

《计算机应用软件基础》教学系列丛书

PASCAL 语言

· 数据结构

· 算法

主编 王海源
仇芒仙
主审 边善裕

成都科技大学出版社

《计算机应用软件基础》教学系列丛书

PASCAL 语言 · 数据结构 · 算法

主编 王海源 仇芒仙

主审 边善裕

成都科技大学出版社

(川)新登字 015 号

内 容 简 介

本书的第一至第六章介绍 PASCAL 语言程序设计,其中包括 PASCAL 基本概念:PASCAL 的程序结构、基本语法单位、数据类型、指针、动态数据、程序流程控制、函数、过程等内容;第七至第十章介绍数据结构及其 PASCAL 程序的实现,其中包括:线性表、多维数组、树、图的数据结构及其算法;第十一章及第十二章介绍数值计算和非数值计算的常用算法和编程技巧;第十三章是上海市高校等级考试的有关试题分析。每章后均附有习题。

本书可作为计算机专科有关课程的教材,也可作为上海市非计算机专业学生参加二级考试的教材及参加三级考试的程序设计和数据结构内容的教材,还可作为各类工程技术人员学习计算机,提高编程能力的参考书。

PASCAL 语言·数据结构·算法

主编 王海源 仇芒仙

主审 边善裕

成都科技大学出版社出版发行

四川省新华书店 经销

无锡市文化印刷厂印刷

开本:787×1092 1/16 印张:18

1995 年 11 月第 1 版 1995 年 11 月第 1 次印刷

印数 1—2000 字数:430 千字

ISBN 7-5616-3158-8/TP·148

定价:19.00 元

《计算机应用软件基础》教学系列丛书

代序

《计算机应用软件基础》教学系列丛书(简称“教学系列丛书”)是上海市高等教育局“计算机应用软件基础课程建设”课题的成果之一。该系列丛书是大专院校计算机专业专科和本科有关课程的教材和教学参考书;是非计算机专业学生学习计算机,参加“上海市高校非计算机专业应用知识和应用能力等级考试”的教材和教学参考书;是《九十年代上海紧缺人才培训工程》的参考丛书,对参加“上海市计算机应用能力考核”的广大市民用作辅助读本和复习迎考的参考书;也是各类工程技术人员学习计算机应用知识与能力的自学丛书。

“教学系列丛书”内容丰富、材料组织合理、结构严谨、概念正确;每本教材既重视科学理论又重视实践操作和实际应用能力的培养;条理清楚、循序渐进、通俗易懂,便于自学;并且在各本教材之间具有紧密的内在联系,选择其中几本就可组成一套附合某一特定要求的教材。这是编著者长期教学实践活动的经验结晶,确是一套值得推荐的计算机软件应用基础教学方面的好教材和参考书。

第一批“教学系列丛书”有:《计算机应用基础教程》、《计算机应用简明教程》、《实用数据库技术 FoxBASE+ 及其应用》、《数据库技术 FoxPro 和数据库图形 FoxGraph 》、《dBASE III 简明教程》、《汉字 LOTUS 1-2-3 及其应用》、《PASCAL 语言·数据结构·算法》、《C 语言·数据结构·算法》和《管理信息系统》等。

各本教材的特点是:《计算机应用基础教程》适用于作计算机专业本科和专科的“计算机导论”课程的教材,以及非计算机专业理工科学生学习计算机,参加“上海市高校非计算机专业应用知识和应用能力等级考试”的教材和参考书;《计算机应用简明教程》和《汉字 LOTUS 1-2-3 及其应用》适用于管理类专业有关课程的教材和非计算机专业文科学生学习计算机,参加“上海市高校非计算机专业应用知识和应用能力等级考试”的教材和参考书;《实用数据库技术 FoxBASE+ 及其应用》和《数据库技术 FoxPro 和数据库图形 FoxGraph 》可作为计算机专业专科学生有关数据库课程的教材,以及非计算机专业理工科选修课的教材;《PASCAL 语言和数据结构》、《C 语言·算法·数据结构》和《管理信息系统》都可作为计算机专业专科和本科有关课程的教材,也可作为非计算机专业理工科本科学生选修课的教材与参考书;《dBASE III 简明教程》适用于中学教师进修数据库技术课程的教材和中学生课外活动的参考书,也适用于各类中专技校计算机有关课程的教材和教学参考书。

《计算机应用基础教程》、《计算机应用简明教程》、《实用数据库技术 FoxBASE+ 及其应用》和《数据库技术 FoxPro 和数据库图形 FoxGraph 》等都是参加“上海市计算机应用能力考核”初级和中级考试很好的复习和参考书。

《计算机应用软件基础》教学系列丛书 评审委员会

1994 年 5 月

《计算机应用软件基础》教学系列丛书

评审委员会名单

- 顾问：胡启迪 上海市高等教育局副局长(教授)
上海高校非计算机专业学生计算机应用知识与应用能力考试委员会主任
- 施伯乐 复旦大学计算机科学系主任(教授)
上海市计算机应用能力考核专家组组长
- 主任：张吉锋 上海工业大学计算机学院副院长(教授)
全国计算机教育委员会副主任
- 副主任：白英彩 上海交通大学计算机研究院副院长(教授)
上海市计算机应用能力考核专家组成员
- 汪燮华 华东师范大学理工学院副院长(教授)
上海市计算机应用能力考核专家组成员
- 俞时权 上海大学工学院计算机系主任(教授)
上海市计算机应用能力考核专家组成员
- 陈华生 南京大学计算中心主任(教授)
江苏省普通高校计算机等级考试中心主任
- 朱关铭 上海科技大学计算机系(教授)
上海高校非计算机专业学生计算机应用知识与应用能力考试委员会委员
- 许宝元 上海市高等教育局教学处处长(副教授) 评估办主任
- 委员：曾广周 山东工业大学计算机系主任(教授)
- 吕家俊 山东师范大学计算机系主任(教授)
- 李怀斌 上海医科大学计算中心(教授) 中国系统仿真学会常务理事
- 王修才 上海师范大学计算机系副主任(教授)
- 蔡绍稷 南京师范大学计算中心主任(副教授)
江苏省普通高校计算机等级考试中心副主任
- 钱维民 福州大学计算机系主任(副教授)
- 乔需荣 华东理工大学基础教学中心主任(副教授)
上海高校非计算机专业学生计算机应用知识与应用能力考试委员会委员
- 刘淦澄 华东师范大学计算机系主任(高级工程师)
- 瞿彭志 上海大学国际商学院计算机系主任(副教授)
- 陆嘉宝 上海建筑材料学院自动控制与计算机系主任(副教授)
- 杭必政 同济大学计算机系(副教授) 全国计算机基础教学课委会委员
- 余宗礼 上海建筑材料工业学院教务处处长(副教授)
- 梁树基 上海师范大学教务处处长(副教授)

前 言

本教材包括 PASCAL 语言程序设计、数据结构基础、程序设计方法和技巧三大部分,适用于非计算机专业理科或文科管理类专业,以及计算机专业专科作为软件基础的教材。由于本书的基本立足点在于提高程序设计的能力,为此编排了较多的程序实例,因此,对已初步学过 PASCAL 程序设计的学生,仍不失为一本实用的教材。出于切实提高学员的程序设计综合能力这一基本考虑,我们瞄准了上海市普通高校计算机等级考这一具体目标,并以它的大纲要求(包括知识范畴和能力要求),它的命题形式,它的试题分析作为本书的编写基点和重要内容。因此,本教材对上海市非计算机专业计算机等级考核和能力考核(二级和三级)的参加者(包括全国软件应用人员水平考的应试者)都有良好的借鉴作用。由于书中的知识与技巧注重简练实用,并配以大量实例,所以也特别适合于自学者使用。

基于精简篇幅和扩大内容容量两方面的考虑,我们把内容重心放在程序设计方法、规律和技巧上,因此,在第二到第六章关于语言的描述方面相应作了紧缩。但考虑到初学者的情况,我们仍安排了一个完整的 PASCAL 语言教学体系。讲述中以每种文法结构的程序特征、程序功能、应用方式的实例为主。因此,即使是已有 PASCAL 语言初步基础的学习者,我们仍不主张跳过这些章节,而建议用较快的速度学习这些章节,并学完其中的程序例子。把它们作为小结、复习和补充。对于初学者,则须记:语言的规范形式是程序设计之必要前提,这点马虎不得,应常去翻阅书后的语法图,并用它和程序实例的形式加以对照。

不管哪一类学习者,在第七到第十章数据结构基础知识学习的过程中都应当把注意点集中到程序设计技巧中,利用已掌握的语言基础,去阅读、分析程序数据的构造和算法的形成,切忌孤立、静止的去了解数据形式。有数据结构就有定义在结构上的程序,数据结构的基本概念是不变的,然而在程序中出现的形式却是灵活多变的。因此,学习数据结构的目的是掌握数据构成的规律和对它处理的规律,以规范的方式设计程序。

第十一到第十三章是程序设计的深化:介绍了常用的数值算法、非数值算法和其它程序设计方法;阐述了程序分析的技巧;提供了高校计算机等级考的实例和解答(程序设计部分)。这对于各类学习者都是有效的。

附录中提供了 PASCAL 语法图, Turbo PASCAL 的基本使用方法,以及上海市普通高校计算机等级考(二级考)的大纲,供学习中参考。

本书的结构及涉及内容,是在教学实践中几经探索和选择形成的,每章后的习题可供练习或上机之用。凡注有 * 号的习题(每章三个左右)建议上机实习选用。

本书的第一至第四章由仇芒仙编写,第五章至第十三章由王海源编写,并由王海源对全书进行了校核。由于受水平、阅历之限制,文中可能会有不足和错误,敬请各位读者同仁斧正。

作者 1995.1

目 录

第一章 导 言	(1)	
§ 1.1 程序设计语言和程序设计能力	(1)	
§ 1.2 计算机等级考在程序设计能力上的要求	(2)	
§ 1.3 迅速有效地提高程序设计水平	(3)	
第二章 PASCAL 基本概念	(5)	
§ 2.1 PASCAL 的程序结构和基本语法单位	(5)	
2.1.1 PASCAL 程序的总体结构	2.1.2 PASCAL 基本字符集	
2.1.3 PASCAL 基本词	2.1.4 语法描述工具	
§ 2.2 标准类型常量和变量	(10)	
2.2.1 整数类型	2.2.2 实数类型	2.2.3 字符类型
2.2.4 布尔类型	2.2.5 变量	
§ 2.3 表达式	(17)	
2.3.1 类型和相容性	2.3.2 使用表达式的几个注意点	
§ 2.4 简单语句及其程序作用	(20)	
2.4.1 赋值语句	2.4.2 输入语句	2.4.3 输出语句
小 结	(25)	
习 题 二	(25)	
第三章 流程控制	(27)	
§ 3.1 分支结构	(27)	
3.1.1 条件语句(IF 语句)	3.1.2 分情况语句(CASE 语句)	
§ 3.2 循环结构	(33)	
3.2.1 条件判别型循环控制语句(WHILE 语句和 REPEAT 语句)		
3.2.2 计数型循环语句(FOR 语句)	3.2.3 循环语句各种形式的对比	
§ 3.3 分支语句和循环语句的使用要点	(39)	
§ 3.4 分支程序设计方法	(41)	
3.4.1 单一划分	3.4.2 重复划分	
§ 3.5 循环程序设计方法	(44)	
3.5.1 简单循环	3.5.2 复杂多重循环的程序设计方法	
§ 3.6 转跳语句	(52)	
小 结	(54)	
习 题 三	(54)	
第四章 类型及其程序作用	(56)	
§ 4.1 用户定义的简单类型	(56)	
4.1.1 枚举类型	4.1.2 子界类型	
§ 4.2 结构类型	(63)	
4.2.1 数组类型	4.2.2 集合类型	

4.2.3 记录类型	4.2.4 文件类型	
小 结		(84)
习 题 四		(84)
第五章 指针和动态结构		(87)
§ 5.1 指针和动态存区		(87)
5.1.1 指针的定义	5.1.2 指针的操作	
§ 5.2 链表		(89)
5.2.1 链表的构成	5.2.2 链表的建立和输出	
5.2.3 链表结构的插入、删除和寻找		
§ 5.3 动态结构程序设计的基本方法		(94)
§ 5.4 动态结构的各种形式		(99)
小 结		(99)
习 题 五		(99)
第六章 函数、过程和程序结构		(101)
§ 6.1 函数、过程的语法规则和基本作用		(101)
6.1.1 函数的定义和简例	6.1.2 过程的定义和简例	
§ 6.2 程序设计的顶向下规划方法		(104)
§ 6.3 递归		(106)
6.3.1 递推式和递归子程序	6.3.2 递归子程序的基本思路——分治法	
小 结		(111)
习 题 六		(112)
第七章 线性表		(115)
§ 7.1 数据结构的基本概念和术语		(115)
§ 7.2 线性表及其存储方式		(115)
§ 7.3 顺序表及其应用		(116)
§ 7.4 线性表的链表结构和静态链表的基本算法		(118)
§ 7.5 链表的实例		(121)
7.5.1 静态链表例	7.5.2 动态链表例	
小 结		(125)
习 题 七		(125)
第八章 线性表的派生结构		(127)
§ 8.1 栈		(127)
8.1.1 栈的定义和基本操作	8.1.2 栈基本算法的实现	
8.1.3 顺序栈的伸展方向和空间共享	8.1.4 栈的应用	
§ 8.2 队列		(132)
8.2.1 队列的定义和基本操作	8.2.2 队列基本算法的实现	
8.2.3 队列的应用和模拟		
§ 8.3 串		(137)
8.3.1 串的存储结构	8.3.2 串的基本算法的实现	

§ 8.4 多维数组	(141)
8.4.1 多维数组的一维数组表示方法	8.4.2 特殊矩阵的压缩存储
8.4.3 稀疏矩阵	8.4.4 数组的链式存储结构
小 结	(150)
习 题 八	(151)
第九章 树和二叉树	(152)
§ 9.1 树的定义、术语和二叉树的性质	(152)
9.1.1 树的定义和表达方式	9.1.2 树的基本术语
9.1.3 二叉树的定义及其性质	9.1.4 树和二叉树的存储结构
§ 9.2 二叉树的基本算法	(158)
9.2.1 二叉树的遍历	9.2.2 二叉树的生成
9.2.3 结点的前驱、后继和二叉树的穿线形式	
§ 9.3 树和二叉树的形式转换	(164)
§ 9.4 树和二叉树的实例和应用	(166)
9.4.1 集合的等价类划分	9.4.2 堆垒和用它进行的排序
9.4.3 表达式的求值	9.4.4 分类树, 编码和判定树
小 结	(175)
习 题 九	(175)
第十章 图的数据结构	(177)
§ 10.1 图的基本术语	(177)
§ 10.2 图的存储结构	(178)
10.2.1 邻结矩阵	10.2.2 邻接表
10.2.3 有向图的十字链表	10.2.4 无向图的邻结多重表
§ 10.3 图的遍历	(182)
10.3.1 深度优先搜索 (Depth-first search)	10.3.2 广度优先搜索
§ 10.4 图结构的应用	(184)
10.4.1 图的最小生成树	10.4.2 拓扑排序
小 结	(188)
习 题 十	(188)
第十一章 基本算法和常用的程序方法	(189)
§ 11.1 数值计算的几种基本方法	(189)
11.1.1 方程的基本解法——迭代	11.1.2 级数的前后项递推方法
11.1.3 数据类型的构造法	11.1.4 多项式算法
11.1.5 数值积分方法	
§ 11.2 非数值计算型程序的常用算法	(194)
11.2.1 基于关键字比较的基本查找算法	11.2.2 基于关键字定位的查找算法
11.2.3 排序算法	
§ 11.3 非数值计算的常用方法的技巧	
11.3.1 穷举搜索法	11.3.2 递归法

11.3.3 回溯法	11.3.4 分治法	
小 结		(215)
习 题 十一		(215)
第十二章 程序分析技巧和实例		(217)
§ 12.1 程序分析的基本特点		(217)
§ 12.2 怎样分析一个程序填充问题		(218)
§ 12.3 程序分析和填充的实例		(220)
§ 12.4 流程图分析		(227)
12.4.1 程序流程图的基本形式	12.4.2 流程图分析	
小 结		(231)
习 题 十二		(231)
第十三章 计算机等级考试试题分析		(239)
§ 13.1 1992 年上海普通高校非计算机专业学生计算机等级考 (二级)PASCAL 试卷及分析解答		(239)
§ 13.2 1993 年(春)上海普通高校非计算机专业学生计算机 等级考(二级)PASCAL 试卷及分析解答		(247)
§ 13.3 1993 年(秋)上海普通高校非计算机专业学生计算机等级考 (二级)PASCAL 试卷及分析解答		(256)
§ 13.4 1993 年(秋)上海普通高校非计算机专业学生计算机等级考(三级)中 关于程序设计的试题及分析解答		(265)
附录		(269)
附录 A PASCAL 语法图		(269)
附录 B 二级考试大纲(上海普通高校非计算机专业学生计算机等级考)		(273)
附录 C PASCAL 语言考试大纲(二级考附件)		(274)
附录 D Turbo PASCAL 的基本使用方法		(275)

第一章 导 言

学习计算机软件基础课程,包括各种程序语言 and 基础理论课程,基本目的在于提高程序设计能力。而要达到这个基本目的,程序语言的学习就不能只把目光停留在语言文法的规定上,程序基础理论的学习也不能只注意静态的数据模式。应该把重点放在程序上,即由浅入深、由低而高地掌握程序的读、写技能,这已是很多软件教学实践者对计算机软件方面课程的共识。本书以 PASCAL 程序设计及其后继课程数据结构和基本算法为背景,以提高计算机程序设计能力为基本瞄准点,力求探索软件教学中的高效率的新路。这就是本书的主旨。对于各专业修学计算机程序设计的学生而言,一个具体的目标,就是在相对比较紧凑的学习时间中,完满地达到高校计算机等级考核的水平要求^[注],进而达到全国软件应用人员水平考试的能力要求。

不少计算机专科或其它专业的文理科学生,参加了高校计算机应用能力的二(三)级考试之后,会对自己的考核结果颇感意外,而这种“意外”又几乎无例外地集中在程序设计(包括程序填充和程序阅读)部分。概括为一个字,就是“难”。确实如此,哪怕经过很认真的程序设计语言的学习,也会有很多人在检验水平的考核中体验这个“难”字。现在,我们就从这个“难”字入手,从“难”字的内涵,“难”的原因,“难”的克服方法诸方面进行探讨,引出一条化难为易的路径,并把本教程沿这条路径编排,力求在这条路径上迅速有效地提高学习者的程序能力。

§ 1.1 程序设计语言和程序设计能力

目前,计算机技术已介入到各个科技领域,在高校中几乎找不到一个专业是和计算机绝缘的,也找不到一个专业是完全不设置计算机课程的。这也是计算机应用能力等级考面对的形势。参加考核的学生,都学过程序设计。学过了,学完了,为什么考不好?“难”字正是由此而生。

其实,分析那些学过、学完甚至学好的程序设计课程,一般都还是偏重于程序语言的课程。这些课程往往是在程序的语言工具方面对初学者进行严格的语法规义规范教学,以完成学习者从不懂到初知的入门。在严格意义上说,这类程序设计语言课还只是一种初等的程序设计课程。具体来说,它无法在介绍语言规范的同时,形成一种十全十美的教材组织结构,能在逐项介绍其语言分支时,比较完整地介绍其程序技巧。例如在 PASCAL 语言中,在循环语句之前讲数组,则难于阐明数组的重要作用,而把循环语句放在数组前面讲解又会显得过于单调;把结构类型放在流程语句之前,难以表达它们对编程的重要作用,显得死板,但反之把流程语句先于类型章节讲解,又会使流程语句显得枯燥;同是结构型数据,数组先于记录讲解,会在一定程度上掩饰数组的构造功能,而把次序反过来,则又会弱化记录的作用;在介绍基本类型时,无法对实数类型在类型构造和语句使用时所受限制完整地介绍,然而语言描述的布局又不会允许把实数这一标准类型放到后面章节……

凡此种种,造成了程序语言教程结构安排上的伤脑筋问题:一方面,在很多新的语法对象出现时,不得不隐没其某些本质特征和重要作用;另一方面,在后续章节出现时,却往往又得回过头来讨论前面被暂时隐没的内容。这样,以介绍程序语言为主旨的教程中在不少方面的实例往

往显得过于简单。

这些就造成了要使程序语言和程序设计能力二者达到理想的同步所面临的困难。所以,以语言教学为主旨的程序语言课程和以程序能力提高为目的的程序设计课程的区别,是显而易见了。就一般情况而言,PASCAL 程序语言的修读完成,并不代表一个学生 PASCAL 程序设计课程的完成。就象会写字认字的人未必会写好文章一样,如果没有一定的算法和数据结构课程作为后盾,则知识上和能力上的“断层”的产生是无法避免的。所以说:学习了程序设计,学好了程序设计语言而又考不好程序设计,正是程序能力断层的具体表现。“难”字产生的原因之一,正是源于“程序语言 $\neq$ 程序设计”这一简单的事实。

§ 1.2 计算机等级考在程序设计能力上的要求

计算机等级考在程序设计上的考核,是对编程能力较为客观的检验。从考试大纲(见附录)中是很难精确体验到如何才算达到标准的,这是因为大纲可以精确标定程序的范围,却无法精确标定能力的深度。就象一棵大树,由树干、树叶可精确表达树冠的广度,但却无法精确感知根系的深度。“难”字的第二个来源,就在于难以估计考核的深度要求。无论是等级考也好,其它形式的水平考也好,大纲都将带给你一个极为乐观概貌:它会列举语言中的各种你所熟悉的成份,这些在程序设计(语言)课中你所学会了的。然而你却大有可能忽略在廖廖数语中包含着的比较沉重的压力。比如:“要有一定的编程能力”,这“一定”定在何处?其内涵用什么天平都很难衡量准足。“难”字的第二个来源盖出于此。

不少学生发现程序能力的考核要求高于自己的估计时,深悔自己准备不足。那么,如何感知考核的实际要求的深度呢?不妨采用如下两种方法:

第一种,是借助已有试题和模拟试题。这至少反映了命题者的命题规范、形式和风格,反映了命题者对考核深度的理解。因为现已有了好几次考试的实例,所以这还是可行的。

第二种,则是以一个较高的标准权作心目中的考核深度。它的要求是:能有十分的信心和准确度,完成程序设计的一切要求。当然,实际达到这个水平是不易的,然而,当我们用尽可能合理的方法,以尽可能高的学习效率向这个目标迈进时,最终我们将达到一个尽可能高的能力水平,而事实上,这也正是考试制度的基本目的。就实际情况而言,试题间本身就有难易之分,有时甚至是比较明显的,而且,不同的人对于同一组题的理解,也因主观因素的影响而相径庭。所以,第一种方法只能作为一种参考方法,第二种方法才是基本的。

这种考核中“难”点,反映在实际程序能力的自我测评和感知方面的“难”。正因为知识面上的“懂与不懂”,与知识深度和能力上的“精与不精”、“能与不能”,是完全不同类型的评价标准;而后者则是困难得多的评价标准,所以,考核的“难”,寻本究源,还在于程序设计能力的水平和程度把握起来比较困难。其实,程序能力考核的目的,正是为了在事实上造成这种把握程度的标尺。这也正是本书在以提高程序设计能力为主旨的前提下,要以计算机等级考和软件水平考的要求为基点和目标的主要原因。

“难”字的第三个来源,则是命题内容的灵活性和命题方式的特殊性。命题的灵活性是命题深度的一个集中反映:不仅有你所并不熟悉的程序处理要求,并且有和你的编程习惯不相一致的程序处理方法。通过程序填充这一考试方法固定了这些要求和处理方法,不允许回避,不能够改换。

这就要求我们有较强的阅读他人的算法说明和程序的能力。不仅要能按问题的要求,按自己的技巧和风格编出程序来,更重要的是要能按他人的思路读懂原先你并不熟悉的程序。这种能力实际反映了一种对程序的学习能力,是编程能力的一种重要反映。这和等级考核制度的基本目的是相容的。

§ 1.3 迅速有效地提高程序设计水平

从前面的分析,我们知道:

——应对“程序语言≠程序设计”有十分清醒的认识。程序设计语言课程后,必须经过一定量的程序设计的严格训练;

——程序能力的深度要求是难以量化的,因此,必须把应考的基本要求定在更高一点的水准上;

——对于程序的阅读能力,理解能力有更高的要求,以顺利地处理各种灵活多变的命题内容和特定的命题形式。

怎样迅速有效地提高程序设计水平,这也是每个学生关心的问题,也是本教程要探索和解决的问题。程序设计能力包括:对于语言工具准确的理解;迅速流畅地编制程序;敏捷精确地阅读和理解程序;以及上述三种能力的有机结合。因此,在教材中把程序语言的基本内容,和作为它的后继内容的数据结构和程序设计方法三方面内容分步骤地有机地组合起来。这种组合兼顾了前后的呼应,避免了重复和冗余;在风格上以求能力的提高为基本目的,取得了风格上的统一;和能力考核相接轨,例题分析带有规范性和客观性。凡此种种,目的在于,在“程序设计”这一课程中以最短的学时要求、最高的教学效率,完成由程序设计语言、数据结构、程序设计方法等多门课程的要求。本教程是以如下三个基本步骤为主线而编排的,各步骤的要点如下:

第一步骤: PASCAL 语言及其基本程序方法。在本书第二章到第六章以程序语言的各要素的讲述为主线,就整个语言的各种成份,提供一整套程序实例。考虑到初学者的实际情况,语言基础仍是重要内容,但基本立足点已是语言要素的程序作用,是以它们来介绍程序设计方法。因此程序将在一定程序上突破现有的语言知识的制约,出现“超前”语法现象。这样,程序设计的教学就有了更为灵活的延伸空间,并能实现语句结构各部分知识间的沟通。为了防止和减少“超前文法”现象所产生的负面影响,在附录中选编了 PASCAL 语法总图,供随时参阅,以尽可能杜绝文法错误。

第二步骤: 数据结构基础知识。这是本书第七章到第十章所包含的内容。这是按等级考试的大纲限定的广度进行的介绍和分析,同时,选材也是以简洁、实用并有一定的深度为原则。仍将立足于程序阅读、分析和设计能力的提高,提供一整套程序实例。这第二阶梯使学习者在掌握数据结构知识的同时,程序能力得到锻炼和提高。

第三步骤: 算法和程序的综合训练。第十一章及以后的各章就是为此而设的。将介绍计算机上的数值或非数值的常用算法,并再推出一整套程序设计的综合实例进行分析和探讨。这些实例主要取材于计算机能力等级考,软件应用人员水平考等一些有质量的考核试题,或取自有相应深度的程序设计实例,并以实例阐明程序填充和程序分析题型的处理方法。

这三个步骤,虽然在任务上存在差别,但方法上是接近的,目标上一致的:使学习者在已有基础上获取强化编程能力的学习环境,循序渐进地掌握一批较为定型的算法模型和程序设计

方法。这样,学习者的程序能力可望有一个扎实的提高。所提供的一批程序实例,通过三个步骤——或是说三个“阶梯”——的积累,会转化为一种能力,成为你解决程序问题的坚定支持者:给你触类旁通的灵感;给你进行类比和引用的基本素材;给你以创造性思维的有力工具。因此,这三个步骤中提供的程序实例是必须消化、熟记,有的甚至要适度地背出。这样就会使你逐渐具备更进一步的分析、综合能力。

特别要指出的是:在学习过程中应尽一切可能进行上机实践。程序设计课程中,上机实习是贯穿于全程的特别重要的环节,它被用以加深对算法和数据结构的理解,程序设计中的种种问题和疑点,也可以通过高试来辨明是非、真伪、优劣。这些都是极有价值的。

[注] 本文中的“高校计算机等级考核”,系指:上海普通高校非计算机专业学生计算机等级考试。

第二章 PASCAL 基本概念

PASCAL 是一种通用的程序设计语言,语言精练,结构严谨,用途广泛。尤其值得指出的是它有杰出的算法描述能力,这使 PASCAL 程序对其它各种程序语言而言,具有更为普遍的意义。这就是各个软件科目中极常用这个语言作为描述算法的工具之基本原因。本章中,我们将全面对该语言基本要素进行一个介绍,同时,在程序设计的角度上对 PASCAL 的基本成份的特点和作用加以说明。

§ 2.1 PASCAL 的程序结构和基本语法单位

2.1.1 PASCAL 程序的总体结构

整个程序由程序首部和作为程序体的分程序构成。首部给出程序的名字,并列出它的参数,这是程序与外界交换信息的窗口,由文件变量名组成;分程序则由说明部分和语句组成;说明部分包含着标号说明、常量说明、类型说明、变量说明和过程(函数)说明部分,它们是可选的,但各种说明间的次序不可颠倒,语句部分则是程序实体。

下面所举的程序实例表达了上述程序结构。这个名为 SAMPLE 的程序将输入的一个圆柱体的底面半径(R)、高(H)和比重(DEN)进行合理性判别,在输入数据合理的前提下,计算出这个圆柱体的总重量,并予以输出。让我们把注意点集中于这一程序的总体结构上:

```
PROGRAM sample(input, output);
LABEL 0;
CONST pi=3.14159;
TYPE cylinder=RECORD r, h: real;
                den: real;
END;
VAR c: cylinder;
    v: real;
BEGIN writeln('Input data: r, h, den! ');
      readln(c.r, c.h, c.den);
      IF (c.r<=0) OR (c.h<=0) OR (c.den<=0)
      THEN BEGIN writeln('DATA WRONG! ');
              GOTO 0
            END;
      v:=pi*c.r*c.r*c.h*c.den;
      writeln('RESULT=',v);
      0;
END.
```

这个程序包含了完整的程序结构,第一行由 PROGRAM 引出了一个名为 SAMPLE 的程序的首部,本程序中那些不处在引号之中的大写词汇是语言中具有既定含义的词汇,例如 CONST, TYPE……均为这类成份。接下去就是说明部分,在这一部分中,把上文述及的各项说明内容按严格的先后次序排列出来:

关键字

第一个被说明的是标号 0,它是由 LABEL 引导出的,这是标号说明。它说明程序中有一个用以确定语句位置的标记 0,事实上可以在程序的最后看到由 0;标出的确切位置;

紧随着由 CONST 引出了对常量名称 PI 的定义,实际上确定了 PI 和 3.14159 的直接对应关系;在程序的任何部位出现了对 PI 值的引用,那就取得了 3.14159;

再下面是由 TYPE 所引出的类型名称的定义,这里定义了一个表达圆柱体全部相关信息(名字为 CYLINDER)的数据形式;R 是底面半径,H 是圆柱高度,DEN 则指明其比重,类型定义表达了数据的模式,可以依据这种模式去定义数据实体;

本程序说明部分最后一项是由 VAR 所引出的变量说明,变量说明是说明一些名字,这些名字是参与运算的可变化的数据对象实体,本例中说明了一个类型为 CYLINDER 的圆柱的实体名字 C 和一个用于存放运算结果(总重量)的数据实体 V。

后随的 BEGIN 和程序尾部的 END,如同“语句括号”,括出了其中的全部语句,本例中的语句主要完成了数据的输入,判别,计算和输出等一系列工作。

如何使用说明和语句,这正是程序设计语言课程中的讲述重点,本书的第二到第六章将分阶段地、系统地讲述,而这些基础知识又将会在第七章到第十三章的内容中得到深化和巩固。这里指出如下五点:

第一,正象程序中执行语句的安排,是由任务的需要而定,说明部分的使用也完全是由任务所决定的,其每一种说明可以有,也可以无,如无使用必要,甚至可以完全不使用说明。例如可以编写如下程序,它输出一个由“#”字符所组成的方框:

```
PROGRAM P(OUTPUT);
  BEGIN WRITELN('#####');
        WRITELN('      #');
        WRITELN('      #');
        WRITELN('#####');
  END.
```

这个程序中不存在任何一种说明,但它仍是一个完整的程序。前例中,因无需要,所以未出现函数过程的说明。

第二,各部分说明虽可以按需选用,但一旦选用,则其相对的先后位置则是严格规定的。每项说明,都使说明结果落实到一个名字上;每项说明,可以只说明一个对象,也可以并列地说明多个对象,例如,在 SAMPLE 程序上 VAR 就引出对两个度量的定义;C 和 V。如果有必要,同样也可以在 LABEL 下引出数个标号,在 TYPE 下引出数个类型名称。

第三,在同一程序层次中,被定义的名称具有专一性,就是说,任何一种对同一名字的重复定义将是编译程序无法接受的,将被判为一个语法错误。所以选择被说明的名字时,除了应当考虑它的字面意义外,还必须避免重复定义。

第四,程序首部由括号括起的名字 INPUT 和 OUTPUT,称为程序参数,它表明了程序在执行过程中和外界的哪些对象(主要是对文件)发生信息交流。INPUT 和 OUTPUT 是特殊的名称,表示标准的输入输出设备;当程序执行过程中要动用标准输入设备(键盘)时须指定 INPUT;当执行时会有信息显示于标准输出设备(显示器)时,须指定 OUTPUT。当有其它文件介入程序运行过程时,也需作相应指定,这在第四章 4.2.4 中会详细说明。

第五,每个程序文本都由最后一个句号(.)结尾,若将其遗漏,则也会产生一个语法错误。

就象任何一种语言一样,程序语言的基本形式也是“字、词、句、篇”。在本节 PASCAL 程序

的总体结构中,只是以某种形式初步讨论了 PASCAL 程序的篇章结构,下面将对它的其它要素进行讨论。

2.1.2 PASCAL 基本字符集

能构成 PASCAL 源程序的基本符号被称为基本字符集,它由如下符号组成:

英文字母: A, B, C, ……., X, Y, Z (26 字母, 大小写不分)

数字字符: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

特殊符号: +, -, *, /, =, <, >, (), [], {}, ., : , …….

这些符号,作为单独的字符,一般并没有独立的含义,只有当它们在程序一定地位上,单独地或与其它程序成份相组合地形成一个“词”,这才具备了独立含义,例如字符 9,可以被构成标号,也可以当作为整数 889 的一位,或充当名字 NUMBER9 的一个符号。总之基本字符集中的字符,一般不独立存在,仅当它被认定一个“词”时才具备独立的含义。

2.1.3 PASCAL 基本词

在 2.1.1 所举例子中,我们可以看到很多具有独立意义的语法成份,例如 PROGRAM, REAL, *Cylinder*, 3. 14159, 逗号(,), 乘号(*), ……。这些具有确定意义的独立成份就是 PASCAL 基本词。有单字符词和多字符词两种形式;若按词义来源及其语法特征则可以分为三大类:第一类是系统给以特定含义的词;第二类是程序设计者以各种方式定义的词,也称“用户定义标识符”;第三类是按语言规定的方式书写的常数。

第一类词,实际上是由包含语言定义的特殊符号或特殊字符序列构成:特殊符号(如+, -, *, /, :, =, ……)充当着运算符和分隔符的专门作用,由于它们的意义是不可更改的,因此也被称作专用符号;特定的字母序列则由不可更改其含义的保留字和可以在程序中重新定义的标准字所组成。

例如,在程序中 BEGIN, 它反映了一句复合语句的开始, FOR 则表达了一个循环语句的开始,无论依据什么理由,都不能把这两个词定义成变量,常量或者其它名字。现将标准 PASCAL 的所有保留字分列如下:

AND	DOWNT0	IF	PACKED	TO
ARRAY	ELSE	IN	PROCEDURE	TYPE
BEGIN	END	LABEL	PROGRAM	UNTIL
CASE	FILE	MOD	RECORD	VAR
CONST	FOR	NOT	REPEAT	WHILE
DIV ?	FUNCTION	OF	SET	WITH
DO	GOTO	OR	THEN	NIL

这些保留字反映了某种特定的语法结构。而标准字和保留字的不同之处在于:尽管系统中已对其意义作了规定,但在用户程序中它的含义是可以改变的。标准字主要被用来命名标准类型、标准常数、标准文件及标准函数和过程。

例如,如下的说明语句是错误的:

```
CONST INTEGER = 1234;  
VAR BEGIN:CHAR;
```