

内 容 提 要

本书是根据教育部提出的非计算机专业计算机基础教学三层次课程体系的要求以及工科计算机基础教学指导委员会的有关精神而组织编写的教材。

本书主要阐述程序设计的基本概念、方法和软件工具及编程技术。全书共七章:第一章概要介绍软件的基本概念和软件技术的发展过程;第二章至第四章主要讲述结构化程序设计的基本概念、方法及C语言编程技术;第五章主要讲述基本的数据结构和算法的模型概念以及算法的C语言实现技术;第六章主要讲述面向对象程序设计方法的基本概念和C++编程技术;第七章主要讲述软件工程的基本概念和工程化观点,介绍了软件生命周期、软件开发模型和软件过程。

本书可作为高等学校计算机或非计算机类本、专科各专业软件技术基础课程的教材,亦可供从事计算机应用开发的科技人员学习参考。

图书在版编目(CIP)数据

计算机软件技术基础/梁光春,曾一,熊壮编著. 重庆:重庆大学出版社,2000.2
大学计算机基础教育系列教材
ISBN 7-5624-1973-6

I. 计… II. ①梁…②曾…③熊… III. 软件-高等学校-教材 IV. TP31

中国版本图书馆 CIP 数据核字(2000)第 13241 号

计算机软件技术基础

梁光春 曾一 熊壮 编著
责任编辑 王 勇

*

重庆大学出版社出版发行
新华书店经销
重庆建筑大学印刷厂印刷

*

开本:787×1092 1/16 印张:19 字数:474 千

2000 年 1 月第 1 版 2000 年 1 月第 1 次印刷

印数:1-5 000

ISBN 7-5624-1973-6/TP·214 定价:22.00 元

大学计算机基础教育系列教材

编 写 委 员 会

主 任 杨天怡

副主任 曾 一 黄 勤 朱庆生 陈今润

编 委 梁光春 李宝珠 郭松涛 熊 壮

陈 莉 李夔宁 曾 一 鲜晓东

宋亚莉 高富强 黄 勤 杨天怡

序

计算机技术的飞速发展以及计算机技术与各学科日趋紧密结合,对高等学校的人才培养提出了新的要求,计算机知识和应用能力已成为当代大学生知识能力结构的重要组成部分。

为了加强非计算机专业学生的计算机基础教育,提高人才培养质量,我们进行了面向 21 世纪计算机基础教学内容及课程体系的改革,提出了《加强非计算机专业计算机基础的教学的意见》和《非计算机专业计算机基础教学内容及课程体系》,教学内容反映了计算机技术及应用的现状与发展。

我们根据《非计算机专业的计算机基础教学内容及课程体系》的要求编写的本系列教材,其基本的目的是使学生掌握计算机软、硬件基础知识,培养学生用计算机分析、解决问题的意识和解决本专业及相关领域中实际问题的能力。为了与课程体系的三个层次紧密结合,我们相应地编写了三个层次方面的教材。

计算机文化基础层相关的教材有:《计算机文化基础》及配套的《计算机文化基础实验指导书》。其主要任务是使学生掌握在信息社会里能更好地工作、学习和生活所必须的计算机基础知识与基本操作技能,培养学生的计算机文化意识。《EXCEL 及其应用》是为文科类学生编写的深层次计算机文化基础课程教材。

计算机技术基础层的教材有:《计算机软件技术基础》、《计算机硬件技术基础》以及配套的《计算机软件技术基础实

验指导书》、《计算机硬件技术基础实验指导书》等。其主要任务是使学生掌握计算机软件、硬件的基本知识,培养学生利用计算机解决本专业及相关领域中实际问题的初步能力。

计算机应用基础层的教材有:《计算机信息管理基础》、《计算机网络技术基础》、《计算机控制技术》、《单片微型计算机原理及应用》、《计算机辅助设计基础》、《计算机仿真技术基础》等。其主要任务是进一步培养学生利用计算机获取信息、处理信息和解决实际问题的意识和能力,增强学生建构本专业及相关领域中计算机应用系统的能力。

愿本套教材的推出,为大学非计算机专业计算机基础教学的发展,为培养更多适应 21 世纪需要的技术人才做出贡献。

《大学计算机基础教育
系列教材》编委会
1999 年 9 月 16 日

前言

计算机技术的迅速发展,把人们从辉煌的 20 世纪带入到崭新的 21 世纪。回顾几十年的发展历程,软件技术的发展最令人激动,从最富于智慧的高级程序设计语言的繁衍到结构化程序设计方法的逐步形成,从试图解决软件危机的软件工程的提出到风靡世界的面向对象程序设计方法及工具的面世及其应用,无不使人心潮澎湃,浮想联翩。

但是,作者在讲述这个迅速变化的领域时,却非常谨慎小心,认为软件技术基础不是语言或方法的单个方面,而应该是方法和工具两方面的有效结合。方法作为理论引入,工具则作为必要的实践环节。这种结合在本书中的表现主要分为四个方面:

第一,把传统的结构化程序设计方法与 20 世纪最具影响力之一、且具有结构化程序设计语言特征、至今仍被广泛使用的 C 语言相结合。

第二,把基本的数据结构作为数据的组织方法和算法模型与 C 语言描述的算法相结合,突出数据结构在解决实际问题时的重要性。

第三,把可以将现实世界中的事物直接映射到软件系统的解空间,使得复杂的问题易于解决和易于维护的面向对象程序设计方法与具有对象及类、对象的封装性、继承性、多态性等特征的 C++ 语言相结合,在结合的过程中注重方法的比较及其适用范围。

第四,把软件工程的思想与软件生产过程进行结合,强调

应用软件开发过程中的工程概念。

为了使这四部分的内容有机地结合起来,既能给出较完整的、最基本的软件技术的基本知识,又易为非计算机专业的学生和其他初学者所掌握,本书注重基础,强调应用,阐述中结合图例,深入浅出,通俗易懂,并配有《计算机软件技术基础实验指导书》。

本书共分七章。第一章概要介绍软件的基本概念和软件技术的发展过程;第二章至第四章主要讲述结构化程序设计的基本概念和方法,详细介绍了 C 语言的特征和机制,通过例题阐述编程技术;第五章主要讲述基本的数据结构和算法的模型概念以及相应算法的 C 语言实现技术;第六章主要讲述面向对象程序设计方法的基本概念和 C++ 编程技术;第七章主要讲述软件工程的基本概念和工程化观点,介绍了软件生命周期、软件开发模型和软件过程。

本书第一章和第六章由梁光春教授编写;第七章由曾一副教授编写;第二章、第三章和第四章由熊壮编写;第五章由梁光春教授和曾一副教授共同编写。

本书的组织和编写得到了我校教务处和计算机科学与工程学院领导的关心和支持,杨天怡教授对本书的结构和内容提出建设性建议,出版社的王勇、李长惠为该书的编辑、出版做了大量的工作,编者在此表示衷心的感谢。

由于编写时间仓促,作者水平有限,书中错误和不妥之处在所难免,敬请读者批评指正。

编 者

2000 年 1 月

目 录

MULU

第1章 概 论	 (1)
1.1 计算机系统	 (1)
1.1.1 计算机系统的发展	 (1)
1.1.2 软件与软件分类	 (2)
1.2 计算机语言和语言处理程序	 (3)
1.2.1 计算机语言概述	 (3)
1.2.2 计算机语言处理程序概述	 (4)
1.3 计算机软件技术概述	 (10)
1.3.1 程序设计	 (10)
1.3.2 数据结构	 (11)
1.3.3 操作系统和计算机网络	 (12)
1.3.4 数据库管理系统	 (18)
1.3.5 软件工程	 (24)
1.3.6 面向对象方法学	 (25)
1.4 软件开发环境	 (26)
1.4.1 软件开发方法	 (26)
1.4.2 软件开发模型	 (27)
1.4.3 软件开发环境	 (27)
第2章 C程序设计基础	 (29)
2.1 C程序设计入门	 (29)
2.1.1 C语言的发展简史与特点	 (29)
2.1.2 C程序的基本结构	 (30)

2.1.3	C语言的基本数据类型	(31)
2.1.4	基本运算符和表达式	(36)
2.1.5	不同类型数据混合运算及数据转换	(40)
2.1.6	C程序设计初步	(41)
2.2	C程序的控制结构	(45)
2.2.1	概述	(45)
2.2.2	分支结构	(45)
2.2.3	循环结构	(51)
2.2.4	其他简单控制结构	(54)
2.2.5	算法与程序举例	(56)

第3章 模块化程序设计

3.1	模块化程序设计	(60)
3.1.1	模块化程序设计概念	(60)
3.1.2	函数	(61)
3.1.3	模块组合与函数的嵌套调用	(65)
3.1.4	函数的递归调用	(66)
3.1.5	模块结构与程序结构	(69)
3.1.6	编译预处理	(76)
3.2	指针与函数	(81)
3.2.1	指针的概念	(81)
3.2.2	指针变量作函数的参数	(84)
3.2.3	函数的指针与函数调用	(87)
3.2.4	返回指针值的函数	(89)

第4章 C程序中的构造数据类型和文件

4.1	构造类型和指针	(91)
4.1.1	数组与指针	(91)
4.1.2	结构体和指针	(114)
4.1.3	共用体	(127)
4.1.4	枚举和位运算	(133)
4.1.5	位段及应用	(140)
4.2	文件	(142)
4.2.1	文件概念与文件类型指针	(142)
4.2.2	文件的打开与关闭	(144)
4.2.3	文件的读写	(145)

4.2.4 文件的定位和随机读写	(156)
4.2.5 文件操作的错误检测	(159)

第5章 基本数据结构

5.1 数据结构的基本概念	(161)
5.1.1 什么是数据结构	(161)
5.1.2 几种基本结构	(162)
5.1.3 数据结构的存储方式	(163)
5.1.4 数据结构的基本运算	(164)
5.1.5 抽象数据类型和数据结构的描述	(164)
5.2 线性数据结构	(165)
5.2.1 线性表及其顺序存储结构	(165)
5.2.2 线性表的链接存储结构	(168)
5.2.3 栈和队列	(175)
5.3 树型数据结构	(184)
5.3.1 树的基本概念	(184)
5.3.2 二叉树	(186)
5.4 集合与查找	(189)
5.4.1 集合	(190)
5.4.2 线性表表示下的集合与查找	(190)
5.4.3 二叉树表示下的集合与查找	(194)
5.4.4 散列表表示下的集合与查找	(197)
5.5 图型数据结构	(203)
5.6 排序	(203)
5.6.1 简单排序法	(204)
5.6.2 希尔排序	(206)

第6章 面向对象程序设计语言C++

6.1 C++ 的起源与特点	(208)
6.2 C++ 封装性:对象类	(211)
6.2.1 类的说明和类的实例	(211)
6.2.2 实例的初始化和析构	(215)
6.2.3 有关类的几点说明	(221)
6.3 C++ 继承性:导出类	(226)
6.3.1 继承	(226)
6.3.2 继承关系分类	(227)

6.3.3	在派生类中初始化基类成员	(232)
6.3.4	继承的多义性	(234)
6.3.5	基类指针与派生类指针	(238)
6.3.6	基类对象的赋值	(239)
6.3.7	保护的构造函数	(241)
6.3.8	派生类初始化的缺省操作和赋值的缺省操作	(242)
6.3.9	继承关系与其他	(244)
6.3.10	程序举例	(246)
6.4	多态性:虚函数	(249)
6.4.1	编译时多态性与运算符的重载	(249)
6.4.2	运行时多态性与虚函数	(263)

第7章 软件工程基础

第7章	软件工程基础	(270)
7.1	软件工程概述	(270)
7.1.1	软件工程的产生	(270)
7.1.2	软件工程概念及目标	(271)
7.1.3	软件工程研究的内容	(271)
7.1.4	软件工程师的任务及其关心的问题	(276)
7.2	软件开发模型	(276)
7.2.1	软件生命周期	(276)
7.2.2	软件开发模型	(277)
7.3	软件工程过程	(282)
7.3.1	基本过程	(283)
7.3.2	支持过程	(284)
7.3.3	组织过程	(285)
7.3.4	软件过程的改进及 CMM 模型	(286)
7.3.5	ISO 9000 - 3	(287)

参考文献

参考文献	(289)
------------	-------

概 论

1.1 计算机系统

由计算机的硬件和软件组成统一的整体称为计算机系统。计算机系统的硬件是构成计算机系统的全部硬设备,如中央处理机(CPU),内存储器(RAM、ROM),外存储器(软盘、硬盘、磁带等)以及各种输入设备(键盘、鼠标、显示器、打印机等)。为了充分发挥计算机硬件的使用效率还需要配备计算机的软件,它是构成计算机系统的全部软设备,如系统软件和应用软件。

1.1.1 计算机系统的发展

计算机系统从 1950 年到现在约有近 50 年的历史,经历了四个时期的发展过程。图 1.1 描述出了以计算机为基础的计算机系统的发展过程。

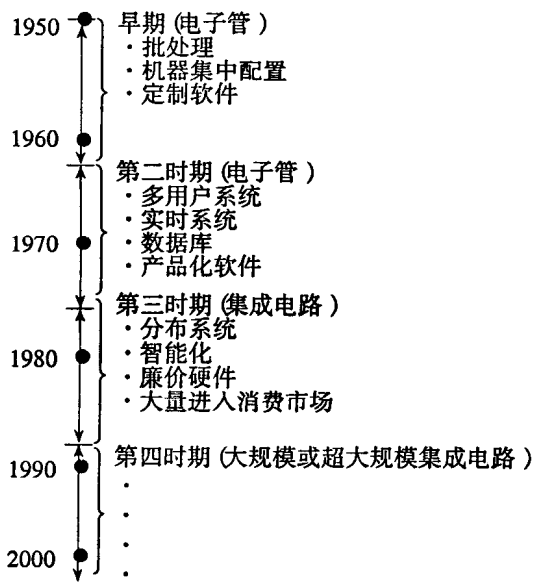


图 1.1 计算机系统的发展过程

1) 早期

在计算机系统发展的早期,硬件经历了不断的变化,而软件则被多数人作为一种事后工作来看待,计算机程序设计几乎没有什么系统的方法可以遵循,软件的开发进度缓慢,或成本一增再增,而软件仅是为每一种用途分别设计,通用性很差,没有形成软件产品。在此期间,多数系统采用批处理工作方式。

2) 第二个时期

在这个时期是以产品化软件的使用和“软件车间”的出现为特点,随着以计算机为基础的系统日益增多,计算机软件库开始膨胀,软件维护的开销开始令人恐慌,更坏的是许多软件带有个人的色彩使得它们实际上不可能维护,“软件危机”开始了。

3) 第三个时期

计算机系统发展的第三个时期从 70 年代初期开始,一直延续到 80 年代初期,在这个时期分布式系统(多个计算机、各机器并行执行和相互通信)极大地增加了计算机为基础的系统的复杂性。由于微处理机和有关部件的功能越来越强且价格越来越低,因此,在计算机应用领域中,具有“嵌入智能”的产品取代了较大的计算机。硬件的迅速发展已经超过我们提供的支持软件的能力,“软件危机”日益严重,为了维护软件,消耗掉了数据处理预算的 50% 以上,而软件的开发又跟不上新系统对软件需求的步伐。为了对付不断增长的软件危机,软件工程才得到了认真的对待。

4) 第四个时期

具有更大主存的 16 位和 32 位微处理机,为以计算机为基础的系统开辟了全新的应用领域。

1.1.2 软件与软件分类

软件是所有程序、数据和文档的总称,计算机软件通常可以分为系统软件和应用软件。

1) 系统软件

系统软件是由计算机系统的设计者和生产者提供的程序和使用及维护手册的总称,这些程序的作用在于扩充计算机功能,控制计算机高效率运行,为用户提供更多的方便。系统软件主要包括:操作系统,程序设计的各种语言的处理程序(汇编器、解释器和编译器),数据库管理系统,计算机网络软件,诊断和故障处理程序,编辑程序和调试程序等。由于配制了各种系统软件,从而大大地改善了用户使用计算机环境。

2) 应用软件

应用软件则是用户利用计算机系统提供的功能为解决特定问题而编制的程序及其使用说明书和维护手册的总称,这类程序种类繁多,不同的应用领域、不同的部门需要编制不同的应用程序。为了减少重复劳动,提高程序的重用率,应用软件也在逐步商品化,形成各种软件包。例如,各种科学计算机软件包、事务管理软件包、辅助教学软件包等等,用户可根据自身的需要选购,十分方便。

有时,系统软件和应用软件不能截然分开,如各种标准库,既可以看成是应用软件,也可以看成计算机厂家提供的系统软件。对于一个使用计算机的工程技术人员来说,熟悉各种系统软件的目的是为了更好地发挥计算机的功能,更有效地从更高水平上开发应用软件,编制应用程序。

1.2 计算机语言和语言处理程序

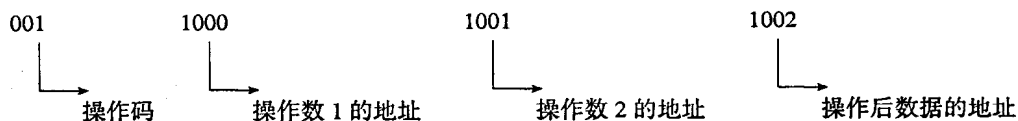
1.2.1 计算机语言概述

从1946年以来的半个世纪中,计算机硬件经历了四个时期,如电子管、晶体管、集成电路和大规模集成电路到超大规模集成电路的换代。现代计算机正向着巨型、微型、并行、分布、网络化和智能化几个方面发展,其相应的软件逐步丰富和完善。随着计算机硬件的发展,计算机语言经历了机器语言,汇编语言,高级程序设计语言,面向程序设计语言等过程。

1) 机器语言

在计算机出现的初期,人们直接使用机器语言编写程序,这种的程序叫机器语言程序。它要求程序设计人员相当熟悉计算机的所有细节,例如计算机的指令系统、存储容量、寄存器的类型与个数等等。这样做工作量大,非常容易出错并且不易修改,并且每个计算机器的指令各不相同,其程序依赖于特定的机器类型,局限性很大,移植性甚差。

所谓机器语言就是指机器的指令系统,它包含了每台计算机能够执行的基本操作,这些操作是由指令形式(一个字节或多个字节的二进制代码形式)来书写。它是裸机惟一直接能识别和执行的指令(或语言)。例如:



表示将地址 1000,1001 中的内容相加的和存放在 1002 地址之中。

2) 汇编语言

为了摆脱机器语言编程的困难,使计算机成为广大工程技术人员都能使用的工具,同时也为了减轻程序设计人员编程和调试机器语言程序的繁重的劳动,提高程序设计的效率,出现了以符号指令来替代机器指令的编程办法。用符号语言编写的程序称为符号程序。它是使用以英文名称的缩写的指令助记符,例如取数用 LD,加法用 ADD 等,它比单纯用“0”和“1”的代码串容易记忆。在此基础上进一步扩充就成为汇编语言,用汇编语言编制程序比用机器指令代码方便得多,不仅易于检查和修改错误,而且指令、原始数据和结果的存放单元可由机器根据定位为指令自动分配。但是符号指令是不能被计算机直接识别和执行,必须将它翻译成机器语言后才能执行。这个翻译工作是由汇编程序来完成的。

程序设计人员在使用汇编语言编程时,仍必须熟悉机器的硬件结构,因此程序设计仍然很繁琐和低效。但是正因为依赖于硬件,熟练的程序员可结合硬件的特点设计出高质量的程序,所以直至今日,汇编语言仍起着重要的作用,常用于对时间和空间效率要求较高的计算机系统核心程序和实时控程序等。

3) 高级语言(面向过程的程序设计语言)

高级语言是由表达各种意义的“词”和“数学公式”,按照一定语法规则组成的面向过程的语言。用高级语言进行程序设计比较接近人的习惯,编出的程序与具体的机器指令无关,可以独立于机器,通用性较强如同汇编语言一样。用高级语言编写的源程序机器是不能直接识别

和执行,必须翻译成机器指令后才能在计算机上执行。

目前,世界上有许多程序设计语言,较流行的有好几十种,例如用于科学与工程计算的 FORTRAN 语言,会话式的 BASIC 语言,教学和系统设计的 PASCAL 语言,数据处理的 COBOL 语言,人工智能中使用的表处理 LISP 语言,逻辑程序设计的 PROLOG 语言、大型通用的 PL/1 语言,一种结构化、模块化、可编译的通用 C 程序设计语言,支持数据抽象类型概念的 Ada 语言等。

4) 面向对象的程序设计语言

与过去面向过程程序设计比较,面向对象程序设计语言的最大特点在于面向的是对象而不是面向过程。所谓对象是现实世界中的实体,例如桌子、电视机、张三、硬件、规律等等。把具有共同行为和特征的实体的集合,可以被归纳成一类,因此每个对象都是属于某一个类的对象,例如人是一个类,而每一个具体的人就是人类中的一个对象。面向对象的程序设计是程序设计中的一种新思想,该思想认为程序是相互联系的离散对象的集合,面向对象程序设计语言是支持这种新思想的程序设计语言,它主要具有封装性、继承性和多态性等特点。

用面向对象的程序设计语言编的程序也不能直接为机器所识别和执行,必须翻译成机器指令后才能在机器上运行。面向对象程序设计语言分为两类,一类是在面向过程程序设计语言的基础上增加了对象的成分构成混合性面向对象的程序设计语言,例如 C++、VC++、Borland C++、Delphi 等等;另一类是按照面向对象设计思想而构成的面向对象程序设计语言称为纯的面向对象设计语言,如 Smalltalk、Eliffel 等。

1.2.2 计算机语言处理程序概述

用计算机语言编写的程序除机器语言程序外,全部程序都不能为机器直接识别和执行,必须经过各种语言的处理程序(或翻译器)处理转换成机器语言程序才能为机器所识别和执行。由翻译所得的结果称为目标程序,目标程序可以用机器语言表示也可以用汇编语言或其他中间语言表示,语言处理程序提供这种翻译功能。

1) 汇编程序

汇编程序的主要功能是把汇编语言源程序翻译成机器能识别的目标程序,其汇编过程如图 1.2。

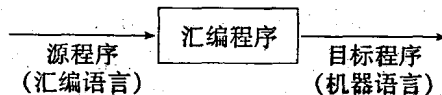


图 1.2

源程序通常是由 ASCII 码表示的符号化的指令串组成。当汇编程序加工源程序时,总是从头到尾一个符号接着一个符号地阅读,称为扫描源程序。从头到尾扫描一次源程序为扫描一遍。具体地说,汇编程序必须完成几个方面任务:

- ①处理语句中的操作符,生成与相应的机器码,产生机器指令;
- ②处理语句中的标号和名字,并代之以具体的单元地址;
- ③处理伪指令。

为了完成汇编程序的翻译工作,需要以下表格:

- 机器码操作表(MOT) 用以确定指令的长度和助记符转换成机器码;
- 伪指令操作表(POT) 用以查找伪指令对应的操作;

- 地址计数器(LC) 用以追踪和确定指令地址;
- 符号表(SYMT) 用以记录各标号、名字。

2) 编译程序

用高级语言编写的源程序可以通过编译的途径将其翻译成语义上等价的目标程序。目标程序可以用机器语言、汇编语言或某种中间语言表示,如果目标程序是用汇编语言表示的,则它必须再经汇编程序汇编成机器语言程序;如果目标程序是用某种中间语言表示的,则该目标程序可以进行由下面讨论的解释器解释执行或再经编译后执行。例如,PASCAL P_编译器所产生的中间代码叫P_代码,这种P_代码程序可以解释执行也可以再编译后执行。

编译途径一般可分为三个阶段,即编译阶段、连接装配阶段和运行阶段。在编译阶段,由编译程序扫描源程序并将其翻译成目标程序。当编译程序的输出是待装配的目标程序模块时,需经过连接装配阶段,即由连接装配程序把目标程序(模块)以及必须的运行子程序连接起来形成可执行的机器代码(图 1.3)。

一般将编译过程分成五个阶段:词法分析、语法分析、语义分析和中间代码生成、代码优化、目标代码生成(图 1.4)。

此外,编译过程中还有两部分工作:一是建表和查表工作;二是进行出错处理。图 1.5 是编译程序的一种典型的逻辑结构。

(1) 词法分析

词法分析程序又称扫描器,它对源程序进行自左向右扫描,识别出一个一个单词(保留字、标识符、常数、运算符和界限符等),并将它们转换成相应的机内表示,从而形成相应的单词符号串(也报告词法错误)。

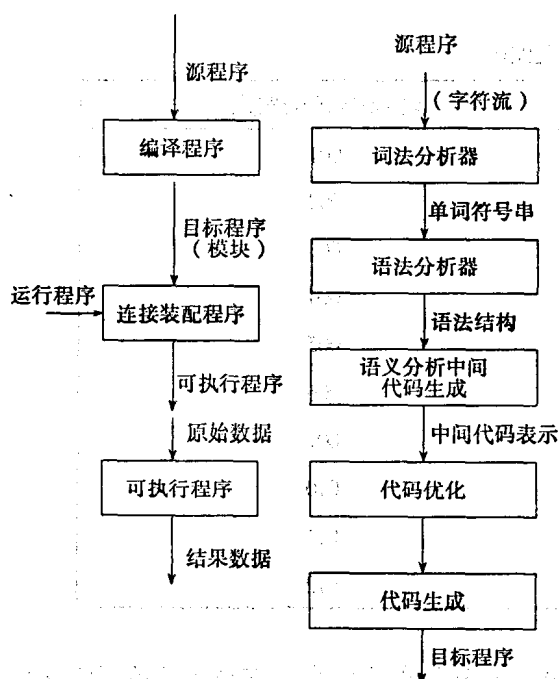


图 1.3 编译途径

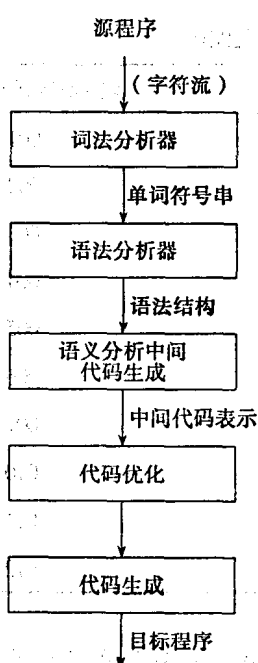


图 1.4 一般编译过程

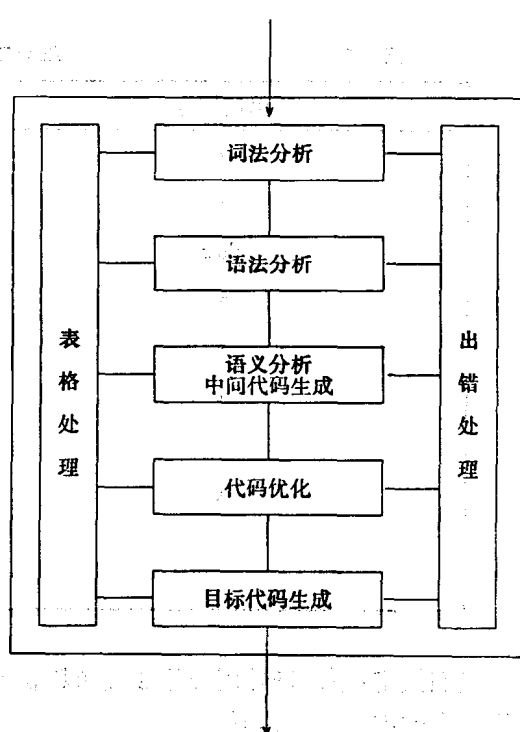


图 1.5 编译程序逻辑结构

请看下面一段简单的 C 源程序：

```
{
    float  r,h,s;
    s=2*3.1416 * r*(h+r)
}
```

这是一个由 33 个字符组成的字符串({和 float, float 和 r 之间各有一个空格)：

```
{float  r,h,s;s =2*3.1416 * r *(h+r);}
```

词法分析程序首先从左到右扫描此字符串,并根据词法规则,将其分解成一个个具有独立语法意义的单词符号(也称“单词”或“符号”),如保留字 float,标识符(r,h,s),常数(2,3.1416),运算符和分界符(, ; = + () *)等。这些单词是组成语句的基本符号。经词法分析,将上述源程序转换成由 22 个单词符号组成的单词符号串,它们通常由特定的内部形式表示。为此,我们可将单词分类,如分为表 1.1 中的四种类别,每种类别由一个整数表示。每个单词也给定一个内部编码(内部值),如表 1.2 所示。

表 1.1 单词类别及编码

单词类别	类别码
保留字及分界符	0
标识符	1
无符号实数	2
无符号整数	3

表 1.2 单词符号及编码

单词符号	内部编码(八进制)
{	001
}	002
float	003
=	004
*	005
+	006
,	007
(	010
)	011
;	012

我们假定:单词符号用四位 8 进制数表示,第一位表示单词类别,后三位表示单词值,则单词符号串的内部形式如下：

0001	0003	1000	0007	1002	0007	1004	0012
{	float	r	,	h	,	S	;
1004	0004	3000	0005	2001	0005	1000	0005
S	=	2	*	3.1416	*	r	*
0010	1002	0006	1000	0011	0002		
(	h	+	R	)	}		

(2) 语法分析

语法分析程序又称分析器,它根据程序设计语言的语法规则,将单词符号序列组合成(识别出)各种语法成分(如表达式、说明语句、分程序、程序等),在组合成语法成分的同时,进行语法检查,即检查语法成分在语法结构上的正确性,这就是语法分析。

例如本例中,经过语法分析,可以识别出“{”是复合语句或分程序的开始符号;“float”是保留字,属于语法成分<类型说明符>;“r, h, s”构成语法成分<标识符表>;<类型说明符><标识符表>构成新的语法成分<类型说明>;由<说明>构成<说明串>。同理,可以识别出“s”是<变量>,”=”是赋值号,“2 * 3.1416 * r * (h + r)”是<表达式>,三者组成<赋值语句>;由<赋值语句>构成<语句>;由<语句>构成<语句串>。最后识别出“{<说明串>;<语句串>|”是一个C语言的分程序。经语法分析可以建立如图1.6的语法树,并证明该源程序在语法结构上是正确的。

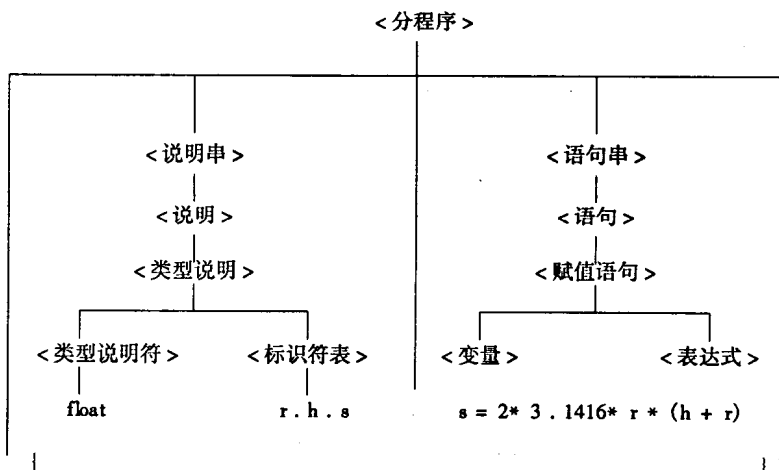


图 1.6 语法树

3.1416 * r * (h + r)”是<表达式>,三者组成<赋值语句>;由<赋值语句>构成<语句>;由<语句>构成<语句串>。最后识别出“{<说明串>;<语句串>|”是一个C语言的分程序。经语法分析可以建立如图1.6的语法树,并证明该源程序在语法结构上是正确的。

(3) 语义分析和中间代码生成

在语法分析基础上,再作语义分析。例如本例中,类型说明语句的语义是:R, H, S 为实型简单变量;赋值语句的语义是:计算机赋值号右边表达式的值,将它送到赋值号左边的变量所指定的内存单元中。计算机语言的语义是要求机器完成的指定操作。为了表达这种语义,并方便代码优化和生成目标程序,通常使用一种中间语言(它介于源程序语言和机器语言之间)来表示。语义分析的任务:一是进行语义正确性检查,二是进行语义处理并生成相应的中间代码。例如,被说明的标识符在同一分程序中被说明了两次,就是语义错误:

```

{
    int a;
    float a;
    :
}
  
```

又如当赋值语句左边变量是整型,而右边表达式是实型,也是一种语义错误。语义处理的