

普通高校本科计算机专业

特色

教材精选

编译设计与开发技术

斯传根 编著

<http://www.tup.com.cn>

14
3

清华大学出版社



TP314
5793

普通高等院校计算机专业**特色**教材精选

编译设计与开发技术

斯传根 编著



A1105830



清华大学出版社
北 京

H588/02

内 容 简 介

本书在讲述与编译相关的文法和形式语言基本理论的基础上,以 PL/0 语言为例,系统地介绍了编译程序从语法定义、词法分析、语法分析、出错处理、代码生成到解释执行的全过程。并详细讲解了作者研究开发的 SI-NS 图表示方法和从程序到 SI-NS 图的自动转换技术。本书有很强的实用性,学生在学完本书后可具备独立设计和完整地开发一个编译程序的能力。本书每章后配有习题,可供学生巩固练习。

本书适合作为普通高等院校计算机专业学生的教材,也可供其他自学人员和编程技术人员参考。

版权所有,翻印必究。

本书封面贴有清华大学出版社激光防伪标签,无标签者不得销售。

图书在版编目(CIP)数据

编译设计与开发技术/斯传根编著. —北京:清华大学出版社,2003

(普通高校本科计算机专业特色教材精选)

ISBN 7-302-07497-6

I. 编… II. 斯… III. 编译程序—程序设计—高等学校—教材 IV. TP314

中国版本图书馆 CIP 数据核字(2003)第 108858 号

出 版 者: 清华大学出版社

<http://www.tup.com.cn>

社 总 机: 010-62770175

地 址: 北京清华大学学研大厦

邮 编: 100084

客户服务: 010-62776969

责任编辑: 王听讲

印 刷 者: 北京国马印刷厂

装 订 者: 三河市李旗庄少明装订厂

发 行 者: 新华书店总店北京发行所

开 本: 185×260 印张: 16.25 字数: 364 千字

版 次: 2003 年 12 月第 1 版 2003 年 12 月第 1 次印刷

书 号: ISBN 7-302-07497-6/TP·5526

印 数: 1~5000

定 价: 23.00 元

编审委员会

主任：蒋宗礼

副主任：李仲麟 何炎祥

委员：（排名不分先后）

王向东 宁 洪 朱庆生 吴功宜 吴 跃

张 虹 张 钢 张为群 余雪丽 陈志国

武 波 孟祥旭 孟小峰 胡金初 姚放吾

原福永 黄刘生 廖明宏 薛永生

秘书长：王听讲

出版说明

INTRODUCTION

在我国高等教育逐步实现普及化后,越来越多的高等学校将会面向国民经济发展的第一线,为行业、企业培养各级各类高级应用型专门人才。为此,教育部已经启动了“高等学校教学质量和教学改革工程”,强调要以信息技术为手段,深化教学改革和人才培养模式改革。如何根据社会的实际需要,根据各行各业的具体人才需求,培养具有显著特色的人才,是我们共同面临的重大问题。具体地,培养具有一定专业特色的和特定能力强的计算机专业应用型人才则是计算机教育要解决的问题。

为了适应 21 世纪人才培养的需要,培养具有特色的计算机人才,急需一批适合各种人才培养特点的计算机专业教材。目前,一些高校在计算机专业教学和教材改革方面已经做了大量工作,许多教师在计算机专业教学和科研方面已经积累了许多宝贵经验。将他们的教研成果转化为教材,向全国其他学校推广,对于深化我国高等学校的教学改革是一件十分有意义的事。

清华大学出版社在经过大量调查研究的基础上,决定编写出版一套“普通高校本科计算机专业特色教材精选”。本套教材是针对当前高等教育改革的新形势,以社会对人才的需求为导向,主要以培养应用型计算机人才为目标,立足课程改革和教材创新,广泛吸纳全国各地的高等院校计算机优秀教师参与编写,从中精选出版确实反映计算机专业教学方向的特色教材,供普通高等院校计算机专业学生使用。

本套教材具有以下特点:

1. 编写目的明确

本套教材是在深入研究各地各校办学特色的基础上,面向普通高校的计算机专业学生编写的。学生通过本套教材,主要学习计算机科学与技术专业的基本理论和基本知识,接受利用计算机解决实际问题的基本训练,培养研究和开发计算机系统,特别是应用系统的基本能力。

2. 理论知识与实践训练相结合

根据计算学科的三个学科形态及其关系,本套教材力求突出学科的理论与实践紧密结合的特征,结合实例讲解理论,使理论来源于实践,又进一步指导实践,学生通过实践深化对理论的理解,更重要的是使学生学会理论方法的实际运用。在编写教材时突出实用性,并做到通俗易懂,易教易学,使学生不仅知其然,知其所以然,还要会其如何然。

3. 注意培养学生的动手能力

每种教材都增加了能力训练部分的内容,学生通过学习和练习,能比较熟练地应用计算机知识解决实际问题。既注重培养学生分析问题的能力,也注重培养学生解决问题的能力,以适应新经济时代对人才的需要,满足就业要求。

4. 注重教材的立体化配套

本套教材的大多数教材都将陆续配套教师用课件、习题及其解答提示,学生上机实验指导等辅助教学资源,有些教材还提供能用于网上下载的文件,以方便教学。

由于各地区各学校的培养目标、教学要求和办学特色均有所不同,所以对特色教学的理解也不尽一致,我们恳切希望大家在使用本书的过程中,及时地给我们提出批评和改进意见,以便我们做好教材的修订改版工作,使其日趋完善。

我们相信经过大家的共同努力,这套教材一定能成为特色鲜明、质量上乘的优秀教材,同时,我们也希望通过本套教材的编写出版,为“高等学校教学质量和教学改革工程”作出贡献。

清华大学出版社

前言

PREFACE

本书以 PASCAL 语言创始人,世界著名计算机科学家 N. Wirth 教授的德文版《编译设计入门》为蓝本,总结了作者多年“编译”课程教学实践和软件开发的经验,在原有讲义和讲稿基础上重新编写而成。教材起点高、内容新而精,并注重应用实践和学生实际软件开发能力的培养。

本书可作为计算机专业本科生的“编译原理”课程教材或教学参考书。同时,如果省略书中内容相对独立的第 12 章、附录 C 和附录 D 后,并不影响整体,也可以用作成人高校、大专班学生以及课时较少学校的“编译原理”课程教材。

由于设计和开发编译程序的基本原理和技术也广泛应用于其他软件的设计和开发,因此本书也是从事系统软件和工具软件开发研究者的参考书;还可以作为初步学习过高级程序设计语言 PASCAL 或 C 语言后,继续深化学习“程序设计”者的参考书。

本书前 5 章讲述与编译相关的文法和形式语言基本理论。第 6 章至第 11 章以语言 PL/0 为例,系统介绍了一个编译程序从语法定义、词法分析、语法分析、出错处理、代码生成到解释执行的全过程。它不仅可以使了解什么是编译,更重要的是在学完了约 60 学时的课程内容后,学生完全有能力懂得怎样从语言的定义出发,系统地去开发一个语言的编译程序。

第 12 章通过“T 形图”的描述,可使学生对编译和解释程序等概念有更深刻的认识,它还直观、清晰地表达了编译程序特有的开发和移植技术。

第 13 章包含的一个精心编写的课程设计,既可以在学完前 12 章内容后集中时间进行,也可以从第 6 章开始结合每一章的练习分散进行。它从给定语言的语法定义开始,先要求作语法规则检查,然后要求系统地写出词法分析、语法分析、出错处理、代码生成和解释程序,可以分两个阶段实施检查。最终要求学生用所给的语言编一个程序(如输出

1~100 之间的所有素数), 用自己开发的编译程序对它编译, 在编译过程中要求能连续指出语法错误不中断, 能生成代码程序, 能解释执行代码程序, 最后输出正确结果。通过课程设计不仅可以系统地巩固学生所学知识; 也可以考核学生是否真正具备了独立设计和完整地开发一个编译程序的能力。

“程序图形化表达”是国际上都在致力研究, 旨在提高程序可读、可理解性的一项极具潜力和应用前景的技术, 得到了发达国家政府和有关部门的大力资助。作者研究发展的 SI-NS 图表示方法和从程序到 SI-NS 图自动转换技术, 要优于日本的 PAD、美国的 CSD 以及国内正在广泛使用的 NS 图技术, 并且在长期的软件开发和教学实践中得到了验证。它使教材的全部实例程序都以文本程序和 SI-NS 程序框图两种形式出现, 使学生不仅更容易读懂、理解编译程序的算法和设计思想, 甚至可以在二维、结构清晰的 SI-NS 程序框图上直接执行和验证程序逻辑。这在一维的、整体结构差的文本程序里是很难做到的。附录 A 提供了 PL/0 语言编译程序 program6 的 SI-NS 程序框图。对 SI-NS 图不了解的读者可以阅读附录 B 的有关文章。

教材内容和课程设计贯穿软件工程提倡的“自顶向下”、“逐步扩充”、“逐步求精”的结构化程序设计思想。学生通过教学实践掌握这些方法和技术后, 为今后走上社会去承担有一定规模和复杂度的实际课题奠定了一定的软件开发基础。

对于部分没有学过“离散数学”或对有关概念已经淡忘了的读者, 建议阅读附录 E。这里解释了如“符号”、“符号串”、“集合”、“集合的表示”、“集合的闭包”等与形式语言有关的一些重要概念。

教材涉及的全部实例程序经过 Turbo PASCAL 7.0 的编译都可以上机执行。另外清华大学出版社可以向读者提供电子版的所有实例程序。

感谢浙江大学城市学院冯树椿和曾抗生两位教授的热心帮助, 他们审阅了全部书稿并提出了非常宝贵的意见。还要感谢清华大学出版社和本套教材的编审委员会, 使得本书得以顺利出版。

由于作者水平有限, 书中难免存在缺点错误, 敬请广大读者批评指正。

作 者

2003 年 7 月

目 录

CONTENTS

第 1 章 引论	1
1.1 什么是编译程序	1
1.2 编译过程概述	2
1.3 编译程序的结构	4
1.4 为什么要学习编译技术	6
1.5 习题	7
第 2 章 形式语言基本知识	9
2.1 形式语言的特征	9
2.2 一个形式语言例子	11
2.3 形式语言的归纳定义	12
2.4 文法的分类	14
2.4.1 0 型文法	14
2.4.2 1 型文法(上下文有关文法)	14
2.4.3 2 型文法(上下文无关文法)	15
2.4.4 3 型文法(正则文法)	15
2.4.5 文法的类型决定了语言的类型	15
2.5 语法树与文法的二义性	16
2.5.1 什么是文法的二义性	16
2.5.2 修改二义性文法	17
2.5.3 计算机语言文法中的二义性问题	17
2.5.4 EBNF 描述符	18
2.6 习题	20
第 3 章 语句分析	23
3.1 语句分析概述	23
3.2 自顶向下分析	24

3.3	自底向上分析	25
3.4	简单辨认算法	26
3.5	对文法的限制规则 1	27
3.6	对文法的限制规则 2	29
3.7	如何构造等价的 LL(1)文法	30
3.8	习题	30
第 4 章	语法图	33
4.1	从 EBNF 表示式到语法图的转换	33
4.2	语法图转换实例	34
4.3	从语法图判别两条限制规则	35
4.4	习题	36
第 5 章	给定语法的语法分析程序构造	39
5.1	给定语法的语法分析程序的主程序	39
5.2	从语法图到语法分析程序的转换法则	40
5.3	给定语法的语法分析程序构造实例	42
5.3.1	语法分析程序构造实例	42
5.3.2	SI-NS 图形式的语法分析程序构造实例	44
5.3.3	语法分析程序的编译和执行示例	45
5.4	习题	48
第 6 章	程序设计语言 PL/0	51
6.1	PL/0 语言概述	51
6.1.1	PL/0 语言的功能	51
6.1.2	一个 PL/0 程序实例	52
6.2	EBNF 定义的 PL/0 语法	53
6.3	PL/0 语法图	54
6.4	判别是否符合两条限制规则	56
6.5	习题	58
第 7 章	PL/0 语言的词法分析程序构造	59
7.1	词法分析程序的任务	59
7.2	词法分析程序的设计	60
7.2.1	过程 getch	61
7.2.2	处理保留字和标识符	62
7.2.3	处理常数	62
7.2.4	处理组合字符和单个字符	63

7.3 习题	63
第8章 PL/0 语言的语法分析程序构造	67
8.1 由语法图到过程调用相关图	67
8.2 从相关图到程序的总体结构	68
8.3 写出每一个语法分析子程序	69
8.4 语法分析程序的扩充	78
8.4.1 建立符号表与主程序的扩充	79
8.4.2 建立 error 子程序过程和出错信息表	80
8.4.3 标识符的处理与程序的扩充	81
8.5 PL/0 语法分析程序及其执行	83
8.5.1 PL/0 语法分析文本程序	83
8.5.2 PL/0 语法分析程序的 SI-NS 图	89
8.5.3 PL/0 语法分析程序的执行	95
8.6 习题	98
第9章 语法出错处理	101
9.1 语法出错处理概论	101
9.2 常见错误的分析处理	103
9.2.1 修改语法分析程序	103
9.2.2 修改语法定义	105
9.3 不可预料错误的处理	106
9.3.1 采取的策略和相应的措施	106
9.3.2 出错处理分析举例	107
9.4 PL/0 语法出错处理程序及其执行	108
9.4.1 具有出错处理的 PL/0 语法分析程序 program5	108
9.4.2 具有出错处理的 PL/0 语法分析程序 SI-NS 图	115
9.4.3 语法出错处理程序执行实例	123
9.5 习题	126
第10章 目标计算机及其解释程序	129
10.1 目标计算机的组织结构和指令格式	130
10.1.1 目标计算机的组织结构	130
10.1.2 目标计算机的指令和指令格式	132
10.2 数据存储器的动态存储管理	133
10.2.1 数据动态存储分析	133
10.2.2 程序静态级别和动态存储分配	134
10.2.3 数据动态存储采取的技术措施	136

10.3	目标计算机指令系统及其解释	136
10.4	解释程序 interpret 及其执行	138
10.4.1	PL/0 目标程序的解释程序	138
10.4.2	解释程序执行实例	139
10.4.3	在 interpret 的 SI-NS 图上执行解释程序	140
10.5	习题	142
第 11 章	语义分析与代码生成	143
11.1	table 数组和 enter 过程的扩充	143
11.1.1	table 数组的扩充	144
11.1.2	enter 过程的扩充	144
11.2	命令代码生成过程 gen	145
11.3	表达式、项、因子及关系式的翻译	145
11.3.1	后缀式处理的规则	145
11.3.2	后缀式翻译在代码生成中的实现	146
11.3.3	对表达式和关系式翻译举例	149
11.4	语句的翻译	150
11.4.1	语句翻译处理规则	150
11.4.2	语句翻译在程序扩充中的实现	151
11.4.3	语句翻译举例	153
11.5	生成 PL/0 目标代码程序	155
11.5.1	在计算机上执行 program6 生成代码程序	156
11.5.2	在 SI-NS 图上执行 program6 生成代码程序	157
11.5.3	掌握代码生成算法,直接翻译写出代码程序	162
11.6	program6 程序文本	163
11.7	习题	175
第 12 章	T 形图与编译开发移植技术	179
12.1	T 形图	179
12.1.1	程序在计算机上执行的表示	179
12.1.2	编译程序的表示	179
12.1.3	编译和执行过程的表示	180
12.1.4	多级编译过程的表示	180
12.1.5	解释执行过程的表示	180
12.2	编译程序开发的自展技术	181
12.3	编译程序改进的自展技术	182
12.4	编译程序的移植技术	182
12.4.1	在 M1 机器上进行从 M1 到 M2 的移植	183

12.4.2	在 M2 机器上进行从 M1 到 M2 的移植	183
12.4.3	用虚拟机 H 进行从 M1 到 M2 的移植	184
12.5	习题	184
第 13 章	课程设计	187
13.1	PL/x 语法定义	187
13.2	统一符号	188
13.3	课程设计要求	189
13.3.1	阶段一要求	189
13.3.2	阶段一检查	190
13.3.3	阶段二要求	190
13.3.4	阶段二检查	190
附录		191
附录 A	PL/0 编译程序 SI-NS 图	191
附录 B	有关 SI-NS 图	203
B.1	读起来比程序容易理解百倍的 SI-NS 图	203
B.2	程序图形化表示方法	206
B.2.1	现有文本式语言程序的弊端	206
B.2.2	“程序图形化表示”是世界上都在致力研究的 一项技术	206
B.2.3	作者提出了具有优势的 SI-NS 图	207
B.2.4	比较 PAD、CSD、NS 与 SI-NS 图	207
B.2.5	SI-NS 图技术在长期的软件开发和教学实践中 得到了验证	212
附录 C	表格控制的通用语法分析程序构造	212
C.1	语法图表示成表格数据结构	213
C.2	表格控制的语法分析程序	215
C.3	通用语法分析程序的实施步骤	216
C.3.1	用 EBNF 定义 EBNF 语法	216
C.3.2	为 EBNF 语法设计一个语法分析程序	217
C.3.3	将语法分析程序扩充为能产生语法分析表格的 翻译程序	219
C.3.4	将翻译程序与前一节的表格控制语法分析程序 组合在一起	220
C.4	习题	227
附录 D	并行多进程语言的扩充	228
附录 E	集合与符号串	233

E.1 集合	233
E.1.1 集合的基本概念	233
E.1.2 集合间的关系	234
E.1.3 集合的运算	235
E.1.4 幂集	236
E.1.5 笛卡儿乘积	236
E.2 符号串	238
E.3 习题	239
附录 F PASCAL 语法图	240
参考文献	243

第 1 章

引 论

CHAPTER

1.1 什么是编译程序

语言是人与人进行交流的工具。程序设计语言则是人与计算机之间进行交流的工具。通过程序设计语言(例如,C或汇编语言),可将问题(例如,求1~100之间所有的素数)编写成计算机语言程序交给计算机,计算机则按程序的计算步骤计算出结果。

程序设计语言可分为两大类。第一类称为低级语言,包括机器语言、汇编语言以及其他面向机器的程序设计语言。这类语言对计算机的依赖性强、可读性差、编写程序的工作量大,只有对相应计算机结构比较熟悉,且经过一定训练的程序人员才能较好地使用此类语言。第二类称为高级语言,如BASIC,FORTRAN,PASCAL,C,JAVA等。高级语言不依赖于特定的计算机,有很好的通用性。它不论在算法描述的能力上,还是在编写和调试程序的效率上,都远比低级语言强。

然而,对特定的一台计算机而言,它的硬件,即计算机的核心部件只懂得自己的指令系统,即只能直接执行用本机器语言编写的机器代码程序,而不能直接执行用高级语言或汇编语言编写的程序。因此,要在计算机上使用除机器语言外的任何一种程序设计语言,就首先应使此种语言为计算机所“理解”。解决这一问题的方法有3种:一种是对程序进行翻译;另一种是对程序进行解释;还有一种是既有翻译又有解释,二者兼而有之。

所谓翻译,是指在计算机中放置一个能由计算机直接执行的翻译程序,它以某一种程序设计语言(源语言)所编写的程序(源程序)作为翻译或加工的对象,当计算机执行翻译程序时,就将它翻译为与之等价的另一种语言(目标语言)的程序(目标程序)。“源”和“目标”这两个术语总是相对于一类特定的翻译程序和翻译过程而言的。如果一个翻译程序的源语言是某种高级语言,其目标语言是某一计算机的汇编语言或机器语言,则称这种翻译程序为编译程序。汇编程序也是一种翻译程序,它的源语言和目标语言分别是某一种计算机的汇编语言和机器语言。

由此可见,欲按编译方式在计算机上执行用高级语言编写的程序,一般需要经过两个阶段:第一阶段称为**编译阶段**,其任务是由编译程序将源程序编译为目标程序,若目标程序不是机器代码,而是汇编语言程序,则还需经过汇编程序再汇编为机器代码程序。第二阶段称为**运行阶段**,其任务是在目标计算机上执行编译阶段所得到的目标程序。

用高级语言编写的程序也可以通过解释程序来执行。**解释程序**也以源程序作为它的输入,它与编译的主要区别是在解释程序的执行过程中不产生目标程序,而是解释执行源程序本身。即每读一条源程序代码,就将它解释成等价的若干条机器代码,并执行之,然后再去读下一条源程序代码。这种边解释边执行的工作方式效率很低,但由于解释程序的结构比编译程序简单,因此一些规模较小的语言,如 BASIC,也常采用此种方式。

单纯的解释程序并不多见,通常的做法是把编译和解释做某种程度的结合,即前面讲的第三种方法。例如 Java 语言,先将源程序翻译为某种易于解释的中间代码程序形式,然后再执行解释程序,对中间代码程序进行解释,得到我们需要的最终结果。这样做的好处是:

- (1) 编译程序的研制与具体的计算机平台没有关系,能获得最大的通用性。
- (2) 研制开发出来的语言编译程序易于修改和移植。
- (3) 单纯解释程序执行效率不高的缺陷得到了弥补。

因此,本书介绍的 PL/0 语言编译程序采用的正是这第三种方法。

1.2 编译过程概述

编译程序完成从源程序到目标程序的翻译工作,是一个复杂的整体过程。它与人们进行自然语言之间的翻译有许多相近之处。当把一种文字翻译为另一种文字,例如把一段英文翻译为中文时,通常需要经过下列步骤:

- ① 识别出句子中的一个个单词。
- ② 分析句子的语法结构。
- ③ 根据句子的语法结构和含义进行初步翻译。
- ④ 对译文进行修饰。
- ⑤ 写出最后的译文。

类似地,编译程序的工作过程一般也可以划分为 5 个阶段:词法分析、语法分析、语义分析与中间代码产生、优化、目标代码生成。

(1) 第一阶段,词法分析

词法分析的任务是:输入源程序,对构成源程序的字符串进行扫描和分解,识别出一个一个单词,也称单词符号或简称符号(symbol)。如关键字(begin、end、if、for、while 等)标识符、常数、算符和界符(TAB 符、空白符、分号等)。例如,对于 PASCAL 的循环语句:

```
while x>0 do x := x-1;
```

词法分析的结果是识别出如下的单词符号:

关键字 while

标识符	x
大于符	>
整常数	0
关键字	do
标识符	x
赋值符	:=
标识符	x
减法符	-
整常数	1

这些单词是组成上述 PASCAL 语句的基本符号。单词符号是语言的基本组成成分,是人们理解和编写程序的基本要素。识别和理解这些要素无疑是翻译的基础。如同将英文翻译成中文一样,如果对英语单词不理解,那就谈不上进行正确的翻译。在词法分析阶段的工作中所依循的是语言的词法规则(或称构词规则)。

(2) 第二阶段,语法分析

语法分析的任务是:在词法分析的基础上,根据语言的语法定义规则,识别出构成单词符号串的各类语法单位,如“因子(factor)”、“项(term)”、“表达式(expression)”、“语句(statement)”、“分程序(block)”和“程序(program)”等。通过语法分析,确定整个输入符号串是否构成语法上正确的“程序”。语法分析所依循的是语言的语法规则。语法规则通常用上下文无关文法描述。词法分析是一种线性分析,而语法分析是一种层次结构分析。例如,在很多语言中,符号串

$z := x + 0.618 * y;$

代表一个“赋值语句”,而其中的 $x + 0.618 * y$ 代表一个“算术表达式”。因而,语法分析的任务就是识别 $x + 0.618 * y$ 为算术表达式,进而再识别上述整个符号串属于赋值语句。

(3) 第三阶段,语义分析与中间代码产生

这一阶段的任务是:对语法分析所识别出的各类语法单位,分析其含义,并进行初步翻译(产生中间代码)。这一阶段通常包括两个方面的工作。首先,对每种语法单位进行静态语义检查,例如,变量是否有定义、类型是否正确等。如果语义正确,则进行另一方面工作,即进行中间代码的翻译。

“翻译”仅仅在这里才开始涉及。所谓“中间代码”是一种含义明确、便于处理的记号系统,它通常独立于具体的硬件。这种记号系统或者与现代计算机的指令形式有某种程度的接近,或者能够比较容易地把它变换成现代计算机的机器指令。例如,许多编译程序采用了一种与“三地址指令”非常近似的“四元式”作为中间代码。这种四元式的形式是:

算符	左操作数	右操作数	结果
----	------	------	----

它的意义是对“左、右操作数”进行某种运算(由“算符”指明),把运算所得的值作为“结果”保留下来。在采用四元式作为中间代码的情形下,中间代码产生的任务就是按语言的语法规则把各类语法单位翻译成四元式序列。例如,下面的赋值语句: