• 精华 • 简明 • 实用系列丛书
Made Simple Books

PASCAL

简明教程

(英)Peter McBride 著
况银瓶 译

机械工业出版社

MADE SIMPLE
BOOKS

北京科海培训中心

·精华·简明·实用系列丛书
Made Simple Books

Pascal 简明教程

（英）Peter. McBride 著

况银瓶 译

机械工业出版社

著作权合同登记号:图字 01-98-1773

内 容 提 要

Pascal 是一门结构化编程语言,可产生清晰可读的代码,是学习其他语言的基础。本书利用许多浅显的例子,解释了那些构成程序结构的词,以及主要的过程和函数。

全书共分 10 章,第 1～7 章给出了 Pascal 语言的中心内核,如程序流,数组、字符串和集合,程序的结构,记录,链表与指针等内容,第 8 章介绍了 Turbo Pascal,第 10 章概括了 Pascal 及 Turbo Pascal 的主要特征。

本书内容简练、结构清晰、重点俱全、是各大专院校学生学 Pascal 语言的好教材。

图书在版编目(CIP)数据

Pascal 简明教程/(英)麦克布赖德(McBride. P. K.)著;
况银瓶译. —北京:机械工业出版社,1998. 12
(计算机精华·简明·实用系列丛书)书名原文:Pascal Made Simple
ISBN 7-111-06833-5

Ⅰ. P… Ⅱ. ①麦… ②况… Ⅲ. PASCAL 语言 Ⅳ. TP312

中国版本图书馆 CIP 数据核字(98)第 40131 号

出版人:马九荣 (北京市百万庄大街 22 号 邮政编码 100037)
责任编辑:科培 责任校对:成昊
门头沟胶印厂印刷·新华书店北京发行所发行
1999 年 1 月第 1 版·1999 年 1 月第 1 次印刷
787mm×1092mm 1/16·12.875 印张·251 千字
0001—5000 册
定 价:18.00 元

计算机精华·简明·实用系列丛书

Made Simple Books

丛书序

在计算机新技术迅猛发展、新知识应接不暇、新软件层出不穷的今天，对学会操作电脑的人需要拓宽使用面，让电脑发挥真正的作用。对有经验的用户，跟上潮流的发展而不落伍，也需要不断地更新自己的知识。

我们热情慎重地向广大读者推荐这套布局谋篇上独具匠心、内容精辟、讲叙简明而又实用的系列丛书；这套丛书取自英国非常畅销的“Made Simple”系列中的一部分，透过此系列让人感受到原作者的写作水平，对要介绍的软件、语言和系统的深刻理解，以及作者群体学术的严谨和扎实。“Made Simple”顾名思义使问题简单化，也就是一种将厚书写薄的丛书。每一本书抓住重点，将讲述的对象介绍得简明扼要，通俗易懂。无论是介绍操作系统，还是介绍编程语言或是开发环境，都坚定不移地遵循了这一原则。

- **书不在厚而在于精。**
  此系列，每本小册子不足 200 页，每本书讲解一种软件或是一门编程语言，内容相对独立，使读者能迅速定位自己的需求。每个问题辅以三两实例、言简意赅、点到为止。最为可贵的是作者不搞“大而全”，而是直书精要之处，将基本概念、难点、常用方法及相关技巧一一展示给读者。
- **语言简朴，引导有方。**
  本套丛书是很好的教材，特别是针对初学者，尤为难得。对关键概念，作者舍得花笔墨，用通俗的语言加以阐释；枝节之处，则当删则删，当漏则漏；而且全书都是用简明的图示来表达要点，让读者学得轻松、容易树立信心。
- **实用性（这是许多书称有而最不容易达到的）。**
  此系列丛书的实用性称得上扎扎实实。全书以问题、任务为主线，辅以大量实例而构成。这些例子实用性强，并且这些例子并不单一，往往例子彼此相关，最后可能组成的是一个比较大的程序，或是一个复合技术。这与我们常见到的一些书，往往一个例子表达一个简单的功能，彼此无关，无助于读者构筑自己较复杂的应用程序。此系列对例子的解释也是用图示表达，其中读者可自行修改、替换。

此系列丛书大部分每章末均有习题，书后有习题答案，这些习题也很有特点，它不是简单地复述前面的概念，也不是前面例子的翻版，而比例子更具有创造性、思考和提高的余地和价值。这是很难得的，也是一本好教材的内涵所在。

我们在翻译此系列丛书时，尽可能地聘请有经验、高水平的译者，目的是为了保持原丛书鲜明的风格。翻译了其中 JAVA 和 UNIX 两本书的钟向群先生认为："此套丛书很像一个隽永精品集，读者极易理解，却又回味无穷，科技书籍中有此效果者廖廖"，翻译 Visual Basic 和 Visual C++两书的熊桂喜教授认为："这是一套非常难得的轻松型教材，值得推荐与学习"。

在当今电脑书籍让人眼花缭乱，汗牛充栋的现状下，指导读者发现和正确选择一些好的读本是我们的义务，也是我们的责任，为读者编写诸如此类的教材是我们工作的方向。

希望此系列丛书帮助你开启电脑知识和程序设计的大门，相信读者是好书真正的评判者。欢迎来函来电联系与指正。

欢迎选购：

《C++简明教程》
《VISUAL BASIC 简明教程》
《VISUAL C++简明教程》
《JAVA 简明教程》
《PASCAL 简明教程》
《DELPHI 简明教程》
《UNIX 简明教程》
《Windows NT 简明教程》
《多媒体简明教程》
《硬盘管理简明教程》

科海丛书编译委员会
1998 年 10 月

前　言

Pascal 语言简单易学，非常适合结构化编程，并能产生清晰可读的代码，因此它已成为广大学生最容易掌握的一门编程语言。虽然它很少用于商业软件产品，但 Pascal 为其他语言的学习打下了良好的基础。像 C 和 C++，它们通常是应用程序、操作系统、游戏软件及其他要求速度的程序的语言选择，但它们作为所学的第一语言，则使人难于理解，此时如果你已有学习 Pascal 的经验，那就会大大降低学习 C 语言的难度了。Borland 公司的 Delphi 是当前的流行软件，非常适合数据库和其他商业应用程序的开始，深入了解 Pascal 语言以后，Delphi 将是另外一门你容易学习的语言。

本书并不试图面面俱到地给出这种语言的全部细节。它的主要目标是为这种语言提供一个清晰、简洁的解释——利用许多浅显的例子，来解释那些构成程序结构的词以及主要的过程和函数。

本书所讨论的大部分内容的焦点是 Pascal 语言的中心内核，这也是 Pascal 语言所有版本的公共部分。在本书前 7 章中给出的技术和程序对于任何装有 Pascal 编译器的计算机都是适用的，无论是 PC 还是多用户系统。第 8 章介绍了 Turbo Pascal 的 Windows 版本。第 10 章概括了 Pascal 及 Turbo Pascal 的主要特征。

你必须通过实践才能学会编程，因此在本书前 7 章中每一章的结尾部分给出了相应的练习，练习的答案在第 9 章中给出。

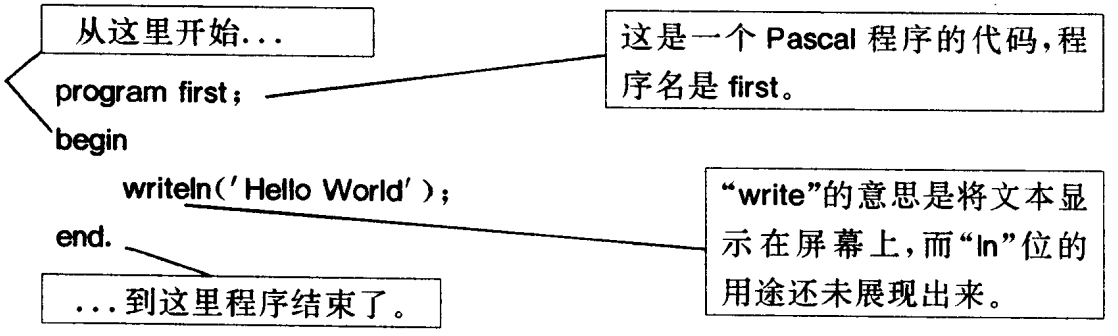
目 录

第 1 章　Pascal 概述

1.1　什么是 Pascal

Pascal 语言大约是在 20 年前由 Nicholas Wirth 开发的，它被设计用来作为学生学习编程的第一门语言，而结果也证明它非常适用于这种目的。Pascal 是最容易被掌握的语言之一，它提倡初学者保持良好的编程习惯。

它之所以容易学主要是因为它有一个相对较小的单词集，而且这些词与一般的英文单词很相近，因而非常便于理解和记忆。甚至在你开始学习这门语言之前，你就能读懂一些 Pascal 程序代码，并在一定程度上明白它的意思。例如，下面有一个用 Pascal 编写的“Hello World”程序，这是一个在语言教程中的典型程序。



Pascal 鼓励好的编程习惯是因为它是一种结构化的编程语言。利用 Pascal 编写的程序很自然地分成块，每块都处理一个不同的操作。它所产生的最终代码不仅易于阅读和调试，而且你可以在同一源程序或其他源程序中重用其中的某些代码块，在那里代码执行同样的操作。

1.1.1　编程与代码

利用一个文本编辑器就可以开始一个 Pascal 程序的产生过程。在一个编辑器中，你写入程序代码，并将它存储为一个文件(带扩展名.PAS)，但计算机并不理解 Pascal 文本——因为语言是人类利用的工具，在你能在一台计算机上运行一个 Pascal 程序之前，必须将其源文本转换成机器语言，而要做到这些必须通过编译器。

编译器的工作分两步。第一步是检查代码的语法错误——错误的拼写、错用的命令、标点符号错误或某项次序错误等等，如果编译器发现存在上述错误，它

将报告出现的错误并停止工作。编程者必须再在编辑器中调出代码进行调试——即去除错误。经修正过的代码再次被传给编译器，也许会又一次被传回编辑器以便进一步调试。只有当程序中的所有错误全被清除时，编译器才开始第二步的工作，将 Pascal 代码转化为机器代码(可执行程序)。

将本节的简单例子输入到机器中，将它存入文件 first. pas。如果你使用一台 PC 上的 Turbo Pascal 版本的编译器，编译命令将编译 first. pas 文件成为一个可执行文件 first. exe。但如果你工作在一台 Unix 系统上，你也许应该用两个命令——第一命令编译你的代码，第二个命令将已编译的代码与系统内部代码块链接从而形成一个可执行文件(所用的命令取决于你的系统)。最终的文件被称为 first。无论哪种情况下你运行此程序，都会看到屏幕上显示：

```
Hello World
```

尽管不是非常激动，但你也许从某处开始了。

提示：

在你进一步学习之前，你必须了解如何使用你的编辑器来编写代码以及编译器的使用。如果你用的是 Turbo Pascal，则可先转到第 8 章去寻找上述有关信息。

1.2 程序的基本结构

最简单地，一个程序必须具有以下形式：

```
program title;
begin
    statement;
    statement;
    statement;
end.
```

重要的注意事项：

- 一个程序总是从保留字 program 开始，program 后紧跟的是程序名，程序名必须是一个独立的词——其中不含空格或标点符号，但其长度一般是任意的。

- 程序的活动部分是从保留字 begin 开始。
- 程序活动部分的结束以保留字 end 为标志。
- program，begin 和 end 全部是具有特殊意义的保留字，它们既可以是大写也可以是小写形式。
- 在结尾处完全结束是基本的观念。
- 一个程序可以包含任何多条语句。
- 一个语句一般自成一行，但也可以被写在多个行中。
- 每个语句之间以分号隔开，而分号一般在行尾。
- 为使程序具有易读性，每行必须以缩进形式来写，这对于比较复杂的程序是非常有利的，因为其中缩进的不同程序有助于程序结构的清晰表现。
- 不论你是否用缩进以及缩进的距离有多远，编译器忽略你的代码中的所有空格与制表符。

Writing 语句行

```
program address;
uses WinCrt;
begin
    writeln('Programming Made Simple');
    writeln('Butterworth-Heinemann');
    writeln('Jordan Hill');
    write('Oxford');
    writeln('OX2 8DP');
end.
```

此语句是 Turbo Pascal 窗口版所必须的。它在程序一开始的地方检查系统的特殊需求。

write 和 writeln，试一下在其他行将 write 替换 writeln，然后观察结果如何。

利用你的编辑器来产生一个地址程序，将以上程序中各行输入其中，而其中的文本字符串可以是你自己喜欢的任何地址——而不必非写我们的地址不可。这个程序介绍了保留字 write 和 writeln 的用法。

write('文本')　它将圆括号内的内容显示在屏幕上，且光标显示在屏幕上最后一个字符后，下一次的字符写入将显示在上一次字符的最后一位字符后。write 可以显示数字、文本或存储在内存中的数据，但任何文本都必须用单引号括起来。

writeln('文本') 同 write 一样显示某些内容在屏幕上，但显示完后，会移动光标在下一行的开始处。

当上述程序编译好并运行时，结果将会显示：

```
Programming Made Simple
Butterworth-Heinemann
Jordan Hill
Oxford OX2 8DP
```

1.3 在屏幕上写

让我们来仔细分析一下“writeln”，这是一个过程——一个命令，对传给它的数据执行一个操作。数据可以是实际的文本或数字，变量或函数(产生相应的文本或数字值)。你可以在一个 writeln 语句中输出多个项，各项中间以逗号隔开。

试一下下面的程序，看一下文本、整数和小数是如何被显示在你机器的屏幕上。

在屏幕上写

```
program writing;
begin
    writeln('This is a text item.','Here is another');
    writeln('One more      ','and the last');
    writeln(1,2,3,4,5);              {integers or whole numbers}
    writeln(12345.678,0.123);        {number with decimal fraction}
end.
```

写在{}括号中的注释被编译器忽略。

最终结果显示为：

```
This is a text item.Here is another
One more      and the last
12345
1.2345678000E+04  1.2300000000E-01
```

- 若在一个 writeln 语句中同时有几项要输出，那么它们会一个接一个被显示，除非在引号中只含空格。

· 小数值——在 Pascal 中被称作实数——被显示为科学记数法的形式。

以上显示的各个方面都可被改变。

1.3.1　writeln 语句的参数

writeln 语句可以通过带参数(也称为变量)来改变它显示数据的方法。writeln 语句中有两个参数,分别用来控制域宽和小数点位置。

1.3.2　域宽参数

它设置被显示数据的空间的宽度。此参数被写在数据项之后,用":"来分隔开。而宽度是以字符数来度量的,数据被安排在这个域的最右边。例如:

```
writeln('Centre stage':46);
```

这个语句将设置"Centre stage"在屏幕上的中心位置,其中最后一个"e"的位置是在第 46 列(在一个 80 列屏幕上)。

1.3.3　小数点位置参数

这个参数仅适用于实数。它将实数显示为更有可读性的"数字.小数"模式,并声明显示的小数点有多少位。

这是 writeln 的第二个参数,因此你必须先设置一个域宽值,然后再设置此参数,如:

```
writeln(123.4567:10:2);
```

则会显示为:

```
123.46
```

并将最后的"6"显示在第 10 列。如果你想数据被安排在左边,或直接显示一个文本,则必须设置域宽为零,例如:

```
writeln('The answer is',42.1834:0:3);
```

其结果为:

```
The answer is 42.183
```

注意:

write 的用法与 writeln 的一样。

域宽的用法

```
program xmastree;
begin
    writeln;
    writeln('*':40);
    writeln('***':41);
    writeln('*****':42);
    writeln('*******':43);
    writeln('*********':44);
    writeln('***********':45);
    writeln('*************':46);
    writeln('| |':41);
    writeln('-----':42);
    writeln('\   /':42);
    writeln('\_/':41);
    writeln;
    writeln('Happy Christmas':47);
end.
```

一个简单的 writeln 过程，它产生一个空白行，此语句通常用来设定显示的垂直位置。

不要忘记每个语句末的分号。

这个程序显示如何利用域宽参数来逐行在屏幕上显示文本。

```
                *
               ***
              *****
             *******
            *********
           ***********
          *************
            | |
            -----
            \   /
             \_/
        Happy Christmas
```