

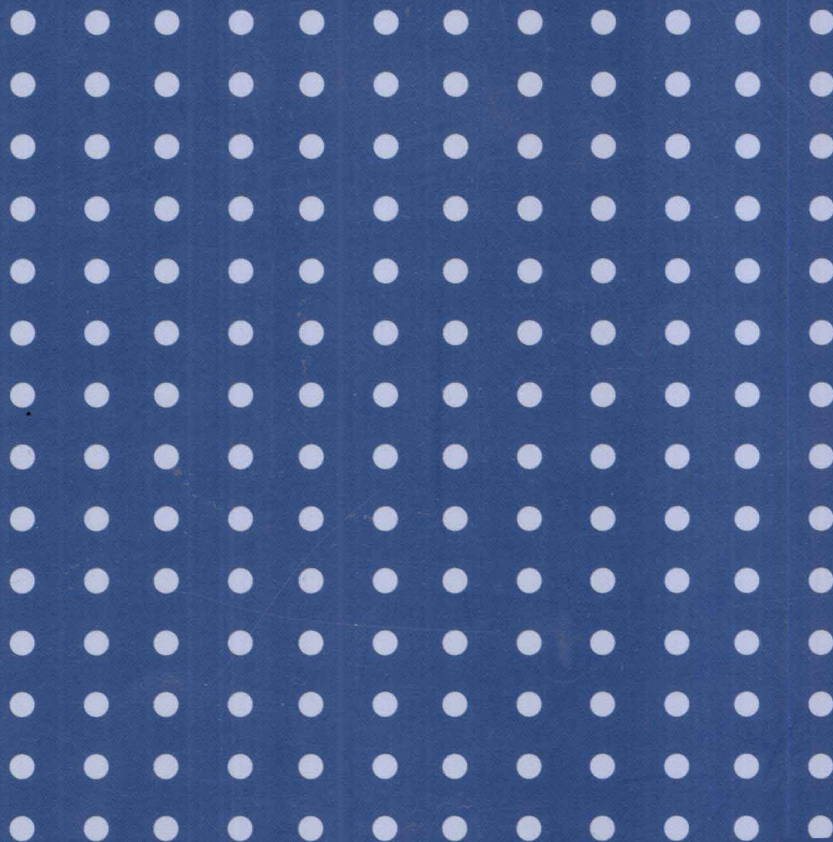


普通高等教育“十一五”国家级规划教材

重点大学计算机专业系列教材

编译原理及编译程序构造

张莉 杨海燕 史晓华 金茂忠 高仲仪 编著



清华大学出版社

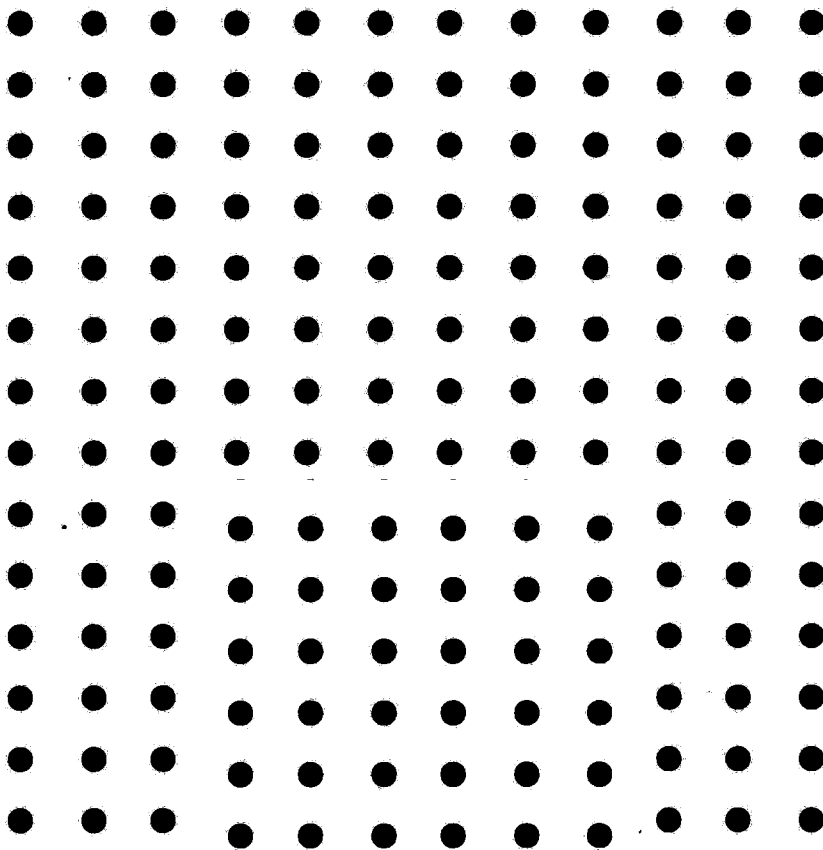


普通高等教育“十一五”国家级规划教材

重点大学计算机专业系列教材

编译原理及编译程序构造

张莉 杨海燕 史晓华 金茂忠 高仲仪 编著



清华大学出版社

北京

内 容 简 介

本书全面地介绍编译系统的构造和相关原理与技术。全书共 15 章,力求展示一个完整的编译过程,在此基础上介绍与编译系统相关的理论和方法。本书围绕这个完整的过程,还介绍并讨论了计算机领域三个非常重要的原理、概念和技术:高级程序设计语言的工作原理、程序模型间的转换方法,以及软件系统的概念。本书强调编译系统的构造及其相关技术,突出对工程师人才的培养要求。书中的算法和示例程序全部采用 C 语言风格。

本书适合作为高校计算机科学与技术专业本科“编译原理”、“编译技术”等专业课程的教材,也可供相关研究开发人员自学参考。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

编译原理及编译程序构造/张莉等编著. —北京:清华大学出版社,2011.6

(重点大学计算机专业系列教材)

ISBN 978-7-302-26314-2

I. ①编… II. ①张… III. ①编译程序—程序设计—高等学校—教材 IV. ①TP314

中国版本图书馆 CIP 数据核字(2011)第 149905 号

责任编辑:梁颖 战晓雷

责任校对:焦丽丽

责任印制:王秀菊

出版发行:清华大学出版社

地 址:北京清华大学学研大厦 A 座

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

邮 编:100084

社 总 机:010-62770175

邮 购:010-62786544

投稿与读者服务:010-62795954,jsjic@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015,zhiliang@tup.tsinghua.edu.cn

印 刷 者:北京市世界知识印刷厂

装 订 者:三河市溧源装订厂

经 销:全国新华书店

开 本:185×260 印 张:28.5 字 数:713 千字

版 次:2011 年 6 月第 1 版 印 次:2011 年 6 月第 1 次印刷

印 数:1~3000

定 价:45.00 元

产品编号:013803-01

出版说明

随着国家信息化步伐的加快和高等教育规模的扩大,社会对计算机专业人才的需求不仅体现在数量的增加上,而且体现在质量要求的提高上,培养具有研究和实践能力的高层次的计算机专业人才已成为许多重点大学计算机专业教育的主要目标。目前,我国共有 16 个国家重点学科、20 个博士点一级学科、28 个博士点二级学科集中在教育部部属重点大学,这些高校在计算机教学和科研方面具有一定优势,并且大多以国际著名大学计算机教育为参照系,具有系统完善的教学课程体系、教学实验体系、教学质量保证体系和人才培养评估体系等综合体系,形成了培养一流人才的教学和科研环境。

重点大学计算机学科的教学与科研氛围是培养一流计算机人才的基础,其中专业教材的使用和建设则是这种氛围的重要组成部分,一批具有学科方向特色优势的计算机专业教材作为各重点大学的重点建设项目成果得到肯定。为了展示和发扬各重点大学在计算机专业教育上的优势,特别是专业教材建设上的优势,同时配合各重点大学的计算机学科建设和专业课程教学需要,在教育部相关教学指导委员会专家的建议和各重点大学的大力支持下,清华大学出版社规划并出版本系列教材。本系列教材的建设旨在“汇聚学科精英、引领学科建设、培育专业英才”,同时以教材示范各重点大学的优秀教学理念、教学方法、教学手段和教学内容等。

本系列教材在规划过程中体现了如下一些基本组织原则和特点。

(1) 面向学科发展的前沿,适应当前社会对计算机专业高级人才的培养需求。教材内容以基本理论为基础,反映基本理论和原理的综合应用,重视实践和应用环节。

(2) 反映教学需要,促进教学发展。教材要能适应多样化的教学需要,正确把握教学内容和课程体系的改革方向。在选择教材内容和编写体系时注意体现素质教育、创新能力与实践能力的培养,为学生知识、能力、素质协调发展创造条件。

(3) 实施精品战略,突出重点,保证质量。规划教材建设的重点依然是专业基础课和专业主干课;特别注意选择并安排了一部分原来基础比较好的优秀教材或讲义修订再版,逐步形成精品教材;提倡并鼓励编写体现重点大学

计算机专业教学内容和课程体系改革成果的教材。

(4) 主张一纲多本,合理配套。专业基础课和专业主干课教材要配套,同一门课程可以有多样本具有不同内容特点的教材。处理好教材统一性与多样化的关系;基本教材与辅助教材以及教学参考书的关系;文字教材与软件教材的关系,实现教材系列资源配套。

(5) 依靠专家,择优落实。在制订教材规划时要依靠各课程专家在调查研究本课程教材建设现状的基础上提出规划选题。在落实主编人选时,要引入竞争机制,通过申报、评审确定主编。书稿完成后要认真实行审稿程序,确保出书质量。

繁荣教材出版事业,提高教材质量的关键是教师。建立一支高水平的以老带新的教材编写队伍才能保证教材的编写质量,希望有志于教材建设的教师能够加入到我们的编写队伍中来。

教材编委会

本教材是在高仲仪、金茂忠老师 1991 年出版的《编译原理及编译程序构造》的基础上修改补充编纂而成的,融入了我们课程组十多年的教学经验。在教材的整体结构上保持了原书的结构,做了部分调整,对原书中的部分内容进行了修订,算法和示例程序改为 C 语言风格,删除了一些已不常用的语言,补充了一些新的内容,增加了编译优化部分的内容。由于本书仍然定位为本科生的第一门编译课程的教材,因此对于新出现的并行编译、较难的一些优化技术,本书没有涉及。

考虑到北京航空航天大学作为工科类院校的特点,我们的编译课程一直采用了“编译技术”这个名字,本书也将继续采用原书的书名《编译原理及编译程序构造》,意在强调编译系统的构造及其相关技术,突出对工程师人才的培养要求。

我们认为编译技术是计算机专业的一门非常重要的课程。虽然本书重在介绍编译系统的构造和相关原理与技术,但是在这个过程中,我们介绍并讨论了计算机领域三个非常重要的原理、概念和技术:

(1) 高级程序设计语言的工作原理;

(2) 程序/模型间转换的方法:实现模型从一种语言表达形式到另一种语言表达形式的(语义)等价转换方法及其相关技术;

(3) 软件系统的概念。对于大多数本科生而言,编译系统是他们构造的第一个相对完整的软件系统,因此我们非常强调系统的完整性。

我从 1987 年学编译,到 1998 年开始讲编译,这么多年来我不断地听到大家说编译是公认的难学亦难教的一门课,理论的东西太多,同时实践性很强。经过十多年的教学,我感觉编译应该不那么难学,因为编译的主线非常清晰。可为什么大家觉得难学呢?因为我们开始讲了过多的理论,从语法基础,到词法分析部分的自动机,再到各种语法分析方法。同学们被灌输了很多的算法,反而忽略了编译的根本任务:翻译,即程序从一种语言形式到另一种语言形式的等价翻译。这就是为什么学生往往在完成课程设计(一个小型编译系统)之后才恍然大悟:我有点明白编译是干什么的了。

显然,这是由于在教学中我们有些舍本求末了。编译技术作为一门重要

的专业基础课程,它讲述的是计算机领域中的一个重要的主题:即程序从一种语言形式到另一种语言形式的等价转换。编译课程通过从高级语言程序到低级语言程序的转换向我们介绍了相关的理论、方法和技术,并进一步阐明了高级语言的工作原理。因此,我们有责任给学生一个完整的编译过程。在这个过程的基础上或者这个过程中介绍相关的理论和方法。

本书的编写仍然沿用了传统的组织形式,以保证知识体系的完整。但是,在课程安排上,可以采用多种方式组织教学,比如:

(1) 首先给学生一个完整的最简单的编译过程(翻译过程),见图1中A和B列对应的知识点,在此基础上再根据培养要求选讲代码优化、词法分析器的自动生成技术和其他语法分析技术(都可以自动生成),对应于图1中C列。我们经过三年的尝试,发现教学效果较好。在讲完A和B列后,学生可以看懂附录中的编译程序,并可以自己设计一个小型编译器。

(2) 按照本书的章节顺序讲解,但是建议教师能随时回到编译过程,让学生理解当前讲解的内容处于编译过程的何处。

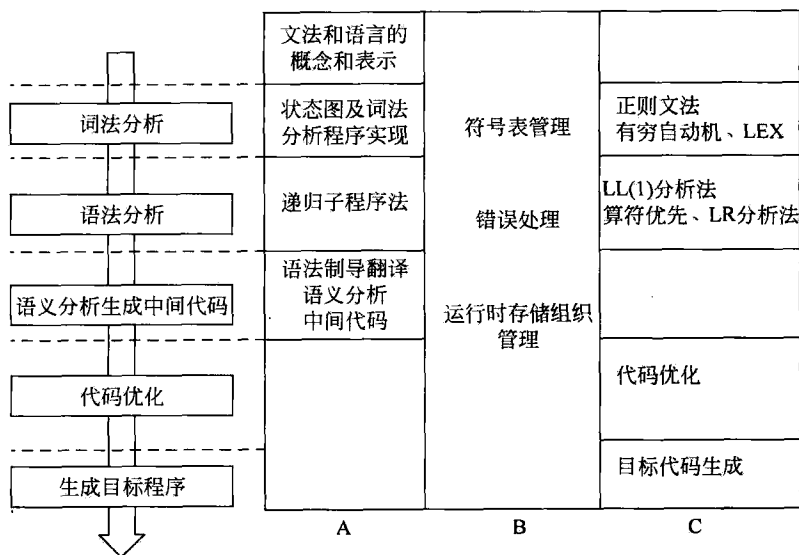


图1 编译过程和知识点组织

因此,我们建议采用本书的老师可根据各学校的培养定位选择讲课的顺序和内容。

本书的完成凝聚着北航编译课程组全体老师的心血,几代师生共同完成了本书的撰写。金茂忠负责第1、4、13章,高仲仪、张莉负责第2、3章,张莉负责第9、10章,杨海燕负责第5、6、7、8、14章,史晓华负责第11、12、15章。张莉负责对全书进行了审校。还要感谢邱翔、田亚伟、申菊芳等在本书的习题、排版、校对等方面给予的帮助。

在本书编写过程中,还参考和引用了国内外优秀编译教材和著作中的相关内容,在此谨向原书著(译)者深表敬意和感谢。本书难免还有不少疏漏和差错之处,欢迎读者批评指正,以便在本书再版时进行修正。

联系方式 compile@buaa.edu.cn

课程网站: <http://www.sei.buaa.edu.cn/compile/> (国家精品课程网站)

张莉

2011年5月

第 1 章 绪论	1
1.1 什么是程序设计语言	1
1.1.1 程序设计语言的定义方法.....	2
1.1.2 程序设计语言的处理系统.....	2
1.1.3 编译程序和解释程序.....	3
1.1.4 T形图.....	4
1.2 与编译程序相关的处理系统	5
1.2.1 各种翻译程序.....	5
1.2.2 预处理器.....	6
1.2.3 宏处理器.....	6
1.3 编译程序和程序设计环境	7
1.4 编译程序的构造	8
1.5 编译技术在软件工程中的应用.....	13
练习 1	15
第 2 章 文法和语言的概念和表示	16
2.1 文法的非形式讨论.....	16
2.1.1 语法树	16
2.1.2 规则	17
2.1.3 由规则推导句子	17
练习 2.1	19
2.2 符号、符号串及其集合的运算	20
2.2.1 字母表和符号串	20
2.2.2 符号串及其集合的运算	20
练习 2.2	22
2.3 文法和语言的形式定义.....	22
2.3.1 文法的形式定义	22

2.3.2	推导的形式定义	24
2.3.3	语言的形式定义	24
2.3.4	递归规则与递归文法	26
2.3.5	短语、简单短语和句柄	27
练习 2.3	28
2.4	语法树和二义性	29
2.4.1	推导与语法树	29
2.4.2	文法的二义性	33
练习 2.4	36
2.5	符号串的分析	37
2.5.1	自顶向下分析	38
2.5.2	自底向上分析	38
2.6	有关文法的实用限制	40
练习 2.5	41
2.7	扩充的 BNF 表示和语法图	41
2.7.1	扩充的 BNF 表示	41
2.7.2	语法图	43
2.8	文法和语言分类	43
第 3 章	词法分析	46
3.1	词法分析程序的功能及实现方案	46
3.2	单词的种类及词法分析程序的输出形式	47
3.3	正则文法及其状态图	48
3.3.1	状态图	49
3.3.2	状态图的使用	49
3.4	词法分析程序的设计与实现	50
3.4.1	文法及其状态图	50
3.4.2	词法分析程序的构造	51
3.4.3	词法分析程序的实现	53
练习 3.1	56
3.5	正则文法与正则表达式	56
3.5.1	正则表达式	57
3.5.2	正则文法转换为正则表达式	59
3.5.3	正则表达式转换为正则文法	59
3.6	有穷自动机(FA)	60
3.6.1	确定的有穷自动机(DFA)	60
3.6.2	不确定的有穷自动机(NFA)	61
3.6.3	NFA 的确定化	63
3.6.4	确定有穷自动机的化简(最小化)	65

3.6.5	正则表达式与有穷自动机的等价性	67
3.6.6	正则文法与有穷自动机的等价性	70
3.7	词法分析程序的自动生成器	71
3.7.1	LEX 源程序(LEX 的输入文件)	72
3.7.2	LEX 的实现	73
练习 3.2	76
第 4 章	语法分析	78
4.1	自顶向下分析方法	78
4.1.1	带回溯的自顶向下分析方法	78
4.1.2	存在的问题及解决办法	80
练习 4.1	85
4.2	递归下降分析法(递归子程序法)	85
练习 4.2	90
4.3	LL(1)分析方法	91
4.3.1	LL(1)分析器的逻辑结构及工作过程	91
4.3.2	LL(1)分析表的构造方法	94
练习 4.3	97
4.4	自底向上分析方法	99
4.5	算法优先分析法	101
4.5.1	方法概述	101
4.5.2	直观算符优先分析法	102
4.5.3	算符优先分析法的进一步讨论	105
练习 4.4	110
4.6	LR 语法分析方法	110
4.6.1	概念和术语	111
练习 4.5	112
4.6.2	LR 分析算法	113
练习 4.6	117
4.6.3	LR 文法	117
4.6.4	构造 SLR 语法分析表	118
练习 4.7	122
练习 4.8	124
4.6.5	构造规范 LR 语法分析表	125
练习 4.9	130
4.6.6	构造 LALR 语法分析表	130
练习 4.10	134
4.7	二义文法的应用	135
4.8	LR 语法分析中的错误恢复	136

练习 4.11	137
第 5 章 符号表管理技术	138
5.1 概述	138
5.1.1 什么是符号表	138
5.1.2 何时建立和访问符号表	138
5.1.3 符号表的重要性和作用	140
5.1.4 在符号表上的操作	140
5.2 符号表的组织和内容	141
5.2.1 符号表的结构与内容	141
5.2.2 符号表的组织方式	143
5.3 非分程序结构语言的符号表组织	144
5.3.1 标识符的作用域及基本处理方法	145
5.3.2 符号表的组织方式	145
5.4 分程序结构语言的符号表组织	150
5.4.1 标识符的作用域及基本处理方法	150
5.4.2 定位和重定位操作	151
5.4.3 符号表的组织方式	152
练习 5	155
第 6 章 运行时的存储组织及管理	157
6.1 静态存储分配	157
练习 6.1	159
6.2 动态存储分配	159
6.2.1 活动记录	160
6.2.2 参数区	161
6.2.3 display 区	161
6.2.4 运行时的地址计算	163
6.2.5 递归过程的处理	164
练习 6.2	166
第 7 章 源程序的中间形式	168
7.1 波兰表示	168
7.2 N 元表示	169
7.3 抽象语法树	171
7.4 抽象机代码	172
7.4.1 可移植性和抽象机	172
7.4.2 Pascal 的 P-code 抽象机	173
7.4.3 P-code 指令	174

练习 7	175
第 8 章 错误处理	176
8.1 概述	176
8.2 错误的分类	177
8.3 错误的诊察与报告	177
8.4 错误处理技术	179
8.4.1 错误改正	179
8.4.2 错误局部化处理	179
8.4.3 目标程序运行时的错误检测与处理	182
8.4.4 遏止重复的错误信息	182
第 9 章 语法制导翻译技术	183
9.1 翻译文法(TG; Translation Grammar)	184
9.2 语法制导翻译	186
9.3 属性翻译文法(ATG; Attribute TG)	187
9.3.1 综合属性	187
9.3.2 继承属性	189
9.3.3 属性翻译文法概述	190
9.3.4 属性翻译文法举例——算术表达式的翻译	191
练习 9.1	193
9.4 自顶向下语法制导翻译	195
9.4.1 翻译文法的自顶向下翻译	195
练习 9.2	200
9.4.2 属性翻译文法的自顶向下翻译	201
练习 9.3	212
9.5 自底向上语法制导翻译	213
9.5.1 波兰翻译	214
9.5.2 S-属性文法	214
练习 9.4	216
第 10 章 语义分析和代码生成	218
10.1 语义分析的概念	218
10.2 栈式抽象机及其汇编指令	220
10.3 声明的处理	221
10.3.1 常量类型	222
10.3.2 简单变量	223
10.3.3 数组变量	224
10.3.4 记录变量	226

10.3.5	过程声明	226
10.4	表达式	227
10.5	赋值语句	233
10.6	控制语句	234
10.6.1	if 语句	234
10.6.2	分情形语句	236
10.6.3	repeat-while 语句	238
10.6.4	for 循环语句	239
10.7	过程调用和返回	240
10.7.1	参数的基本传递形式	241
10.7.2	过程调用	242
10.7.3	返回语句和过程终止	245
10.8	输入和输出语句	246
10.8.1	输入语句	246
10.8.2	输出语句	248
10.9	编译程序的辅助功能	249
	练习 10	250
第 11 章	代码优化	251
11.1	基本块和流图	252
11.2	基本块内优化	254
11.2.1	基本块的 DAG 图表示	254
11.2.2	消除局部公共子表达式	255
11.2.3	数组、指针及函数调用	255
11.2.4	从 DAG 图重新导出中间代码	256
11.2.5	窥孔优化	258
11.2.6	常数合并和传播	259
11.3	全局优化	260
11.3.1	数据流分析	260
11.3.2	活跃变量分析	264
11.3.3	定义-使用链、网和冲突图	266
11.3.4	消除全局公共子表达式	269
11.3.5	复制传播	270
11.3.6	死代码删除	270
11.4	循环优化	271
11.4.1	循环交换	271
11.4.2	循环展开	272
11.4.3	代码外提和循环强度削弱	272
	练习 11	273

第 12 章 目标代码生成	275
12.1 微处理器体系结构简介	276
12.1.1 指令集架构	276
12.1.2 存储层次架构	279
12.1.3 流水线	281
12.2 地址空间	283
12.2.1 程序地址空间的实例分析	284
12.2.2 程序运行栈的设计	286
12.3 寄存器的分配和指派	288
12.3.1 全局寄存器分配	289
12.3.2 临时寄存器分配	291
12.4 指令选择	292
练习 12	294
第 13 章 编译程序生成方法和工具	296
13.1 编译程序的书写语言	296
13.2 自展	297
13.3 移植	298
13.4 编译程序的生成工具	299
13.4.1 语法分析器的生成器 Yacc	299
13.4.2 用 Yacc 处理二义文法	302
13.4.3 用 Lex 建立 Yacc 的词法分析器	304
13.4.4 Yacc 的错误恢复	305
练习 13	306
第 14 章 PL/0 简单编译系统	307
14.1 PL/0 语言	307
14.2 PL/0 编译系统结构	311
14.3 PL/0 的词法分析	312
14.4 PL/0 的语法分析	313
14.5 出错处理	315
14.6 目标代码的生成和解释执行	316
14.7 PL/0 程序编译和运行举例	318
第 15 章 Pascal-S 编译系统	328
15.1 Pascal-S 语言	328
15.2 Pascal-S 编译程序的结构	333
15.3 Pascal-S 编译程序	336

15.3.1	表格	337
15.3.2	编译初启	342
15.3.3	实用程序	343
15.3.4	词法分析及处理	343
15.3.5	语法分析处理	344
15.3.6	出错处理	349
15.4	Pascal-S 解释执行程序	351
15.4.1	P-code 指令系统	351
15.4.2	运行栈	353
15.4.3	运行时的 display	354
15.4.4	运行出错处理和现场剖析打印	355
15.5	编译及运行的例子	356
附录 A	PL/0 编译系统源代码	366
附录 B	Pascal-S 编译系统源代码	382
参考文献	439

绪 论

第 1 章

把高级程序设计语言翻译成汇编语言或机器语言的工作称为编译,完成这项翻译工作的软件系统称为编译程序或编译器。本书介绍编译原理及编译器的设计和实现技术。同时,编译原理和技术也是程序设计语言的工作原理和设计高质量程序的基础。

编译程序设计原理和构造技术具有十分普遍的意义,是计算机科学工作者、软件工程技术人員所必须掌握的专业基础知识,可以说每一个计算机科学工作者在他一生的研究生涯中,都会反复用到与编译有关的原理和技术。编译程序的设计涉及程序设计语言、计算机体系结构、形式语言理论、算法、数据结构和软件工程等学科。本章通过描述程序设计语言、编译过程、编译程序的组成以及编译程序的工作环境来简要介绍本书的主要内容。

1.1 什么是程序设计语言

语言是思维的表现形式,人与人之间通过自然语言来传达彼此的思想,因此,语言是人与人之间用来传递信息的媒介和手段。在计算机的应用领域,程序设计语言(也称为“程序语言”或“编程语言”)充当了人与问题以及协助解决该问题的计算机之间的通信工具,简而言之,人与计算机之间的中介就是程序设计语言。一种有效的程序设计语言能提高计算机程序的开发效率和对问题的表达能力,它必须能填补人们通常的思维方式和计算机执行所要求的精确性之间的沟壑。程序设计语言正是沿着为更好地满足这方面要求的方向而向前发展的。

一个好的程序设计语言应该容易表达人想要计算机所做的事情和便于进行程序的开发,即图 1.1 中的间隔(1)要小。如果沿着使该间隔减小的方向来看一下程序设计语言的发展道路,就是从机器语言、汇编语言等与机器相关的低级语言向通用编程语言(面向过程语言)、面向问题语言、逻辑编程语言、面向对象语言等与机器无关的高级语言发展的过程。

但是,越是沿着这条道路前进,图 1.1 中的间隔(2)就会越来越大,而用

来填补该间隔的就是程序设计语言的处理系统。

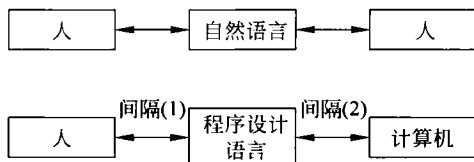


图 1.1 程序设计语言的位置

1.1.1 程序设计语言的定义方法

程序设计语言是一种人工语言。各种程序设计语言是有严格定义的(即该语言的规格说明)。语言的定义由语法和语义来规定。

例如,考虑以下简单的赋值语句:

$$s := (\text{up} + \text{low}) * \text{h} / 2 \quad (1-1)$$

赋值语句的形式为:

标识符 := 表达式

而表达式的形式为:

表达式 * 表达式
 表达式 / 表达式
 表达式 + 表达式
 (表达式)
 :

我们把对上述语言结构形式的描述称为文法(grammar)或语法(syntax)。

下面再来看一下式(1-1)的含义。它表示变量 up 和变量 low 相加,再和变量 h 相乘,然后除以 2,最后把结果赋给变量 s。如果这里 up、low、h、s 是实型变量,则 +、*、/ 就是实型数的加法、乘法和除法,式(1-1)中的 2 应该先转换成实数 2.0 以后再进行实型数的除法运算。我们把上述语言所表达内容的含义称为语义。

如何表示语言的语法和语义,将在第 2 章中详述。程序设计语言的定义由语言的规格说明书来描述。通常用巴科斯范式(BNF, Backus-Naur Form)来表示上下文无关文法语言的语法。

而表示程序设计语言的语义有很大困难。通常我们在程序设计语言规格说明书中用自然语言来描述语义。语义的形式化定义在形式语义学中研究,但这些定义难以理解。根据形式语义构造的语言分析系统效率较低且实现困难,至今尚未有成功的实用处理系统。本书将采用近年来广泛应用的属性文法来描述程序设计语言的语义。属性文法是一种半形式化的语义描述方法,它比较容易理解,而且可根据它构造出相当实用的分析系统。

1.1.2 程序设计语言的处理系统

程序设计语言的规格说明书给出了该语言的定义。当输入该语言的程序时,程序语言