TURBO PASCAL

程序设计及其应用

薛伟　胡进　高洁平　编著

中国科学技术大学出版社

Turbo Pascal
程序设计及其应用

薛伟 胡进 高洁平 编著

中国科学技术大学出版社

内 容 简 介

本书以近年来风行全球的微机软件 Turbo Pascal 为蓝本，详细介绍了 Pascal 语言及其程序设计，Turbo Pascal 的使用及其在工程中的应用，并提供了丰富的实例、习题和软件。在语言教学上，本书始终贯穿着结构化程序设计思想，注意体现良好的程序设计风格，并注重编程和实际应用能力的培养。在 Turbo Pascal 的使用方面，本书以 5.5 版本（目前国际上最新版本）为主，详尽地介绍了 Turbo Pascal 基本功能和工程应用，并对已流行的其它版本也作了介绍。

本书可作为计算机专业 Pascal 课程教科书，也可以作为一般大、中专理工科专业"程序设计"课程教科书，还可以作为计算机应用人员的自学读物和参考书。

Turbo Pascal 程序设计及其应用

薛伟 胡迪 蔚伟平 编著

*

中国科学技术大学出版社出版

（安徽省合肥市金寨路 96 号，邮政编码：230026）

合肥工业大学印刷厂印刷

安徽省新华书店发行

*

开本：787×1092 /16 印张：22.75 字数：608 千

1991 年 7 月第 1 版 1991 年 7 月第 1 次印刷

印数 1—8000 册

ISBN7-312-00069-X/TP・25 定价：10.00 元

前　言

Turbo Pascal 是目前微机上最流行的 Pascal 语言系统软件。由于它具有理想的程序开发环境，高效的数值运算功能和多种系统低级调用的手段，因此可用来在微机上编写任何类型、任何规模的计算机应用程序。Turbo Pascal 自 1985 年 3.0 版本问世以来，很快风行全球，受到广大计算机用户的喜爱。由于它的出现，改变了 Pascal 语言只限于教学的历史，并大有取代一些传统的工程计算和数据处理语言的趋势。

Turbo Pascal 的出现，不仅巩固了作为主要教学语言的 Pascal 语言在大学计算机专业中的地位，而且推动了其它理、工科专业的"程序设计"课程从以 Fortran、Basic 为主向以 Pascal 为主的转变。

Turbo Pascal 作为教学软件，除了具有 Pascal 语言本身的优点外，还具有：

1. 快速、高效。同样的机时可以比传统程序设计软件多开发几倍的程序。

2. 一个软驱，一张软盘就可以做设计练习了。

3. 功能齐备。Turbo Pascal 既可以象 Basic 那样绘图，也可以象 C 语言那样用于编写系统程序和调试，这对于后续的计算机课程如"计算机绘图"、"软件工程"等带来极大方便。

4. 学以致用。随着微机的迅猛发展，Turbo Pascal 将成为以后工作必不可少的软件。

本书详细介绍了 Pascal 语言及其程序设计，Turbo Pascal 的使用及其在工程中的应用，并提供了丰富的实例、习题。在语言教学上，本书始终贯穿着结构化程序设计思想，注意体现良好的程序设计风格，并注重编程和实际应用能力的培养。在 Turbo Pascal 的使用方面，本书以面向对象的 Turbo Pascal 5.5 版本为主，详尽地介绍了 Turbo Pascal 的基本功能和工程应用，并对已流行的其它版本也作了介绍。

本书总结了我们长期从事 Pascal 教学和 Turbo Pascal 软件开发的经验和体会，适合于作为大专院校"Pascal 语言"或"程序设计"等课程的教科书，也可作为 Turbo Pascal 软件自学教材和参考书。

此外，我们还提供了 40 多种和本书配套使用的教学辅导、应用软件。除了部分开发程序量较大需收少量开发费外，其余均可凭书末"软件索取单"免费索取。

本书由合肥工业大学计算机应用专业博士生指导教师张奠成教授和中国计算机软件公司高级顾问吴锤红先生主审。第 1、3～8 章由胡进为主编写，第 2、12～19 章及附录由薛伟为主编写，第 9～11 章由高洁平、朱卫国为主编写。全书由薛伟、胡进定稿。

参加本书部分章节编写的同志还有：合肥工业大学张维勇，中国科技大学夏勇，洛阳工学院陈科，南京机械专科学校王小龙。吴锤红还为本书提供了部分例题和软件。

本书的编写参考了美国 University of Michigan (Dearborn)的 Prof. T. Thomasma 在合肥工业大学开设"Turbo Pascal"、"计算机绘图"、"软件工程"等课程的教学过程和部分示例，在此特表示感谢。

由于水平有限，本书还存在一些错误和疏漏之处，欢迎读者指正。

作　者

1991 年 3 月于合肥工业大学

目　　录

第一章 概 述

计算机程序语言 Pascal 是由瑞士苏黎世工学院的 N. Wirth 教授于 60 年代末提出来的。从 1971 年正式推出迄今只不过 20 年的时间，Pascal 已成为世界上最广泛使用的程序设计语言之一。Pascal 语言系统地体现结构化程序设计的概念，具有丰富完备的数据类型、简明灵活的通用语句和清晰明了的模块结构，而且书写格式自由，运行效率高，查错能力强，移植性好，程序设计风格优美。Pascal 语言不仅适用于开发各种系统软件和应用软件，而且适用于程序设计的教学，在程序设计、数据结构、软件工程等方面的书刊中，大都用 Pascal 语言作为描述工具。这样可使学生养成结构化程序设计的良好习惯。因而国内外越来越多的学校都将 Pascal 语言作为程序设计教学的第一门语言。

§1.1 程序设计与软件

计算机根据输入给它的指令序列进行工作。一条指令对应于计算机执行的某一基本动作。程序就是一个指令序列，它规定了计算机执行的动作序列从而完成某一特定的任务。所谓程序设计就是对某一要处理的问题设计出解决它的指令序列，因而程序设计是一项创造性的工作。进行程序设计必须借助于描述语言，这就是程序设计语言。提供给计算机的所有指令都必须按照程序设计语言的语法规定来书写。程序设计语言分为低级语言和高级语言二大类。低级语言是指面向机器的语言，如机器语言和汇编语言；高级语言独立于机器，用高级语言编写的程序在不同类型的机器上要借助于不同的翻译程序。各种语言尽管性质、规则不同，但都含有如下的四种成分：

(1)数据成分，用以描述程序中所涉及的数据；

(2)运算成分，用以描述程序中所包含的运算；

(3)控制成分，用以表达程序中的控制结构；

(4)传输成分，用以表达程序中的数据传输。

这四种基本成分构成任意复杂的程序。从处理上看，这四种基本成分可分成数据成分和对数据成分操作二大部分，因而程序可以说是：

程序＝处理对象(即数据成分)＋处理规则

Pascal 语言程序作为一种高级程序设计语言的程序在大多数计算机上是不能直接运行的。它必须首先被翻译成计算机能懂的语言，即机器语言，然后才能在计算机上运行。完成这种翻译工作的是所谓的编译程序。Pascal 语言的编译程序接受由 Pascal 语言编写的程序——源程序，并将它翻译成等价的机器语言程序——目标程序。如果翻译成功，该目标程序就被保存在计算机的存贮的设备中，以便执行。上述过程，如图 1.1 所示：

执行目标程序时，将根据需要把数据输入给对应的数据对象，同时把运行结果按指定的方式显示或打印，如图 1.2。

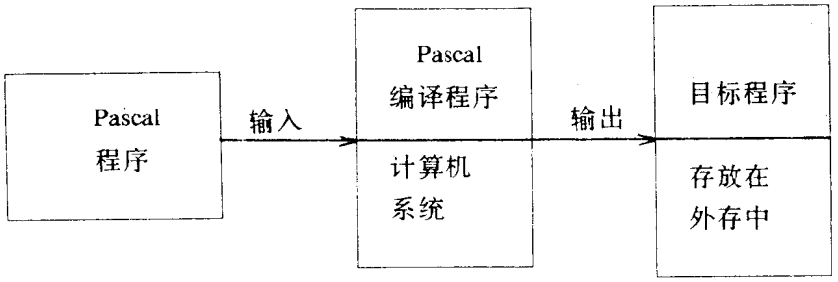


图 1.1　编译 Pascal 程序以备执行

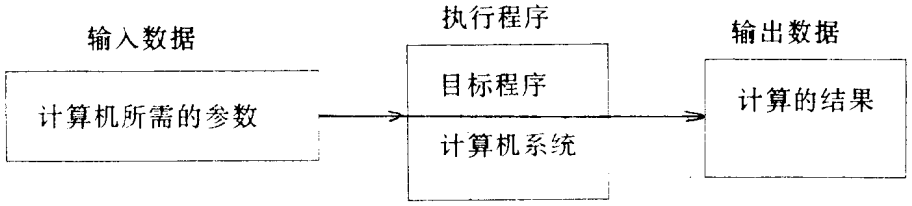


图 1.2　Pascal 语言的执行过程

一般而言，在一个源程序中难免会存在一些错误，错误一般可分为三类：

(1)编译时的错误：程序不符合 Pascal 语言的语法规定，又称为语法错误；

(2)运行时的错误：编译成功后在执行目标程序时还可能出现一些错误，使程序无法继续执行，例如数值的越界，除数为零等；

(3)逻辑错误：一个程序在编译时没有出现编译时的错误，在执行时也没有出现运行时的错误，但还可能得到不正确的结果。这是由于在算法的设计中或程序的表达中存在逻辑的错误。

随着计算机的不断发展，软件的开发和生产已成为一种被称为软件工程的新兴行业。按照国际标准化组织(ISO)的定义，软件是指“与计算机系统的操作有关的计算机程序、过程、规则及任何与之有关的文档说明”；通俗地说，软件由计算机程序与相应的各种文档组成，如研制计划书、设计说明书、使用说明书等都是程序文档的典型例子。程序文档在提高程序的可读性以及便于程序维护、发展与交流等方面可发挥明显的效益。因此把软件产品单纯理解为计算机程序是不全面的。请大家记住：

软件＝程序＋相关文件。

§1.2　算法和框图

我们在编制程序之前，首先要想好求解这个问题的方法和步骤，然后再按照这个思路用程序设计语言来具体地实现之。非形式化地，我们把求解问题的方法就称为算法。正确的算法必须满足下面三个条件：

(1)算法中的每一步必须是一条能执行的指令，即算法的有效性；

(2)各步的顺序必须精确地确定，即算法的明确性；

(3)算法最后必须能结束，即算法的有穷性。

一个问题往往有不止一种算法，不同的算法之间便有优劣好坏之分。评价算法一般是按照执行

算法所需要的时间长短及所占空间的大小来进行，这二条原则有时可能相互抵触。评价算法是“数据结构”、“算法的分析和设计”、“软件工程 ”等课程的内容，超出了本书的范围。本书第 4.7 节也给出了判断一个程序好坏的一些标准。

描述算法往往用框图或伪码，本书的算法均是用框图来描述的。传统的框图比较直观，它用矩形框表示要进行的工作，用菱形框表示判断，用箭头表示执行顺序的方向。这种框图的随意性很大，可以描述任意结构的算法。还有一种叫做结构化的框图，它适合于描述结构化算法。下面我们介绍一种结构化框图的画法，该种框图又称为纳萨－施内德曼图(Nassi－Schneiderman)。

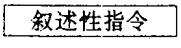
把简单的指令、叙述性的指令写在一个矩形框中，如图 1.3(a1)和(a2)。

把判断性条件放到三角形中，根据三角形中的条件的真假，决定执行哪一部分矩形中的指令。如图 1.3(b1)和(b2)。

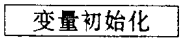
多分支情况根据分支的具体值决定执行哪部分指令，如图 1.3(c1)所示。图 1.3(c2)是表示当情况为 i 时，输出 Si 的值。

用包围循环中指令的 L 形框表示算法的循环，外框中的条件就是控制循环的条件。图 1.3(d1)和(d2)表示，当条件为真时，循环执行重复部分，直到条件不满足。图 1.3(e)表示先执行重复部分，然后再检验条件，若条件不满足，继续循环；否则就跳出循环。

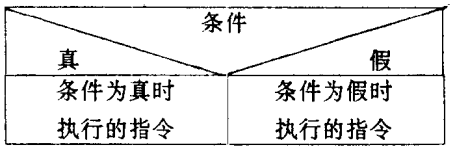
过程的调用如图 1.3(f)所示，过程和函数的定义如图 1.3(g)、图 1.3(h)所示。



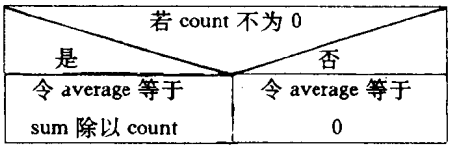
(a1)



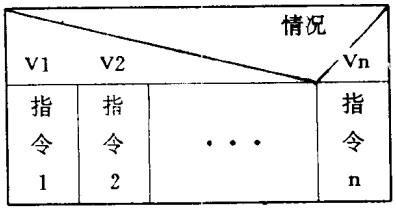
(a2)



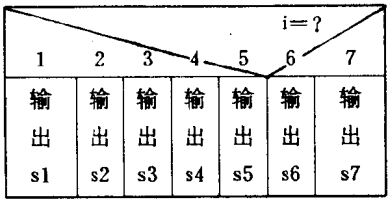
(b1)



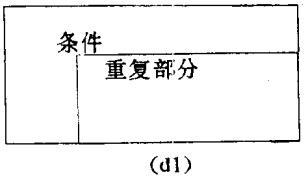
(b2)



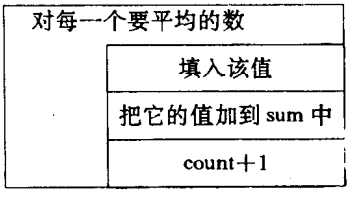
(c1)



(c2)



(d1)



(d2)

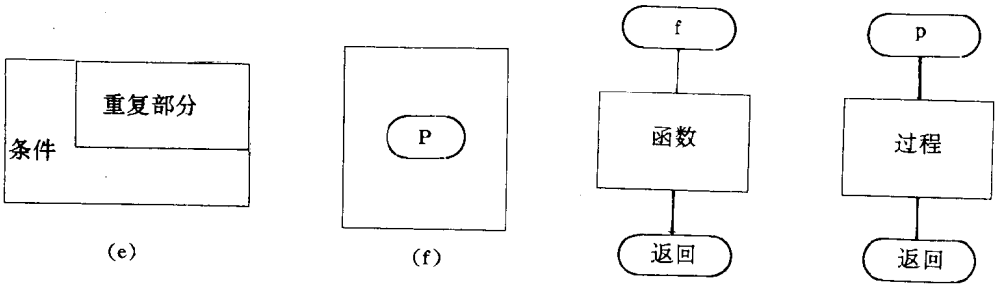


图 1.3

通过组合这些框图，我们就可以描述复杂的算法。

例 1.1 用结构化框图表示求平均数的算法。

框图如下：

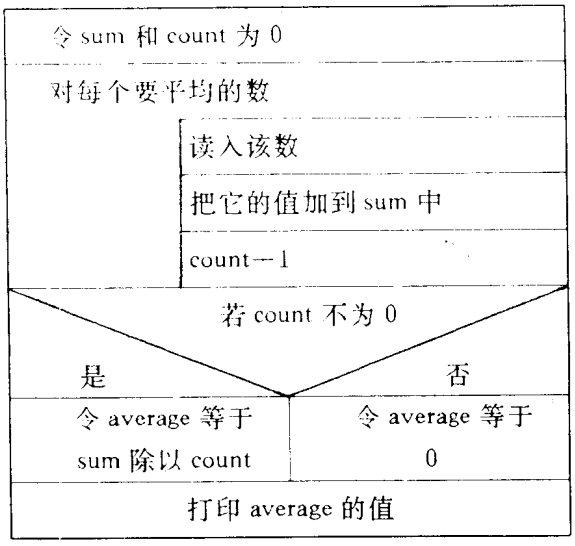


图 1.4

最后，请读者记住一个著名的公式：

程序设计＝算法＋数据结构

N. Wirth 教授曾以此作为他的一部书的书名。

§1.3 结构化程序设计与 Pascal 语言

结构化程序设计的概念和方法是由荷兰学者 E. W. Dijkstra 首先提出来的。从表面上看，它的提出起因于对 goto 语句(转移语句)的评价。众所周知，goto 语句是一种相当灵活，可改变程序执行流程的控制语句，有人喻之为“万能语句”。在缺乏其它控制语句的 Basic 语言和标准 Fortran 语言(Fortran IV)中，goto 语句往往用得很多。但是，过多地使用 goto 语句，特别是逆向转移的 goto 语句，会使程序的静态结构(由程序员书写的程序文本结构)与动态结构(机器执行计算时的结构)差别很

大，有时还会导致乱线团式的程序，这一点正是降低程序可读性、增加调试与维护困难的因素。但是，goto 语句有时又很有用处，比如从一个循环中退出。所以不能从形式上以是否使用 goto 语句或其多少作为判定程序质量的标准，而应以程序的可读性、静态结构与动态结构的一致性作为判定的依据。

结构化程序设计就是为了使程序具有合理结构，以便保证和验证其正确性而规定的一套进行程序设计的方法。用结构化程序设计的方法设计出来的程序称为结构化程序，而反映了结构化程序设计的要求和限制，便于用来书写结构化程序设计的语言就称为结构化程序设计语言。用这种语言书写的程序易于保证正确性。以上的叙述，与其说是一种定义，不如说是一种要求。现将它的具体含义归结为以下几点：

(1)基于自顶向下，逐步求精的设计方法。我们在第 4.7 节中将详细介绍；

(2)程序按页书写，遵循一定的格式，使结构清晰。我们将在第 4.7 节集中说明程序书写的格式；

(3)程序分成模块，每个模块具有独立的功能，模块之间的联系要简单，每个模块只能有一个入口和一个出口；

(4)程序中只包含三种最基本的结构：顺序、分支、循环。即描写程序的框图只能是下面三种框图的组合：

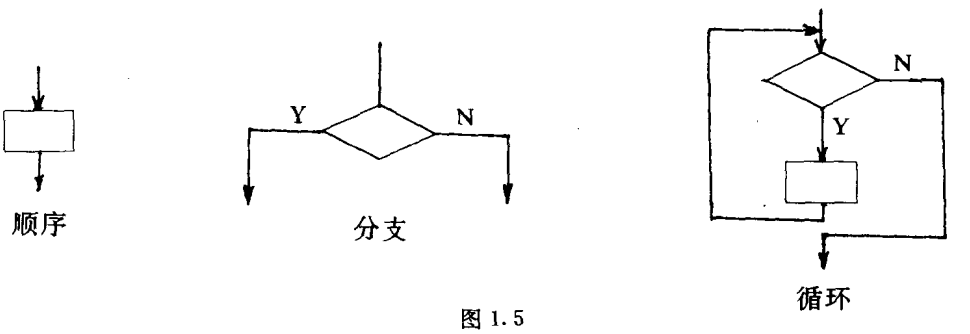
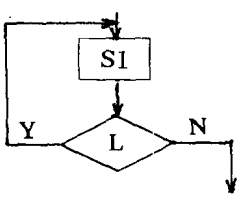


图 1.5

下面我们看几个例子：

例 1.2　下面的框图不符合结构化程序设计的要求，请将其结构化。

(1)



S1 是一个功能描述，L 是判断条件，该框图不是上面三种框图的组合，可改造如右图所示。这二个框图的功能是完全一样的。