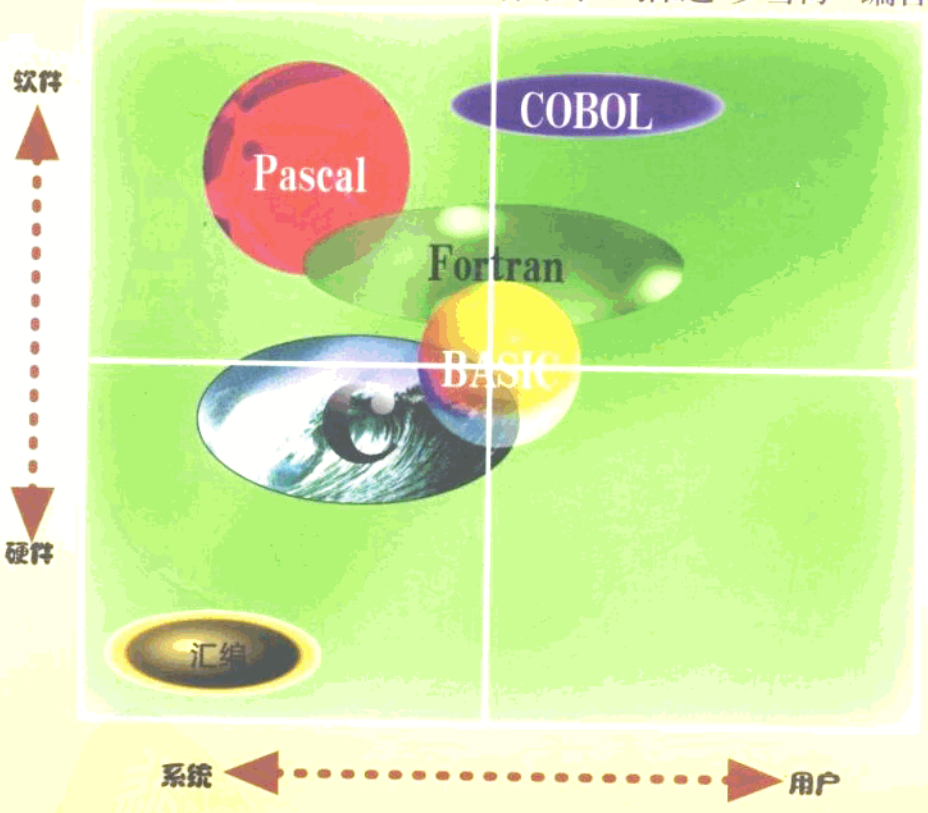
按国际语言教学新经验编写的
半 双 语 系 列 教 材
Semi-Bilingual Series Course

从BASIC跃到Pascal

韩耀军 刘保进 罗雪梅 编著



利用你已有的BASIC知识
帮助你更快地掌握Pascal

北京大学出版社

序

世界上现今一共有多少种程序设计高级语言已经很难说清。早在1984年《科学的美国人》出的计算机软件专辑中就曾说过："程序设计语言加上它们的'方言'为数至少有几百种，甚至可能有几千种。"[①] 11年后，《BYTE》杂志创刊20周年特刊中发表了20篇综述文章，在"程序设计语言发展简史"一文中，列举出从1946到1995年为止，在社会上产生了一定影响的四十几种程序设计语言[②]。这些语言中的一部分传到我国，几经筛汰，有些随着教学和应用逐渐得到普及，其中最常见、最流行的莫过于微机上的BASIC，Pascal，Fortran和C等几种。

这几种高级语言中，BASIC被公认为是易学易用、发展快、变化大、覆盖面广的初学者语言。自80年代初，在美国出现兼具解释和编译功能的结构化BASIC（即我们称之为**第二代BASIC**的True BASIC，Turbo BASIC和Quick BASIC）以来，BASIC在美国成了最受欢迎的程序设计语言[③]。美国人对它的评价是"Low threshold，high ceiling"，其直译是"门槛低，天花板高"，即入门易而潜力大的程序设计语言。在三田典玄先生提供[④]而被我们选作本书封面的图案中，读者可以看到BASIC语言兼顾了软件和硬件、系统和用户四大领域的应用，几乎成了其他任何一种语言无法代替的语言。

Windows风行后，程序设计的难度陡然提高，几乎酿成广大业余程序员的灭顶之灾。在此期间，一批甩开TEXT状态，采用GUI状态，能在Windows（OS/2）环境下运行的BASIC（即被称为**第三代BASIC**的Visual Basic，CA-REALIZER，GFA BASIC和Power BASIC等）恰如雨后春笋，应运而生，于危厄中解救了一大批非软件专业程序员，使他们很快就能在Windows（OS/2）提供的广阔天地中纵横驰骋，开发他们所熟悉专业的应用软件。只有各专业的应用软件极大的丰富了，才会有计算机应用的真正繁荣。

因此，近七八年来，我们一直呼吁社会各界（尤其是教育界）要重视BASIC的推广与应用，抓住Quick BASIC与第三代BASIC有最好的兼容性这一特点，以Quick BASIC为计算机专业与非计算机专业学生的入门语言，通过两代BASIC的衔接，解决我国大中专学生和广大计算机用户从DOS向Windows（OS/2）的过渡这一棘手问题。经过三年实践检验，证实了这一方案的有效性。

但是，这绝不意味着我们主张"一花独放"——用BASIC去贬低或取代其他几种常用语言，绝非如此！

每种经受住时间检验，得到广泛使用的程序设计语言，都有它特定的目标和存在的价值，都有自己的特长和不足。BASIC的特点决定了它是很理想的初学者语言；至于从事繁复的科学和工程计算，则应首推Fortran；结构严谨的算法描述，则属Pascal之所长；而构筑具有良好可移植性的系统，又是C语言的拿手好戏。为了适应多方面的需要，一个好的程序员掌握好几

---

① 科学（中译本）计算机软件专辑1985 No.1 21－30页。
② BYTE Vol.20，Num.9，121－122 PP. Sept. 1995。
③ BASIC仍是最常用的编程语言，计算机世界，1987年8月23日。
④ 《实习C语言》三田典玄，アスキ-出版局，1987。

种汇编和几种高级语言势在必行。这里就发生了先学的BASIC与后学的其他各种高级语言之间的关系问题。

这问题在国内似乎从未有人提出过，翻翻案头上、书店里、书库中各种各样讲授高级语言的教材就会发现，尽管编者不同、出版社不同，教材有新有旧，有详有略，但有一点是共同的：它们都假定自己的读者是计算机程序设计语言的初学者，每本教材都是孤立地讲授自己这种语言，而且都是从零开始的。

人们获取知识的最好方法莫过于充分利用他已有的知识，通过对比进行学习。一个好的BASIC程序员，他的BASIC知识和编程经验，是非常宝贵的，这是他掌握其他程序设计语言的一块很好的基地和跳板。更何况几种常用高级语言，在漫长的发展过程中互相渗透、互相借鉴，逐渐形成了你中有我、我中有你的局面。其中变化最大的是BASIC语言，它最初是从Fortran脱胎而来，以后陆续吸收了其他语言的诸多特点，在程序设计语言种族中，它是一个不折不扣的"混血儿"。国外某些计算机教育专家早已注意到了这点，他们发现已掌握BASIC的人可以很快地学会任何一种(哪怕是很晦涩的)程序设计语言。

Robert J. Traister在他的《从BASIC一步跳到C++》一书的"前言"和"跋"里分别写下了这样两段话：

> 你的BASIC编程知识，在你学习C++和其他任何一种计算机程序设计语言时，是你的无法估价的好帮手[①]。
>
> 如果你会用BASIC编写程序，那么你将很快就能用C++写出同样的程序来。带上你所有的BASIC知识踏上C++的征途吧，你一定会很快适应这种高效编程环境。很快你就会非常乐意并且轻松愉快地用C++这种语言去处理你面临的各种任务，就像你今天用BASIC去处理它们一样[②]。

我们对此深有同感。经过两年多的酝酿筹划，设计并编写了这套系列教材，它们是：

- 从BASIC跃到C
- 从BASIC跃到Pascal
- 从BASIC跃到Fortran

在众多的程序设计语言教材中，这套教材的特点是力求在两种语言(一主，即读者要学的新语言，如Fortran或Pascal或C；一副，即读者已经掌握的BASIC语言)的对比中，更快捷、更深刻地去掌握新语言。两种语言相同之处一笔带过，相似之处加以区别，赢得篇幅和课时去仔细讲授不同语言的独特处，把功夫下在这些地方，以求取得互相对比、相得益彰、事半功倍的效果。

本系列教程的任务：首先是教会读者将他能用BASIC写出的程序，转变成另一种语言(Pascal或Fortran或C)形式。这任务比起教一批对程序设计一无所知的初学者学会一门程序设计语言，显然要容易得多。在此基础上，本系列教材将引导读者进一步考虑：如何发挥这种语言(Pascal或Fortran或C)的特长和优势，用它写出在它专长的领域内功能更强、效率更高的程序来。

---

① Robert J. Traister, Leaping from BASIC to C++, Epilogve 357 PP. AP PROFESSIONAL 1994

② Robert J. Traister, Leaping from BASIC to C++, Preface XVI PP. AP PROFESSIONAL 1994

这套教材彼此间没有横的联系，可根据需要选其中任何一种，也可供已掌握 BASIC 程序设计者自修 Pascal 或 Fortran 或 C 语言之用。这套教材在编写时均采用当前微机上广泛流行的软件，同时考虑了这些软件的升级版本。书中例题均经过上机检验，准确无误。

双语教材甚至半双语(Semi-Bilingual)教材在外语教学中已得到应用，取得了好的效果。本系列教材在计算机程序设计语言的教学中，引入类似的构想，希望也能取得好的效果。

诚挚的盼望广大读者尤其是有志于程序设计语言教学改革的教师，选用本教程作教材，将您在使用中发现的问题、意见和感想写给我们，让我们携手合作，共同来推动并完善这项改进。

编者们怀着崇敬真挚的心情，感谢北京大学出版社的领导及责任编辑段晓青，深深感谢他们在当前科技书出现的困境中，帮助我们将多年的理想变成了现实——这套教材能够摆在读者面前，接受社会实践的检验。

编者们向在美国的杜殿海、曾瑞华、陈涌和在加拿大的陈贺平诸位先生表示感谢，感谢他们不断地提供国外的有关信息、新书和资料。

本系列教程参考了诸多国内外书籍、论文和资料。在此向所有作者表示感谢。

编者们虽已竭尽努力，终因知识和经验所限，教材中难免出现这样那样的错讹，恳请读者指正，不胜感谢！

潘正伯

1998 年 4 月

前　言

BASIC 语言是一种会话式语言，小巧玲珑、操作方便、使用灵活、易学易用，早已被很多人所熟悉和掌握，成为一种深受欢迎的程序设计语言。许多大、中专院校都将 BASIC 语言作为学生的入门语言。而 Pascal 语言则具有崭新的结构化程序设计思想、清晰严谨的程序结构、丰富的数据结构、简明高效的编译程序以及良好的可靠性、可读性和可移植性等特点，是目前最流行的几种程序设计语言之一，并被确定为全国计算机等级考试的一种可选语言。

目前许多 Pascal 方面的书，都是孤立地讲授 Pascal 语言，并且都是从零开始。实际上，对于已经掌握了 BASIC 语言的读者来说，应该并且也可能充分利用自己已有的 BASIC 编程知识，花较少的时间和精力学会多种程序设计语言。我们知道 Pascal 等几种常用高级程序设计语言与 BASIC 语言(本书以 Quick BASIC 4.5 为例)在很多方面都有相同或相似之处。为了已经熟悉 BASIC 语言的人员能在较短时间内掌握 Pascal 语言作者编写了这本书。

本书有下述特点：

1. 在内容的组织上，既照顾到与 BASIC 语言的内容相对应，又保持了 Pascal 语言的内容体系。这不仅便于读者自学，而且也便于用作教材。

2. 本书内容详略得当。凡是 Pascal 语言与 BASIC 语言相同之处，只是简单地带过；与 BASIC 语言相似之处，均加以比较，指出它们的异同，并尽可能给出用两种语言实现的程序实例；而对于 Pascal 语言的一些特殊功能，则用了较大篇幅加以详细介绍。

3. 本书配有大量的例题和习题，这些题目针对性强，由浅入深，由易到难，以帮助读者理解和掌握 Pascal 程序设计的方法和技巧。

4. 全书贯穿了结构化程序设计思想。在一些例题中使用了模块化程序设计、自顶向下、逐步求精的程序设计方法，以体现出结构化程序设计的良好风格。

5. 本书以介绍标准 pascal 为主，但考虑到 Turbo Pascal 在微型机上较为普及，及其自身的特点，凡是 Turbo Pascal 对标准 Pascal 在功能上的扩充，均一一指出，并在第十章专门介绍了 Turbo Pascal 中具有模块化程序设计思想的包含技术、单元等内容，以及 Trubo Pascal 的图形功能。

为便于读者上机练习，本书在附录 A 介绍了 Turbo Pascal 的上机操作。书中所有例题都已在 Turbo Pascal 6.0 版本下调试通过。

本书在编写过程中，得到了山东矿业学院潘正伯教授的热情指导，他认真阅读了本书，并提出了许多宝贵意见，在此表示衷心感谢。

由于作者水平有限，疏漏错误在所难免，恳请读者批评指正。

作　者

1998 年 4 月

目　录

第一章　Pascal 语言的基础知识

1.1　Pascal 语言简介

1.1.1　Pascal 语言的产生和发展

Pascal 语言是由瑞士苏黎世联邦工业大学的 N. wirth 教授，在对 ALOGOL 60 语言进行改进的基础上，本着“简单、有效、可靠”的原则，于六十年代末研制成功的一种结构化程序设计语言。该语言之所以这样命名，是为了纪念世界上第一台机械式加法器的创造者法国数学家 B. Pascal。

在计算机应用的初期，尤其是在高级语言出现以前，人们要花费大量的精力来编制比较简单的程序。在当时的程序设计中，人们的主要目的是要编制出指令条数少、运行速度快、存储单元省的程序。随着计算机的应用及软硬件的不断发展，大型系统程序（如操作系统、数据库系统等）开始出现，这就给程序设计带来了新的问题。一方面，由于程序规模的不断扩大，使程序编制的成本不断提高，而且这种花费了大量财力和人力的程序中，往往蕴含着大量的错误；另一方面，由于程序的复杂性，使得程序中的错误很难发现，即使发现也不易修改，导致程序难以维护，使程序的正确性及软件系统的可靠性越来越难以保证，以至出现了所谓的“软件危机”。于是，人们不得不对程序设计方法进行研究，寻求一种新的、能适应简明性、可靠性和易修改性要求的程序设计方法。荷兰的著名计算机科学家、图灵奖获得者 E. W. Dijkstra 提出了“结构化程序设计”这一方法。为适应这一新的程序设计方法，人们在积极研制相应的程序设计语言。Pascal 语言就是在这种情况下产生的，并被认为是第一个实现结构化程序设计的语言。

Pascal 语言本身也在不断发展。1969 年 N. Wirth 教授成功设计出 Pascal 语言，1971 年经他不断改进，在瑞士的《ETH》杂志上正式发表了 Pascal 语言用户手册，它标志着 Pascal 语言的正式诞生。1974 年 N. Wirth 和 J. Jensen 对 Pascal 语言又作了进一步的修改，发表了 Pascal 语言的修改报告，并将修改报告作为“标准 Pascal 语言”。1980 年国际标准化组织（ISO）发表了 ISO（DP/7185）关于 Pascal 的建议草案。并于 1983 年正式发表了 Pascal 语言国际标准（ISO 7180-1983）。

自从标准 Pascal 语言问世以来，相继推出了标准 Pascal 的各种实现系统，出现了 Pascal 的各种版本。有些版本是参照 1974 年的标准 Pascal 实现的，如 UCSD Pascal，PDP-11 Pascal，OMSI Pascal 等。也有一些版本是参照 1983 年 ISO 的标准 Pascal 实现的，如 MS-Pascal，SVS Pascal，Turbo Pascal 等。其中 Turbo Pascal 是近几年微机上使用很广的一种软件。

Trubo Pascal 语言是美国 Borland 公司推出的用于微型机的一种 Pascal 语言系统。Turbo Pascal 面世以后倍受推崇，成为微机上的必备软件。它除具有标准 Pascal 语言的功能外，还具有用户界面好，编程效率高，查错便捷，编译和运行速度快，数据类型更加丰富，图形功能完善等特点。尤其是它所提供的高性能的、具有“友好用户界面”的操作环境，将编辑、调试、编译、运行、存储等集于一体，使用非常方便。Turbo Pascal 的版本在不断更新，自 1986 年推出 3.0 版本以后，陆续推出 4.0，5.0，5.5，6.0，7.0 等版本，其功能越来越强。

本书以介绍 ISO 标准 Pascal(以下简称为标准 Pascal)语言为主,并适当介绍 Turbo Pascal 的一些特殊功能。本书所有例题均已在 Turbo Pascal 6.0 版本下调试通过。

1.1.2 Pascal 语言的特点

1. 崭新的结构化程序设计语言

所谓结构化程序设计,是为了使程序具有合理结构,以便保证程序的正确性、易维护性以及良好的可读性而规定的一整套程序设计的方法。用结构化程序设计方法设计出来的程序称为结构化程序。要想设计出结构化程序,必须有相应的结构化程序设计语言。结构化程序设计语言是这样一种语言:它的成份反映了结构化程序设计的要求和限制(例如无 GOTO 语句的程序设计),因而便于用它来书写结构化程序,而且以此写出的程序易于保证其具有良好的可读性、可维护性和正确性。Pascal 语言就是最早的一种结构化程序设计语言。Quick BASIC(本书均简称为 QB)语言也是一种结构化程序设计语言。尽管它们都保留着不符合结构化程序设计思想的"GOTO"语句,但不使用这个语句,只用符合结构化程序设计要求的三种基本程序结构,就可完全实现程序的任何复杂的逻辑要求。

2. 严谨的程序结构和程序设计方法

Pascal 语言提供了比较严密的逻辑思维和严谨的程序结构,是教学语言的典范。在程序设计方面比 QB 语言要规范、严格得多。比如:Pascal 语言要求对程序中出现的每一个变量都要加以说明,而在 QB 语言中,尽管也可以对程序中出现的所有变量进行说明,但这种说明并不是强制性的。Pascal 语言这种对变量使用的严格要求,可以在很大程度上避免因乱用变量而引起程序出现错误,而且还可以提高程序设计者的程序设计能力,培养良好的程序设计风格。

3. 丰富的数据类型

Pascal 语言不仅具有与 QB 一样的整型、实型等数据类型,而且还具有枚举、子界等用户定义类型以及集合、记录、文件等构造类型,这些丰富的数据类型,大大增强了 Pascal 的功能,使编程更方便、更简洁。

4. 高效率的递归调用

Pascal 语言中的函数与过程,同 QB 语言一样,既可以调用其它函数和过程,也可以自己调用自己,即所谓的递归调用。Pascal 语言的这一特点,可以使一些复杂问题转化成一系列简单问题得以解决,编程时,只使用少量的语句就可以解决以前要很多语句才能解决的问题,从而大大提高了编程效率。

5. 编译型的高级语言

QB 语言是一种既能用解释方式,又可用编译方式运行的语言,它可以边编程、边运行、边调试,非常适合于初学者进行交互会话式调试,而且对于许多语句(如 PRINT 语句),可以不编程序,而直接使用。Pascal 语言则是一种编译型语言,用它编出的程序无论多么简单必须先经过编译,只有通过编译没有错误才能运行。Pascal 程序一旦编译成可执行文件,便可快速执行,并且可以脱离 Pascal 环境。

6. 自由、方便的程序书写格式

Pascal 语言程序的书写格式没有严格的规定,它允许一行写多个语句,也允许一个语句写在多行上,QB 语言可以在一行写多个语句,但是一个语句必须写在同一行上。Pascal 语言这种自由、方便的程序书写格式,可以使写出的程序紧凑、格式优美、便于阅读。

以上所介绍的 Pascal 语言的六大特点，基本上都是 Pascal 语言的优点，但任何一门语言都不是十全十美的。同样，Pascal 语言也存在一些不足之处。比如：Pascal 语言没有提供可调数组的功能，数组的上界必须是常数，不能是变量，这对熟悉 QB 语言的用户来说，使用起来感到很不方便。又如：在标准 Pascal 语言中，没有提供字符串变量；过程必须包含在主程序中一起编译成一个文件，这对实现程序结构的模块化极为不利。Pascal 语言的一些不足之处，有些已在具体的某一版本中得以完善，例如 Turbo Pascal 就提供了字符串变量，并且利用单元的概念，可以将过程单独编译成一个文件等等。我们相信，随着 Pascal 语言的不断改进和发展，其功能会越来越完善，越来越强。因此，目前国内外许多高校都把 Pascal 语言作为高级程序设计语言的教学内容。

1.2 Pascal 语言的程序结构

1.2.1 引例

为了使读者一开始对 Pascal 程序的结构有一个完整的概念，并便于将 Pascal 程序与 QB 程序的结构进行比较，首先引进一个简单的例子，并分别给出相应的 QB 程序和 Pascal 程序。

【引例】 键盘输入半径的值，然后计算并输出圆的面积。

QB 程序

```
CONST PI=3.1416
DIM R AS SINGLE
DIM S AS DOUBLE
INPUT "R=" ; R
S=PI*R*R
PRINT "S=" ; S
END
```

Pascal 程序

```
PROGRAM BTOP1_0(INPUT, OUTPUT); 程序首部
CONST                          } 常量说明
   PI=3.1416;                  }
VAR                            } 变量说明   } 说明部分
   r, s: REAL;                 }
BEGIN                          }
   WRITE('r=');                }
   READLN(r);                  } 语句部分
   s:=PI*r*r;                  }
   WRITE ('s=', s)             }
END.                           }
```

从上面的两个程序可以看出，Pascal 语言的源程序与 QB 的源程序非常相似，但在总体结构及具体的语句表示上有所不同。现将上面的 Pascal 程序说明如下：

整个 Pascal 源程序可分成三大部分：

第一行为程序的第一部分，称为程序的首部。它包括 Pascal 语言的保留字 PROGRAM，程序的名称 BTOP1_0 以及用圆括号括起来的程序的参数 INPUT 和 OUTPUT。

第二行至第五行是程序的第二部分，称为程序的说明部分。其中 CONST 是 Pascal 用来说明常量的保留字，在其后将 PI 说明为常量 3.1416。VAR 是 Pascal 用来说明变量的保留字，在其后给出了程序中使用的两个变量 r 和 s，并说明它们为实型(REAL)变量。

第六行至第十一行是程序的第三部分称为程序的语句部分，即程序的执行部分。以 BEGIN 开始，以 END. 结束。在 BEGIN 和 END. 之间是程序的执行部分。通过执行 WRITE('r=')语句输出"r="这两个字符；读语句 READLN(r)读入一个实数给 r；赋值语句 s:=PI*r*r 则计算圆的面积存放到 s 中；WRITE('s=', s)语句输出圆的面积。

1.2.2 Pascal 语言的程序结构

从上面对 Pascal 源程序的分析可以看出，一个标准的 Pascal 源程序包括程序的首部、说明和语句三个部分。

1. 程序首部

程序首部的格式如下：

PROGRAM〈程序标识符〉(参数表)

其中，PROGRAM 是 Pascal 的保留字，程序标识符是给该程序取的一个名字，如引例中的 BTOP1_0(它可以不同于用来保存源代码和目标代码的文件名)，参数表中的参数用来表示程序与外界有输入/输出联系，一般为 INPUT 和 OUTPUT，它们是两个标准文件，在微型机中，INPUT 表示标准输入设备——键盘，OUTPUT 表示标准输出设备——显示器。程序首部以“;”为结束标志。

Trubo Pascal 规定，程序首部可以全部省略，也可以只省略参数表。

同 QB 语言程序相比，程序首部是 Pascal 语言程序所特有的一部分内容。

2. 程序说明部分

在 Pascal 程序中，除一些标准常量、标准类型、标准过程和标准函数可以不加说明而直接引用外，其他用到的标号、符号常量、变量、用户定义的数据类型、过程和函数等均需在程序的说明部分加以说明，然后才能在程序中使用。但在 QB 程序中，除符号常量、用户定义的数据类型以及引用其它模块中的过程或库中的过程等，必须在引用之前加以说明外，对于变量说明并不是必须的。

Pascal 程序的说明部分包括标号说明、常量说明、类型定义、变量说明、函数和过程说明五部分内容。但在具体的一个程序中，不一定全部包括以上五项内容，根据程序设计的需要，可以部分或全部省略。如引例中只有常量说明和变量说明两部分。

标准 Pascal 语言规定：每一种说明只能出现一次，且顺序不能改变。而在 Turbo Pascal 中就取消了这两项规定。

3. 语句部分

语句部分是 Pascal 程序的核心部分，这部分内容规定了计算机所完成的一系列操作，因此也称为执行部分。它与 QB 程序相比，有以下几点不同：

① Pascal 程序的语句部分以保留字 BEGIN 开始，以 END 结束。而 QB 程序从结构上并没有严格区分哪一部分为语句部分。事实上，整个 QB 程序均是由语句组成的，如果与 Pascal 程序的语句部分相对应，一般把 QB 程序的输入、处理、输出这三部分内容认为是其语句部分。如引例中的 QB 程序中的第四行至第七行对应 Pascal 程序的语句部分。

② Pascal 程序中的一个语句可以写在多行，一行也可写多个语句，语句与语句之间用分号“;”隔开，但 END 之前的一个语句可以不加。而在 QB 程序中，一个语句只能写在同一行，一行可写多个语句，同一行的语句与语句之间用冒号“:”隔开。

③ Pascal 程序的语句部分以 END 结束，在 END 后加上一个英文句号(即小数点)“.”表示整个程序的结束。在 Pascal 程序中，BEGIN 与 END 并不是语句，它们只作为语句部分的起止标志，相当于一对括号将要执行的若干个语句括起来。而 QB 程序的 END 是一个结束语句，程序运行到 END 语句后，不论后边是否还有语句，均结束程序的运行。

此外，为了便于程序的阅读和理解，同 QB 程序一样，Pascal 语言也允许在程序中使用注

释语句，而且注释语句可以写在程序的任何位置。Pascal 语言的注释语句有以下三种形式：

{〈注释内容〉}

或(*〈注释内容〉*)

或/*〈注释内容〉*/

在 Turbo Pascal 中允许使用前两种形式。

1.2.3 字符集和标识符

以上我们从总体上给出了一个 Pascal 程序的基本结构。作为一种高级语言，与 QB 一样，Pascal 也有自己的一套基本元素。这些基本元素按照语法规则构成 Pascal 语言的各种成份（如常量、变量、表达式、语句等），进而组成 Pascal 程序。Pascal 语言的基本元素同 QB 一样，也包括基本字符集和标识符两大类。

1. 基本字符集

Pascal 语言的基本字符集由字母、数字和专用字符组成。其中字母和数字与 QB 语言完全一样，即大、小写英文字母各 26 个以及 0～9 十个数字。专用字符指在 Pascal 语言中有特殊含义的字符，有些与 QB 相同，有些则不同。现分类列出如下：

① 算术运算符：+　−　*　/

② 关系运算符：=　<　>　<=　>=　<>

③ 赋值运算符：:=

④ 注释符：{　}　(*　*)　/*　*/

⑤ 其它字符：[　]　(　)　.　,　;　:　^　$　#　空格等

2. 标识符

Pascal 语言的标识符与 QB 语言一样，是由字母、数字组成，且以字母打头。在 Turbo Pascal 中还允许使用下划线。Pascal 语言的标识符可分成三类：保留字、标准标识符及用户标识符。

(1) 保留字

随着 Pascal 版本的不断改进，其保留字也在不断增加，像 Turbo Pascal 6.0 的保留字已达 50 多个。现列出标准 Pascal 的 35 个保留字。

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

(2) 标准标识符

标准标识符是 Pascal 语言预先给标准常量、标准类型、标准文件、标准函数及标准过程等

定义的标识符。在QB语言中，把它们都归到保留字中，在Pascal语言中之所以把它们与保留字分开，其根本差别在于：保留字不允许再作为用户标识符，而标准标识符则没有这种限制。尽管如此，建议用户最好不要把标准标识符再定义成用户标识符，以免造成混乱和错误。为了便于读者区分使用，现将标准Pascal的标准标识符列出如下：

- 标准常量：FALSE，TRUE，MAXINT。
- 标准类型：INTEGER，REAL，CHAR，BOOLEAN，TEXT。
- 标准文件：INPUT，OUTPUT。
- 标准函数：ABS，ARCTAN，CHR，COS，EOF，EOLN，EXP，LN，ODD，ORD，PRED，ROUND，SIN，SQR，SQRT，SUCC，TRUNC。
- 标准过程：READ，READLN，WRITELN，WRITE，PAGE，PACK，UNPACK，RESET，REWRITE，NEW，DISPOSE，GET，PUT。

(3) 用户标识符

用户标识符是用户根据程序设计的需要为自己定义的常量、变量、类型、函数、过程以及所编写的程序等取的名字。在Pascal语言中，用户标识符一般应遵循以下规则：

- 不能与保留字同名。这一点与QB语言中变量的取名规则相同。
- 尽量避免与标准标识符同名，以免发生混淆。
- 必须先定义或说明，然后才能在程序中使用。
- 标准Pascal语言只识别标识符的前8个字符，而Turbo Pascal最大允许长度为127个字符，有效字符为63个。

以下是一些合法的用户标识符：

a，b2，book，l3，name

下面是一些非法的用户标识符：

2b　不是以字母开头

end　与保留字相同

ab,cd　出现非法字符(即“，”不能出现在标识符中)

last name　标识符中有空格

此外，为使程序清晰易读，在选用标识符时尽量选取有相应含义的英文单词作为标识符，这样做既便于阅读，又便于记忆。

尽管Pascal语言对所有标识符中的大小写字母不加区分，但为了统一起见，今后凡是保留字与标准标识符中的字母一律用大写，用户定义的标识符中的字母一律用小写(程序名除外)。

1.3　Pascal语言的数据类型

Pascal语言具有丰富的数据类型，按其特点可分为简单类型、构造类型和指针类型三大类，如图1.1所示。

这些数据类型中，有的已经由系统预先定义，有的则在用户需要时，根据类型定义的规则自己定义。系统预先定义的数据类型，称为标准类型，总共有四种(Turbo Pascal有五种)，它们是整型(INTEGER)、实型(REAL)、字符型(CHAR)、布尔型(BOOLEAN)(Turbo Pascal还

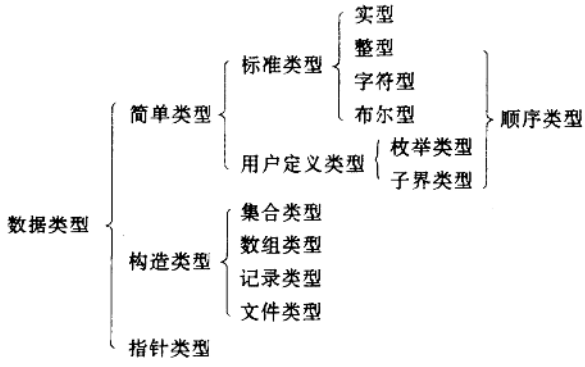


图 1.1 Pascal 数据类型

有字符串型(STRING[n]))。这些类型标识符,可以在定义变量时直接引用,不需要进行类型定义。非标准类型,在程序设计时必须先进行类型定义,再说明其相应的变量。

1.3.1 与 QB 语言相同的数据类型

在 Pascal 语言的数据类型中,与 QB 语言相同的数据类型有整型和实型。

1. 整型

Pascal 语言的整数类型与 QB 语言的整数一样,包括正整数、零和负整数。但由于受字长的限制,对于一个特定的计算机所能表示的整数只是整数集的一个有限子集。QB 语言的整数是指在 $-32768\sim+32767$ 之间的整数。在 Pascal 语言中,用一个标准常数 MAXINT 来表示计算机所能接受的最大整数,它的取值与具体的计算机的字长有关。一般地,若计算机的字长为 W 时,MAXINT $=2^W-1$。对于 16 位字长的机器来说,所能接受的整数范围为 $-32768\sim+32767$,Turbo Pascal 的整数取值就是这个范围(占用两个字节),与 QB 语言是一致的。如 -100,256,0 等都是整型数。Pascal 语言的整型是一种顺序类型。所谓顺序类型,是指该类型的每一个值都对应一个整数序号。整数值的序号就是它本身。

Turbo Pascal 除了具有基本的整数类型(INTEGER)外,还支持以下四种特殊的整数类型(见表 1.1)。

表 1.1 Turbo Pascal 的四种特殊整数类型

名称	类型标识符	数据表示范围	占用字节数
短整型	SHORTINT	$-128\sim127$	1
长整型	LONGINT	$-2147483648\sim2147483647$	4
字节型	BYTE	$0\sim255$	1
字　型	WORD	$0\sim65535$	2

在表 1.1 中,长整型与 QB 中的长整型完全一致。以上四种类型的标识符也都是由系统预先定义的,在变量说明中,可以直接引用。这四种整型数的使用方法和 INTEGER 的使用方法完全相同,可根据需要选用任何一种。

此外,在 Turbo Pascal 中,整型数除了可以用十进制数表示外,还可用十六进制数表示。

其表示方式与QB有所区别，在QB中，用十六进制表示时，开头要写上&H或&h。而在Turbo Pascal中，开头写上$来表示十六进制数。例如：

QB	Turbo Pascal	对应十进制数
&H27	$27	39
&h3B	$3B	59

注意：在QB中，整数除了用十进制、十六进制表示外，还可以用八进制表示。用八进制表示时，在数字前面写上&O或&，但Turbo Pascal没有提供八进制数。

2. 实型

Pascal语言的实型与QB语言中的实型一样，包括正实数、负实数和零。而且也有定点数和浮点数两种表示方法，其书写规则也完全相同。如正数的"+"号可以省略，若有小数点，整数部分或小数部分为零时，都不得省略，因此，.5与5.都是非法的，必须写成0.5和5.0。而3.14，-456.0，1.2E+5等都是合法的实型数。

① 在Pascal语言中，无论用定点数还是用浮点数来表示实数，在计算机中总是用浮点数方法来存放。对于不同的计算机系统，实数的范围不同。如在16位计算机系统中，实数用四个字节表示，其范围（指绝对值）为：1.0E-38～1.0E+38。在Turbo Pascal中，实数（REAL型）占六个字节，取值范围（指绝对值）为：2.9E-39～1.7E+38。

② Turbo Pascal除支持REAL型实数外，还支持下列四种实型（见表1.2）：

表1.2 Turbo Pascal支持的四种实型

名称	类型标识符	数据表示范围	有效位数	占用字节数
单精度型	SINGLE	$1.5\times10^{-45}\sim3.4\times10^{38}$	7～8	4
双精度型	DOUBLE	$5.0\times10^{-324}\sim1.7\times10^{308}$	15～16	8
扩展型	EXTENDED	$3.4\times10^{-4932}\sim1.1\times10^{4932}$	19～20	10
装配型	COMP	$-2^{63}+1\sim2^{63}-1$	19～20	8

在表1.2中，单精度型、双精度型和扩展型的数据表示范围均指的是绝对值。其中单精度型、双精度型与QB的单精度数、双精度数是一致的。以上四种类型的标识符也都是由系统预先定义的，在变量说明中可以直接引用。

但是，处理上述四种实型需要系统有硬件的支持，即需要有数学协处理器，或者系统有仿真程序。如果系统有数学协处理器，只需在Turbo Pascal的集成环境的菜单项Options中选择Compiler，并置该项为8087 80287即可。如果系统不具备数学协处理器，则需要有仿真程序，并在程序中打开编译指令{$N+}（详见附录A）。

1.3.2 Pascal与QB都有，但定义方法不同的数据类型

数组是Pascal与QB都有的一种数据类型，但它们的定义方法不同。在Pascal语言中，数组属于构造类型，有关数组的详细内容将在第七章介绍。另外，Pascal语言的记录类型与QB中的用户定义的数据类型含义相同，其定义方法稍有不同，详细内容见第八章。字符串是Turbo Pascal与QB语言都有的一种数据类型，但它们的定义与使用方法不同，详见第七章。

1.3.3 Pascal 特有的数据类型

Pascal 特有的数据类型包括标准类型中的字符型、布尔型，用户自定义型中的枚举类型、子界类型，构造类型中的集合类型、文件类型及指针类型等。对于 Pascal 特有的数据类型，本节仅介绍标准类型中的字符型与布尔型，其余数据类型将在以后章节中介绍。

1. 字符型

字符型数据类型是由单引号括起来的合法的单个字符组成。如：'a'，表示字母 a，'8' 表示数字 8，' 凵 ' 表示空格字符(注：为清楚起见，在本书中，用"凵"表示空格，但实际上空格在屏幕上什么也不显示)。不同的实现方案对 Pascal 字符集有不同的定义，目前比较广泛使用的是 ASCII 码字符集(如 Turbo Pascal 采用扩展的 ASCII 码字符，共 256 个，其中有些不能显示)。字符类型是一种顺序类型，每一个字符都对应一个相应的序号。对于采用 ASCII 码作为字符集的系统，其字符对应的序号即为该字符的 ASCII 码值。同 QB 语言一样，两个字符大小的比较，事实上就是它们所对应的 ASCII 码进行比较，如字符 'A' 所对应的 ASCII 码为 65，字符 'B' 所对应的 ASCII 码为 66，因此有 'A'＜'B'。

QB 语言没有字符型数据，但有字符串数据。Pascal 的字符型数据类似于 QB 中单个字符组成的字符串。但在 QB 语言中，即使字符串仅由一个字符组成，也必须用双引号括起来，而 Pascal 语言中的字符型数据必须用单引号括起来。在标准 Pascal 中，字符串不是简单数据类型，而是属于构造类型。但 Turbo Pascal 增加了字符串类型，详细内容将在第七章介绍。

2. 布尔型

布尔型的数据只有两个：TRUE(真)和 FALSE(假)。布尔型数据为顺序类型，FALSE 的序号为 0，TRUE 的序号为 1，所以有 FALSE＜TRUE。

1.4 常量和变量

1.4.1 常量

所谓常量是指在程序运行中保持不变的量。Pascal 语言与 QB 语言一样，有两种类型的常量。一种是以整数、实数或字符等形式直接出现的常量，称为直接常量。如：在 Pascal 语言赋值语句

```
S:=3.1416*2*2
```

中的 3.1416 和 2 都是直接常量。另一种是以标识符的形式出现的常量，这个标识符代表某个数字或字符等，称为常量标识符。在 Pascal 程序中，常量标识符有两种：一种是系统预先定义的标准常量标识符，如 MAXINT(在 Turbo Pascal 中 MAXINT＝32767)，TRUE(为布尔真值)，FALSE(为布尔假值)等，在 Trubo Pascal 中，PI 也是系统预先定义的标准常量标识符，它的值为 3.14159265358979，它们在程序中可以直接引用。另一种是用户在 Pascal 程序的说明部分加以定义的常量标识符，一旦定义，在程序的后面凡用该常量的地方，均可用相应的常量标识符代替。如，1.2 节引例中的 PI 就是一个常量标识符。使用常量标识符可以增加程序的可阅读性和可修改性。在 Pascal 语言中，同 QB 一样，可以利用 CONST 来定义常量标识符，其定义的一般格式如下：

- 格式：CONST 〈常量标识符 1〉＝〈常量 1〉；

⋮

〈常量标识符 n〉=〈常量 n〉;

● 功能:用标识符来标识常量。

● 说明:① CONST 是用来定义常量的 Pascal 保留字。

② 等号"="右边的常量一般为直接常量。可以是一个整数、一个字符或字符串,也可以是由已定义的常量标识符与直接常量组成的表达式。如:

```
CONST
biggest=1000;
smallest=-biggest;
sex= 'M';
name='Zhang Shan';
blank=' ';
```

③ 常量标识符必须遵循先定义后使用的原则。

④ 常量标识符不能与程序中其它标识符同名,而且在程序中不能再改变常量标识符的值,即不能再给常量标识符赋新值。这是常量标识符与变量标识符的根本区别。

1.4.2 变量

在程序运行过程中其值可以改变的量称为变量。变量是以标识符来命名的,该标识符称为变量名,这些与 QB 是相同的。二者区别在于:Pascal 语言规定:在程序中出现的每一个变量都要在说明部分加以说明。变量说明的一般格式如下:

● 格式: VAR

〈变量名表1〉:〈类型标识符1〉;

⋮

〈变量名表 n〉:〈类型标识符 n〉;

● 功能:对程序所使用的变量进行类型说明。

● 说明:① VAR 为 Pascal 语言用于说明变量的保留字。

② 变量名表中若有一个以上的变量名时,各变量名之间用",",隔开。

③ 类型标识符可以是标准类型标识符,如:INTEGER(整型),REAL(实型),CHAR(字符型),BOOLEAN(布尔型)等,它们可以在变量说明中直接引用;也可以是用户定义类型标识符,此时,必须先定义数据类型,然后才能使用该数据类型标识符来说明变量(定义数据类型用 TYPE 保留字,详见以后章节)。

④ 若程序中不需要变量说明,则此说明项(连同 VAR 保留字)可以省略。

⑤ 同一变量标识符不允许在同一程序说明中重复定义。

例如,下面的变量说明是正确的:

```
VAR
  i,j,k: INTEGER;        {i,j,k 被定义成整型}
  x,y,z: REAL;           {x,y,z 被定义成实型}
  ch: CHAR;              {ch 被定义成字符型}
  t,f:BOOLEAN;           {t,f 被定义成布尔型}
```

下面的变量说明则是错误的: