高等专科学校教材

PASCAL程序设计

姜文清　王宇颖　王孟石

西安电子科技大学出版社

高等专科学校教材

PASCAL 程序设计

姜文清　王宇颖　玉孟石

西安电子科技大学出版社
1990

内 容 简 介

本书系统地介绍 PASCAL 语言和结构程序设计的基本方法．全书共分 10 章，主要内容包括：PASCAL 语言的各种语句、各种数据类型、程序构造以及程序设计与调试的基本方法．各章后面附有习题，这有利于学生消化理解所学内容．书后的附录为学生上机实践提供了指导性的参考资料．

本书力求体现"结构程序设计"的基本思想，注重培养学生良好的程序设计风格和实践能力．全书写得深入浅出，通俗易懂．

本书经电子工业部大专类计算机教材编审委员会推荐，可作为高等专科学校计算机各专业"PASCAL 程序设计"课程的教材，也可作为计算机系本科各专业的教材，并可供其它有关专业的学生和从事计算机工作的人员参考．

高等专科学校教材

PASCAL 程序设计

姜文清　王宇颖　王孟石

责任编辑　夏大平

西安电子科技大学出版社出版

西安电子科技大学印刷厂印刷

陕西省新华书店发行　各地新华书店经售

开本 787×1092 1/16　印张 16 8/16　字数 388 千字

1990 年 6 月第 1 版　1990 年 6 月第 1 次印刷　印数 1—3 000

ISBN 7-5606-0117-0 / TP・0042　　定价：3.30 元

出 版 说 明

根据国务院关于高等学校教材工作分工的规定，我部承担了全国高等学校、中等专业学校工科电子类专业教材的编审、出版的组织工作。由于各有关院校及参与编审工作的广大教师共同努力，有关出版社的紧密配合，从1978年至1985年，已编审、出版了两轮教材，正在陆续供给高等学校和中等专业学校教学使用。

为了使工科电子类专业教材能更好地适应"三个面向"的需要，贯彻"努力提高教材质量，逐步实现教材多样化，增加不同品种、不同层次、不同学术观点、不同风格、不同改革试验的教材"的精神，我部所属的七个高等学校教材编审委员会和两个中等专业学校教材编审委员会，在总结前两轮教材工作的基础上，结合教育形势的发展和教学改革的需要，制订了1986～1990年的"七五"(第三轮)教材编审出版规划。列入规划的教材、实验教材、教学参考书等近400种选题。这批教材的评选推荐和编写工作由各编委会直接组织进行。

这批教材的书稿，是从通过教学实践、师生反映较好的讲义中经院校推荐，由编审委员会(小组)评选择优产生出来的。广大编审者、各编审委员会和有关出版社为保证教材的出版和提高教材的质量，作出了不懈的努力。

限于水平和经验，这批教材的编审、出版工作还会有缺点和不足之处，希望使用教材的单位、广大教师和同学积极提出批评建议，共同为不断提高工科电子类专业教材的质量而努力。

电子工业部教材办公室

前　言

PASCAL 语言是在探索结构程序设计方法的过程中诞生的。这种语言用于描述结构程序设计，目前在我国已经普遍成为程序设计、数据结构以及有关课程的教学工具。本书系统地介绍 PASCAL 语言和结构程序设计的方法，使学生掌握程序设计规范化和工程化的基本技能。

本书由哈尔滨工业大学姜文清副教授主编，南京大学钱士钧副教授主审，南京化工动力专科学校李中震副教授任责任编委。参加编写的还有哈尔滨工业大学的王宇颖同志和沈阳黄金学院的王孟石同志。

本书第一章为概论，介绍计算机、程序设计、结构程序设计的发展，以及 PASCAL 语言及其程序的概貌。第二章介绍 PASCAL 语言的标准数据类型、标准函数、常量和变量的定义说明，并且介绍赋值与读写数据的基本语句；最后讨论结构程序设计的方法和步骤，以期引导学生入 PASCAL 语言和程序设计之门。第三章介绍 PASCAL 语言程序的控制结构，包括构成分支和循环的各种控制语句；并且通过较多的程序设计问题实例，使学生掌握结构程序设计的基本方法。在第三章的最后，介绍阅读程序和验证程序正确性的传统方法——算法跟踪，并且讨论程序的调试与测试方法，为学生在计算机上开发程序打下基础。第四章介绍子程序的定义说明及其调用，并且讨论自顶向下的程序设计以及模块化程序的调试方法。第五章介绍枚举、子域数据类型及其在程序设计中的应用。这一章的最后介绍一种简单的结构化数据类型——集合以及实现数学中集合基本运算的方法。第六、七两章分别介绍 PASCAL 语言的两种结构化数据类型：数组和记录，阐明它们的结构形式、访问方法及其应用。第八章介绍文件类型以及在 PASCAL 语言程序中读、写顺序文件和正文文件的基本方法。第九章讨论子程序调用的一些较深入的问题：递归、向前引用、子程序做参数以及函数的副作用等。第十章介绍指针类型，并且通过实例讨论两种基本的动态数据结构：链表和二元树。

为了帮助学生学习结构程序设计方法，对一些应用实例，本书以"问题"的形式，按照结构程序设计的方法和步骤给出详细的解答。本书例题充分，题材广泛，能激发读者的学习兴趣。为了加强程序风格训练，除了通过实例作出示范之外，在有关章节的最后还给出了关于程序风格的说明。

考虑学生上机和实际应用的需要，在介绍标准 PASCAL 语言的同时，在有关的章节里，扼要地说明了目前普遍使用的在微型机上实现的 PASCAL 语言：MS PASCAL、UCSD PASCAL 和 TURBO PASCAL，说明它们对标准 PASCAL 语言所做的若干修改和扩充。除此之外，在书后的附录中简明地介绍了这 3 种实现系统及其上机过程。最后的附录七是在 VAX-Ⅱ计算机 VMS 操作系统上实现的 PASCAL 语言及其上机过程。这些附录为学生上机提供了指导性的参考资料。

本书内容系统、丰富，深入浅出，适用于 60～80 学时的教学。目录中带有 * 号的部分可根据教学时数适当地选择。

本书适用于高等学校专科和本科计算机专业学生学习，也可供有关专业的读者阅读参

考。

在编写本书的过程中，得到了哈尔滨工业大学计算机系领导和软件实验室全体同志的大力支持，南京大学计算机系钱士钧副教授详细地审阅了本书的初稿，提出了许多宝贵意见，对此，编者一并表示衷心感谢。

由于编者水平有限，书中难免存在缺点和错误，殷切希望广大读者批评指正。

编　者

1989 年 2 月

目　录

第一章 概 论

1.1 计算机系统

计算机是一种能自动、高速地进行数值计算、信息处理、自动控制等多方面工作的电子设备。通常所说的计算机，指的是电子数字计算机系统。计算机系统由两部分组成，一部分称为硬件，另一部分称为软件。硬件是计算机系统中实际装置的总称，包括电子、磁性、机械、光学等元器件，以及由这样的元器件组成的部件和装置。软件是相对硬件而言的，包括计算机运行所需要的各种程序、数据和有关资料。软件在计算机日常工作时是不可缺少的，它可以扩大计算机的功能，使之适于各种应用。

下面，简要说明计算机硬件和软件的基本组成以及它们的基本功能。

图 1.1 是计算机硬件系统的基本组成。其中：

(主)存储器 它存放计算机运行时执行的程序和它所处理的数据。存储器分成许多存储单元，每个单元由若干二进制位组成，每个二进制位能表示 0 或 1 的一位二进制数字。通常，8 个二进制位组成一个字节，每个字节都有一个编号。这些编号称为地址。2 个、4 个或更多字节组成一个机器字。机器字中包含二进制位的个数称为字长。有的机器对每个机器字设置编号，这种方式称为按字编地址。常见的微型计算机，比如 IBM PC，其字长为 16 位，即每个字包含 2 个字节。小型或大型计算机的字长有 32 位、48 位或者 64 位等等。一般来说，计算机主存储器的容量是有限的，比如 IBM PC 计算机主存储器的容量为 640 K(1K = 1 024)字节，只能存放当前执行的程序和有关的数据。

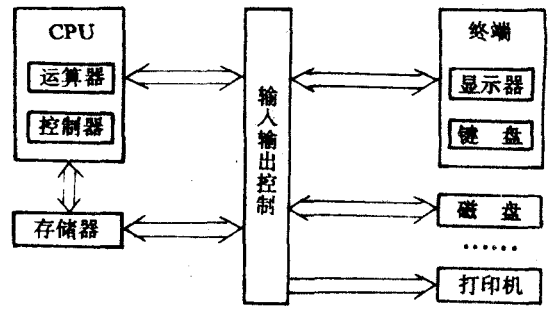

图 1.1 计算机硬件系统

运算器 它从主存储器取出数据，按照控制器发出的运算命令，执行算术或逻辑运算，并且把运算的结果存入主存储器。

控制器 它从主存储器中取出程序指令，按照指令要求的操作，向有关的功能部件发出操作命令，并且检查功能部件的工作状态，控制整个硬件系统协调工作。

通常把运算器和控制器统称为 CPU(Central Processor Unit)，即中央处理器。在使用大规模集成电路的计算机中，把 CPU 做成一个芯片。CPU 的性能在很大程度上决定计算机的运算速度和基本功能。

输入输出控制 它按照控制器发出的命令，控制主存储器与外部设备之间的数据传送，并且把外部设备的工作状态反馈给控制器。外部设备指的是磁盘驱动器、终端、打印机等等。

磁盘 它是一种外存储器，用于长期存放程序和数据．磁盘有两种：软盘和硬盘．软盘的容量小、速度慢，盘片可以随时更换．硬盘的容量大、速度快，但盘片不能更换．IBM PC 计算机上一般使用存储容量为 360 k 或 1.2 M 字节的软盘，20 M 字节的硬盘．

终端 它是人与计算机之间传送命令、程序和数据的常用设备．终端由两部分组成：键盘和屏幕显示器．键盘是一种输入设备，用于输入命令、程序和数据等等．屏幕显示器是一种输出设备，用于输出数据．通常，由键盘输入的信息也在屏幕上显示出来．

打印机 它是一种输出设备，把计算机执行的结果打印在纸上，供人们阅读参考．打印时按行进行，通常，每行最多可以打印 80 或 132 个字符．

在计算机系统中，除以上列举的外部设备之外，还可能有卡片输入机、磁带机、绘图机以及专用的控制设备等等．在不同种类不同用途的计算机系统中，可能配有不同的外部设备．图 1.1 所示的是常见微型计算机的基本配置．

下面说明计算机的软件．计算机与其它机器最主要的差别在于，它能按照人们预先编制的程序完成各种不同的工作．程序是一个指令集合．人们把要求计算机完成的工作用一定形式的指令编排出来，便得到程序；并且把这一过程称为程序设计．图 1.2 所示的是常见的计算机软件．其中：

操作系统 它是一个程序集合，用于管理计算机系统中全部硬件和软件资源，维持整个系统日常协调工作．通常，提供给用户的是在操作系统管理之下的计算机，用户通过操作命令控制计算机的工作．

编辑程序 它是在计算机上开发程序的工具，借助编辑程序可把编好的程序输入计算机，并且存入外存储器．对已经存在外存储器上的程序，还可以借助编辑程序进行修改．

编译程序 编程序的时候要使用一种便于描述解题过程的语言，称为程序设计语言，比如 PASCAL．用程序设计语言编写的程序称为源程序．通过编译程序可把源程序翻译成计算机能够直接执行的程序．这种计算机能够直接执行的程序称为目标程序．目标程序是用二进制代码表示的机器指令序列，每条指令表示计算机执行的一步操作．

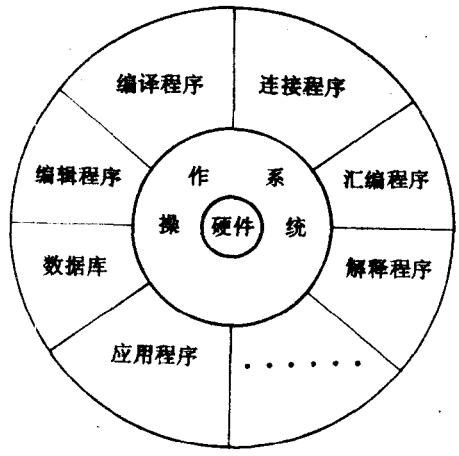

图 1.2 计算机软件系统

连接程序 其作用是把同一个程序中独立存在的几部分目标程序连接、装配在一起，形成一个完整的可执行的目标程序．通常，把编译程序产生的目标程序和系统提供的库子程序连接装配成可执行的目标程序．

汇编程序 在程序中表示计算机一步基本操作的命令称为指令．一种计算机能执行的所有种类的指令，构成这种计算机的指令系统．为了便于用指令编程序，用约定的符号表示指令．用符号指令写出的程序称为汇编语言程序．汇编程序把汇编语言程序翻译成目标程序．汇编语言程序和诸如 PASCAL 之类的高级语言程序的差别在于，汇编语言程序和目标程序具有一一对应的关系，并且依赖于计算机的硬件；而高级语言程序和目标程序不

是一一对应的关系，并且不依赖于计算机的硬件。

随着计算机科学的发展，软件的种类愈加繁多，日新月异。本书对与学习本课程无关的内容不再赘述。

1.2 结构程序设计

本世纪 50 年代，编程序使用二进制的指令代码或者汇编语言。在编程序之前，对计算机主要功能部件的工作方式，存储器地址的形成和表示方法，指令的形式等等，都要有相当清楚的了解。编程序时要考虑实现每一步运算的详细操作过程，用一条条的指令表示出来。这种程序设计相当麻烦。编制和调试一个具有实用价值的程序往往要用几个月甚至几年的时间。而且，用一种机器的指令或汇编语言编写的程序，在另一种机器上完全不适用。这样，对非计算机专业人员来说，使用计算机谈何容易。当时，评价程序的坏好就是以指令条数的多寡、占用空间的大小和执行速度的慢快作为衡量标准的。

本世纪 50 年代中期出现了 FORTRAN 语言，开始改变了只有计算机专业人员才能编程序的局面。用 FORTRAN 语言编程序不必考虑计算机的内部情况，也不需要了解其指令系统。一个 FORTRAN 程序很接近于一个解题过程的英文说明。对非计算机专业人员来说，只要经过短期学习，写一个具有实用价值的程序，用几小时或几天的时间便可以完成。而且，同一个程序，不需要修改，可以在不同种类的计算机上运行。这样，为计算机的普及应用开辟了前景。但是，最初的 FORTRAN 语言，其程序的质量差，错误多，编译的时间也长。

随着计算机日益广泛地渗透到各个学科和技术领域，本世纪 60 年代发展了一系列不同风格、为不同对象服务的程序设计语言。当时被采用的语言竟达三四百种。其中较为著名的有 ALGOL、LISP、SNOBOL、COBOL 和 PL／I 等等。高级语言的蓬勃兴起，使得编译理论日趋完善。但是，对程序设计方法来说，并没有实质性的改进。

自本世纪 60 年代末到 70 年代初，出现了大型软件系统，比如操作系统和数据库。研制一个大型软件系统，不仅需要花费大量的人力和资金，而且研制出来的产品常常可靠性差，错误多，维护和修改也很困难。一个大型操作系统，投入几千人年的工作量，研制的结果，常常会隐藏着几百甚至几千个错误。当时，人们把这种现象称为“软件危机”。

“危机”震动了软件界，促使人们开始重新研究程序设计中的一些最为基本的问题。例如，程序的基本组成部分是什么？应该用什么样的方法来设计程序？程序设计的主要方法和技术应该如何规范化和工程化等等。

1968 年，荷兰计算机科学家图灵奖获得者 E.W.Dijkstra 首先提出了结构程序设计的思想，强调从程序结构和程序风格来研究程序设计。经过几年的探索和实践，结构程序设计的应用取得了成效。结构程序设计方法是为了使程序具有合理结构，以保证其正确性而规定的一套程序设计方法。用结构程序设计方法设计出来的程序称为结构化程序。结构化程序不仅结构合理，易于编写和阅读，而且易于验证它的正确性。在这样的历史背景下，瑞士的 N. Wirth 教授推出了结构程序设计语言 PASCAL。结构程序设计语言的语言成分反映了结构程序设计的要求和限制，因而用它可以写出结构化的程序。

推行结构程序设计方法的意义远不止于当前的实用价值，更深的意义在于它向人们揭

示了研究程序设计方法的必要性。程序设计并不是一种纯技术性的活动，与任何科学一样，它自身也有一套基本的原理和方法。本世纪 70 年代程序设计技术的发展是以方法论为特征的。

程序设计的一个基本原则是抽象。为了解决一个复杂的问题，人的智力往往不可能一下子就触及到问题的细节方面。在分析了问题的要求之后，我们总是首先制定一个大体的解题方案，称为抽象算法。抽象算法在抽象数据上实施一系列抽象操作，这些数据和操作反映了解题的基本思想，而将所有细节都抽象掉了。然后，下一步再考虑这些抽象数据和抽象操作的具体实现。在抽象级，我们只需要知道"做什么"，而在实现级才考虑"如何做"。由于采用抽象技术，使大问题分解成相对独立的一些子问题，它们分别只涉及局部的环境和条件，可以独立地一个一个地解决，从而使整个问题得到圆满解决。这种程序设计思想将贯穿本书的始终。从第二章开始，我们将通过问题实例，详细介绍结构程序设计的方法。

从计算机教育的角度来看，一个好的程序员必须经过程序设计规范化和工程化的严格训练。美国康奈尔大学 D.Grics 教授曾做过一次试验。他邀请了一些研究生、程序员和来自工业界的专业人员共四五十人，同时编写一个关于处理"行向右对齐"问题的程序。结果，竟有半数人程序编得不对。而 D.Grics 本人按照规范化的结构程序设计方法写出来的程序，结构合理，清晰易读，而且用的时间也不比别人长。这个例子说明，不用结构程序设计的方法，甚至采取拼、凑、对付、勾勾抹抹的做法也能编出程序，但是，这样的程序很容易出错；倘若程序大一点，则难免漏洞百出，有时还会束手无策。所以，在一开始学习程序设计时，就应进行程序设计规范化和工程化方面的训练，把按步骤写出程序设计的全过程视为一种艺术创作。

1.3 PASCAL 语言及其程序的编译与执行

1.3.1 PASCAL 语言

PASCAL 语言是 N.Wirth 教授于 1969 年在瑞士苏黎世联邦工学院设计成功的。1970 年在 CDC-6000 计算机上首次实现 PASCAL 编译。1971 年发表了 PASCAL 语言用户手册，1974 年又发表了修订报告。1978 年国际标准化组织(ISO)发表了关于 PASCAL 的建议草案。命名为 PASCAL 是为了纪念法国数学家 B.Pascal，他是世界上第一台机械式加法器的创造者。

PASCAL 语言是在算法语言 ALGOL-60 的基础上发展起来的，它能够清晰地描述程序设计中算法和数据的逻辑结构。PASCAL 语言特别适用于"程序设计"、"数据结构"、"编译技术"等课程的教学，所以普遍地被世界各国高等学校计算机专业所采用。此外，由于 PASCAL 语言的语法规范、严谨、简明、精炼，便于在各种不同的计算机上实现，所以，近几年来，各种计算机上几乎都配有 PASCAL 编译程序。

PASCAL 的各种实现系统与国际标准化组织建议的标准 PASCAL 相比，一般都做了适当的修改和扩充，并且和标准 PASCAL 兼容。这样，按照标准 PASCAL 写出的程序，在不同的实现系统上一般都能通过。本书以介绍标准 PASCAL 为主。考虑目前我国普遍使用以 Intel 8086、8088、80286 系列产品为 CPU 芯片的微型计算机，比如

IBM PC、长城 0520、Apple-Ⅱ等，而在这类计算机上实现的 PASCAL 主要有 MS PASCAL、UCSD PASCAL 和 TURBO PASCAL。所以，在介绍标准 PASCAL 的同时，简要说明这些实现系统有关的修改和扩充。但是，本书不是"用户手册"，不能逐一列举其全部细节。本书中的程序例基本上都是按照标准 PASCAL 写的。有的章节，考虑教学与上机实践的需要，引用了 MS PASCAL 所做的扩充，比如串类型，CASE 语句中的缺省标号(OTHERWISE)以及编译控制行(确切地说，它不属于 PASCAL 语言)等等。事实上，在大多数系统中都有类似的扩充。

1.3.2 程序的编译与执行

我们知道，计算机能直接执行的是机器代码程序。为了把 PASCAL 语言源程序变成机器代码程序，要用 PASCAL 语言编译程序对源程序进行编译，如图 1.3 所示。在编译过程中，如果发现源程序中存在不符合 PASCAL 语法规则的错误，则输出诊断信息，指出错误的位置和性质。我们把这种错误称为编译错误。然后，要修改源程序，排除编译错误。如果在编译过程中没有发现编译错误，则产生机器代码的目标程序。

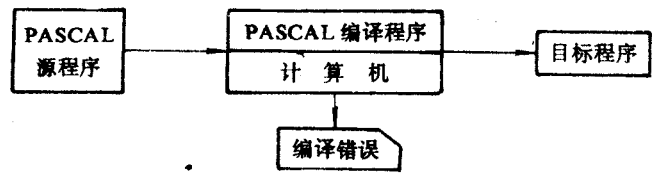

图 1.3 编译源程序

执行时，计算机按照目标程序中的指令完成所要求的操作。如果程序要求输入数据，则启动输入设备，等待我们输入数据；当输入数据之后，程序继续执行。如果程序中存在逻辑错误，则在执行时可能半途而废，也可能产生离奇的输出结果。我们把这种情况称为执行错误。在产生执行错误的情况下，大多数系统都能输出诊断信息，指出程序执行失败。在这种情况下，我们要按照程序设计的步骤，仔细地进行检查，最后纠正程序中的错误。如果程序没有逻辑错误，则按照要求把计算的结果输出，如图 1.4 所示。

如何避免程序的语法错误，这是学习 PASCAL 语言应该注意的问题。怎样避免程序的执行错误，这是学习程序设计应该注意的问题。只要严格地按照步骤进行程序设计，牢固地掌握 PASCAL 语言的语法规则，总可以避免可能出现的各种错误。

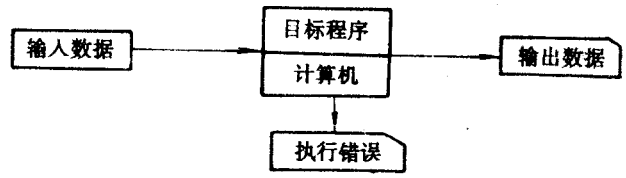

图 1.4 执行目标程序

1.4 程 序 结 构

这一节通过一个程序例子介绍 PASCAL 程序的结构和 PASCAL 语言的概貌。

[例 1.1] 表 1.1 程序执行时在显示器上显示出一道道的算术题。比如，

217 + 366 = ?

当从键盘打入其和数 583 之后，接着显示出下一道算术题，我们再打入相应的和数。如果打入的和数不对，比如，

591 + 972 = ? 1463

下边画横线的是输入数据，这时，显示器上将显示出一句英文

YOU ARE WRONG.

说明你的答案不对。当做完 20 道算术题之后，计算机给你评分。比如，

YOUR SCORE IS 85%

说明你得 85 分，然后程序结束。

表 1.1　例 1.1 的程序

```
{1}   PROGRAM MATHGAME (INPUT, OUTPUT);
{2}   (* 加法游戏程序 *)

{3}   VAR
{4}     FIRST, SECOND, SUM, SEED: INTEGER;
{5}     TIME, SCORE: INTEGER;

{6}   FUNCTION RANDOM: INTEGER;
{7}   (* 产生 0 到 1024 之间的一个随机整数 *)

{8}   CONST
{9}     MUDULO = 1024;
{10}    FACTOR = 29;
{11}    INCREMT = 217;

{12}  VAR
{13}    NEW: INTEGER;

{14}  BEGIN
{15}    NEW := FACTOR * SEED + INCREMT;
{16}    SEED := NEW MOD MUDULO;
{17}    RANDOM := SEED
{18}  END; (* RANDOM *)

{19}  BEGIN
{20}    SEED := 0;    (* 随机数初值 *)
{21}    SCORE := 0;   (* 记分初值 *)
```

```
{22}      FOR TIME := 1 TO 20 DO
{23}        BEGIN
{24}          FIRST := RANDOM;
{25}          SECOND := RANDOM;
{26}          WRITE (FIRST: 4, ' + ', SECOND: 4, ' = ');
{27}          READLN (SUM);
{28}          IF SUM = FIRST + SECOND THEN
{29}            SCORE := SCORE + 5
{30}          ELSE
{31}            WRITELN('YOU ARE WRONG.')
{32}        END; (* FOR *)
{33}      WRITELN('YOUR SCORE IS', SCORE: 3, '%')
{34}  END.
```

为了便于说明，在程序的每一行之前以注释的形式加一个编号——在实际的程序里是不需要的。在本程序中用一对花括号{ }或者用一对(* *)括起来的部分都是注释，是给人看的，为阅读程序提供方便，对程序本身不起作用，而且可以出现在空格或换行能出现的任何位置。

每个 PASCAL 程序结构由程序首部和分程序组成。

程序首部 程序的开头部分是程序首部，比如例 1.1 程序中的第{1}行。其中的 PROGRAM 称为保留字，标识程序首部。紧跟在 PROGRAM 之后的 MATHGAME 是一个标识符，标识本程序的名称。括号里的 INPUT 和 OUTPUT 是程序的参数，在微型计算机的标准运行环境中，分别表示键盘和显示器。

PASCAL 语言的各种语法成分都可以用一个语法图表示。比如，程序和程序首部可以分别用图 1.5 所示的两个语法图表示。在语法图中，用圆框或者带有圆弧的框围起来的

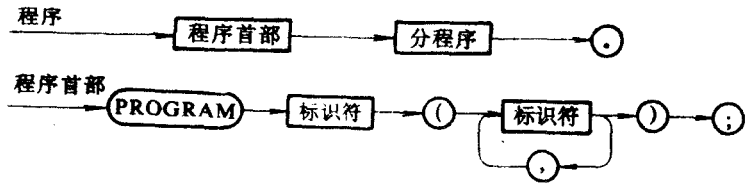

图 1.5 程序和程序首部语法图

是程序中出现的具体成分；用方框围起来的是需要进一步定义说明的语法成分。比如，在"程序"的语法图中，程序首部和分程序都需要进一步定义说明；而在程序首部的语法图中，标识符需要进一步定义说明。

分程序 分程序包括说明部分和语句部分。

说明部分。在程序的说明部分里包含若干定义与说明，比如程序中引用的变量、用标识符表示的常量、自定义的函数和过程等等。在例 1.1 的程序里，从第{3}行到第{18}行

都是程序的说明部分。第{3}行的 VAR 是一个保留字，标识它后面的是变量说明，在第{4}、{5}行共说明 6 个变量，用来表示整型数。第{6}行的 FUNCTION 是一个保留字，标识它后面的是一个函数定义，其函数名是 RANDOM，函数值是一个整型数；这一行是函数的首部；其后，第{7}行到第{18}行是这个函数的分程序。程序说明部分的语法图如图 1.6 所示。

本书中以后各章节涉及到的语法图都集中列于书后附录一，不再单独列出。

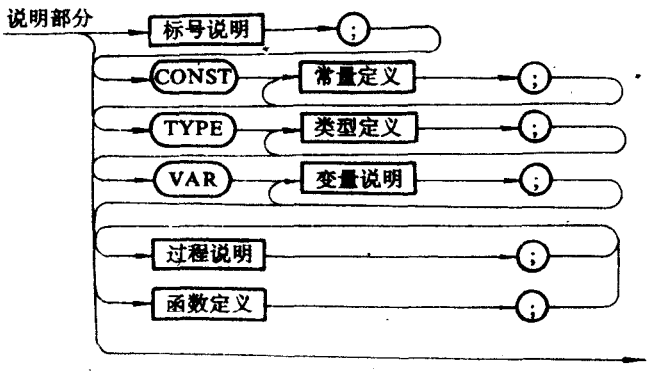

图 1.6　说明部分的语法图

语句部分。程序的语句部分继说明部分之后，从 BEGIN 开始，到和它配对的 END 结束。在 BEGIN 和 END 之间都是程序将要执行的语句。每个语句执行一定的操作，语句之间用分号分隔。由一对 BEGIN END 括起来的一组语句称为一个复合语句。在例 1.1 的程序里，从第{19}行到第{34}行是程序 MATHGAME 的语句部分。现对各语句将要执行的操作说明如下：

```
{19}  BEGIN
{20}     随机数 SEED 的初值设置为 0;
{21}     记分 SCORE 的初值设置为 0;
{22}     变量 TIME 的值从 1 增到 20，重复{24}到{31}的操作:
{23}       BEGIN
{24}          置 FIRST 的值为一个随机数;
{25}          置 SECOND 的值为另一个随机数;
{26}          输出算式"一个数+另一个数 = ";
{27}          输入我们算的和数，用 SUM 表示;
{28}          如果我们算对了，那么
{29}            记分 SCORE 的值增 5
{30}          否则
{31}            输出"你错了"
{32}       END; (* 重复到此为止 *)
{33}     输出"你的得分是 SCORE 记下的值"
{34}  END.
```

另外，在本例的函数定义里，第{14}行到{18}行是这个函数的语句部分。

从例 1.1 的程序我们可以看到，在程序 MATHGAME 的分程序里，其说明部分包含函数 RANDOM 的定义，而在语句部分通过函数名 RANDOM 引用这个函数。类似地，

在函数和过程的分程序里还可以包含另外的函数定义和过程说明，并且在语句部分引用这些函数和过程。PASCAL 语言的这种程序结构称为嵌套结构，这是 PASCAL 程序结构的重要特点。与此相反，FORTRAN、C 等语言不允许有嵌套的程序结构。

1.5 PASCAL 语言的符号

基本字符 在程序中使用的字符受计算机系统的限制。在一般情况下，PASCAL 程序中允许出现的字符如下：

(1) 英文字母，共 26 个，不分大写和小写；本书中，英文字母一律采用大写，即

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

(2) 数字，共 10 个：

0 1 2 3 4 5 6 7 8 9

(3) 其它符号，共 20 个：

+ − * / ^ . , : ; 空格 = < > ′ ( ) [ ] { }

在字符串里可以出现实现系统中能表示的任意字符，比如，

! ″ # $ % & _ ? @

等等，而且区分大写和小写字母。

程序中的任何语法成分都是由以上三种基本字符表示的。

保留字 下列符号在 PASCAL 语言中表示特定的语法单位，称为保留字。

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

下列单个或者固定组合或者配对使用的特殊符号，在程序中用做运算符或者分隔符：

+	−	*	/	^	.	:=	..	<	<=
=	<>	>	>=	,	:	;	空格		
( )	[ ]	′ ′	{ }	(*	*)				

标识符 在程序中用来表示名称的符号称为标识符。比如，例 1.1 程序中的程序名 MATHGAME，函数名 RANDOM，常量名 MUDULO 以及变量名 FIRST、SCORE 等等都是标识符。标识符是在编程序时确定的。标识符必须以字母开头，后边可以跟若干字母或者数字。标识符中符号的个数，按照标准 PASCAL，没有具体规定。但是，在具体的实现系统中，总是把前边的有限个符号认为是有效的。比如，在 UCSD PASCAL 系统中，按照前边的 8 个字符区分不同的标识符。这样，像 *OPERAND1*、