
College Park, Maryland

Contents

清华大学出版社
Prentice Hall 公司合作影印计算机类图书目录

1997 年 1 月—1998 年 7 月

1. *Computer Networks 3rd ed.* 1996/**Andrew S. Tanenbaum**(计算机网络 第 3 版 832 页)
2. *Distributed Operating Systems* 1995/**Andrew S. Tanenbaum**(分布式操作系统 628 页)
3. *The C Programming Language 2nd ed.* 1988/**Brian W. Kernighan, Dennis M. Ritchie**(C 程序设计语言 第 2 版 284 页)
4. *The C Answer Book*: ***Solutions to the Exercises in The C programming Languange, second edition by*** **Brian W. Kernighan & Dennis M. Ritchie** *2nd ed.* 1989/**Clovis L. Tondo, Scott E. Gimpel**(C 程序设计语言第 2 版习题解答 第 2 版 216 页)
5. *Multimedia*: ***Computing, Communications & Applications*** 1995/**Ralf Steinmetz, Klara Nahrstedt**(多媒体技术：计算，通信及应用 876 页)
6. *Data Stracture with C + +* 1996/**William Ford, William Topp**(数据结构 C + + 语言描述 916 页)
7. *Computer Organization And Architecture*: ***Designing for Performance*** *4th ed.* 1996/**William Stallings**(计算机组织与结构：性能设计 第 4 版 696 页)
8. *Use Case Maps for Object-Oriented Systems* 1996/**R. J. A. Buhr, R. S. Casselman**(用于面向对象系统开发的使用实例图 326 页)
9. *Operating Systems*: ***Design and Implementation 2nd ed.*** 1997/**Andrew S. Tanenbaum, Albert S. Woodhull**(操作系统：设计与实现 第 2 版 960 页 附光盘)
10. *Data And Computer Communications 5th ed.* 1997/**William Stallings**(数据通信与计算机通信 第 5 版 820 页)
11. *Digital Logic Circuit Analysis & Design* 1995/**Victor P. Nelson, H. Troy Nagle, Bill D. Carroll, J. David Irwin**(数字逻辑电路分析与设计 868 页)

12. *Discrete Mathematical Structures* 3*rd ed*. 1996/**Bernard Kolman, Robert C. Busby, Sharon Ross**(离散数学结构 第3版 544页)
13. *Computer graphics C Version* 2*nd ed*. 1997/**Donald Hearn, M. Pauline Baker**(计算机图形学 C语言版 第2版 864页 附彩插)
14. *Computer Networks and Internets* 1997/**Douglas E. Comer**(计算机网络与因特网 530页 附光盘)
15. *A First Course in Database Systems* 1997/**Jeffrey D. Ullman, Jennifer Widom**(数据库系统基础教程 484页)
16. *Digital Image Processing* 1996/**Kenneth R. Castleman**(数字图象处理 686页)
17. *Business Data Communications* 3*rd ed*. 1998/**William Stallings, Richard Van Slyke**(事务数据通信 第3版 578页)
18. *Software Architecture*: ***Perspectives on an Emerging Discipline*** 1996/**Mary Shaw, David Garlan**(软件体系结构 264页)
19. *Digital Video Processing* 1995/**A. Murat Tekalp**(数字视频处理 548页)
20. *IBM PC Assembly Language and Programming* 4*th ed*. 1998/**PETER ABEL**(*IBM PC* 汇编语言与程序设计 第4版 622页)
21. *Operating Systems*: ***Internals and Design Principles*** 3*rd ed*. 1998/**William Stallings**(操作系统:精髓与设计原理 第3版 800页)
22. *Programming Languages*: ***Design and Implementation*** 3*rd ed*. 1996/**Terrence W. Pratt, Marvin V. Zelkowitz**(程序设计语言:设计与实现 第3版 570页)