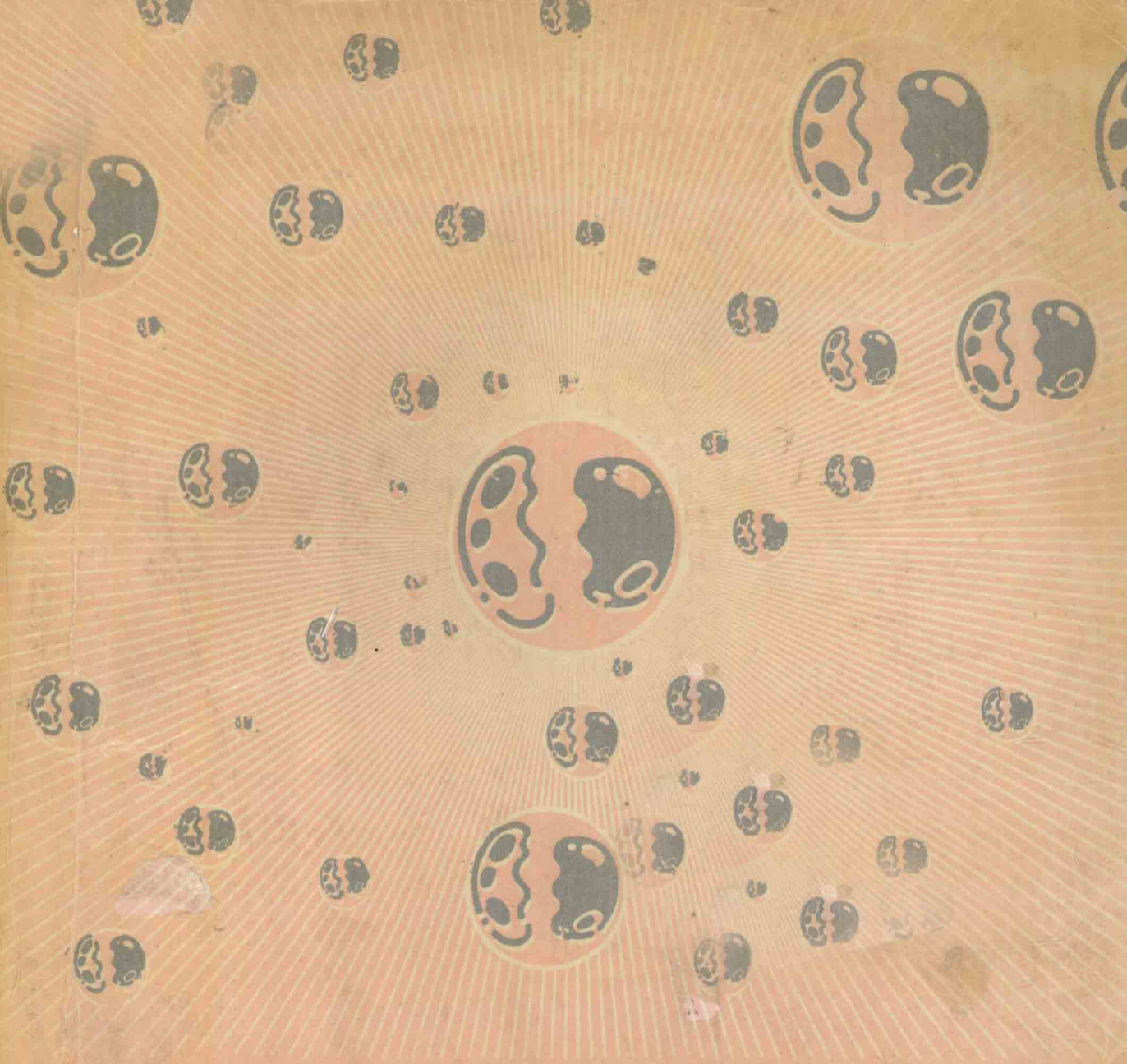


计·算·机·科·学·系·列·教·程

PASCAL 程序设计

陈世鸿　梁意文　编著

武汉大学出版社

PASCAL 程序设计

陈世鸿　梁意文　编著

武汉大学出版社

(鄂)新登字 09 号

图书在版编目(CIP)数据

PASCAL 程序设计/陈世鸿,梁意文编著·—
武汉:武汉大学出版社,1995.1
ISBN 7-307-01899-3

Ⅰ.P…
Ⅱ.①陈… ②梁…
Ⅲ.Pascal 语言—程序设计
Ⅳ.TP312 PA

武汉大学出版社出版
(430072 武昌 珞珈山)
湖北省安陆市印刷厂印刷
新华书店湖北发行所发行
1995 年 1 月第 1 版　1995 年 6 月第 2 次印刷
开本:787×1092　1/16　印张:15.5
字数:359 千字　印数:3001—8000
ISBN 7-307-01899-3/TP·17　定价:12.50 元

内 容 简 介

Pascal 语言是基于程序设计方法论原则设计的一种语言，语法简洁，数据结构和程序结构比较丰富，是培养程序员良好程序设计风格和习惯的最合适语言。

本书向读者系统而全面地介绍了如何用 Pascal 语言来进行程序设计，教会你程序设计的一般方法和程序设计风格。全书共分 12 章，分别论述了程序设计的初步知识、问题求解过程、Pascal 的数据结构和控制结构、程序设计风格、程序测试等内容。

该书不仅使没有学过 Pascal 语言的人能够轻松地学会这种语言，而且通过学习这种语言能掌握自顶向下、逐步求精的结构程序设计方法，通过努力实践，你将成为优秀的程序设计者。

本书内容丰富，深浅兼顾，配有丰富的实例和大量习题，她不仅可以作为大专院校计算机专业的教科书或教学参考书，而且可使从事计算机软件工作的科技人员从中获得程序设计技巧和程序设计风格的良好教益。对于广大社会青年读者，则是一本适宜的自学读本。

前　言

本书是写给教授Pascal程序设计的教师和希望学会用Pascal语言去写程序的人们。无论你是初学者还是具有一定程序设计基础的读者都可从中学会Pascal语言编程的方法和一般程序设计技巧。

70年代以来，Pascal程序设计语言被公认为系统地体现了“结构程序设计原则”，是一种典型的结构化程序设计语言。她以其基本结构简单、数据类型丰富、编写的程序易读、易维护、实现效率高、查错能力强、移植性能好等特点深受程序设计者的偏爱。同时她的丰富数据结构使其不仅适用于数值计算和非数值计算的各种应用软件设计，也适用于计算机系统软件的设计；她的良好分程序结构不仅适合于设计小程序，而且适合于多人协作开发的大型程序，因而她也是教授程序设计方法的最好语言。目前国内外大多数高等院校计算机专业及其它相关专业都选用Pascal语言作为教学的第一程序设计语言，以培养学生的基本程序设计方法和良好的程序设计风格。

本书是作者在多年Pascal教学基础上，对所编讲稿几经修改，并参考十多本国内外有关书籍编写的。本书的宗旨是强调基本方法的训练并注重实用性。全书共列举了120多个例题，尤其是非数值计算方面例题较多，复杂些的程序全都在微型机上用Turbo Pascal调试通过，这对深入理解Pascal语法规则和编程方法极有帮助。尤其是从第5章开始的实例中，我们严格采用由顶向下逐步求精的结构化设计方法与步骤进行程序设计，为读者提供了良好的基本程序设计方法的训练。

本书共分12章。第1章为程序设计概述，着重论述了程序设计语言的发展过程及程序设计的基本步骤，并对Pascal的诞生背景及有关特点作了简单介绍；第2章介绍了Pascal的标准数据类型，并讨论了顺序结构程序设计的方法；第3、4两章分别介绍了Pascal的选择控制结构与循环控制结构，并引进了结构化程序设计思想；第5章讨论函数与过程程序设计，这一章更明确地论述了结构化程序设计思想、方法和步骤。第6章至第11章我们分别讨论了用户自定义类型、数组、集合、记录、文件四种构造数据类型和动态数据类型——指针类型。第12章讨论了有争议的GOTO语句和语法形式化描述的问题。

由于作者学识水平有限，时间仓促，书中不当之处，恳请读者批评指正。

作　者

1994年9月于武汉

目　录

第 1 章　程序设计概述

§1.1　计算机系统简介

现代电子计算机是一个非常复杂的系统，它由硬件和软件两部分组成。计算机硬件由一些执行物理功能的机械和电子线路组合的模块相互连接而成，主要由图 1.1 所示的四个部件组成。由计算机处理的所有信息首先通过输入设备送进计算机存储器，在存储器中的信息由中央处理器处理，处理结果也存储在存储器中，存储器中的信息可以通过输出设备输出。

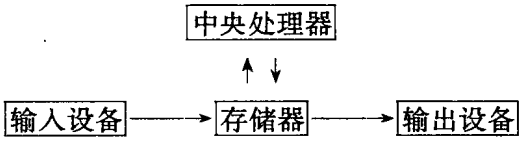


图 1.1　计算机基本部件

中央处理器由控制器和运算器组成，连同存储器一起完成类似人脑的三种功能：分析判断、运算和记忆。输入设备常用的有终端键盘和卡片输入机。常用的输出设备有打印机、绘图仪和显示器。当今发展的多媒体新技术为输入输出设备添加了图象、声音等处理的新成员。

计算机上运行的程序由指令和数据组成，一条机器指令对应于计算机执行的一个基本动作，一个指令序列称为程序。没有计算机硬件，程序无法执行；没有程序，计算机什么动作也没有。所以，计算机系统的硬软件是相辅相成的。

计算机系统大致可分为四个层次（图 1.2 所示），外层利用内层提供

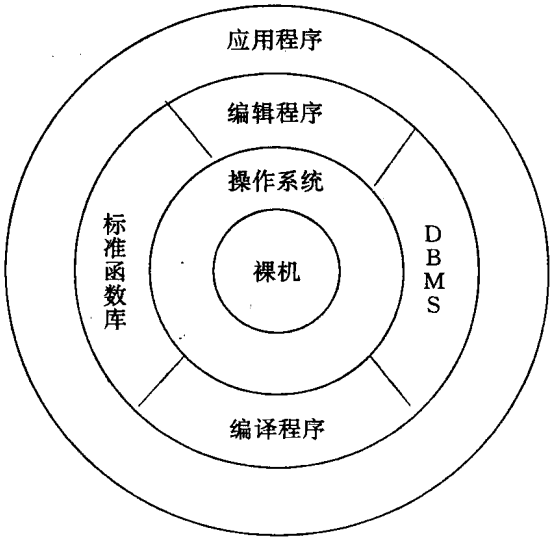


图1.2　计算机系统的逻辑结构

的功能所构成。最低层的裸机是加工数据的物质基础、操作系统担负起组织和管理各种软硬件资源协调、高效地工作；公用程序包括编辑程序、标准函数库（如数学函数库等）、各种语言的编译程序、数据库语言的数据库管理系统等。任何一种高级语言编写的程序都需要相应的编译程序处理后才能执行，所以每台计算机都配置有多种语言编译程序，如 BASIC 编译或解释程序、FORTRAN 编译程序、Pascal 编译程序、C 编译程序等。应用程序是用某种高级语言编写的实用程序，这是用户为求解某个特定问题而编制的。

§1.2 程序设计语言的发展史

众所周知，无论什么人要用计算机求解问题都离不开计算机语言，计算机语言是人与计算机交流信息的最重要工具。

在计算机诞生初期，人们（计算机工作者）只能用各计算机自己的语言－机器语言编制程序。机器语言是一种二进制代码表示的语言（为书写方便，通常写成 8 进制数或 16 进制数），即用 0、1 表示计算机的存储地址（操作数）和操作符。例如，我们假定某个 16 位的计算机（编址为 0000～FFFF）用“01”表示加法运算、“04”表示取数运算、“05”表示存数运算、R 表示累加寄存器，并且假定数“3”、数“5”分别存储在地址 0010，0011 中，其和存储在地址 0012 中，而指令的地址从 0100 开始，则求 SUM＝3＋5 的机器语言程序是

```
0010  0003
0011  0005
0012  [      ]        准备存储 SUM 的值
0100  04  R  0010  3→R
0102  01  R  0011  (R)＋5→R
0104  05  R  0012  (R)→SUM
```

其中右部的(R)表示 R 的内容。事实上，寄存器 R 是可以隐含的，上述指令或写成

```
0100  04  0010
0102  01  0011
0104  05  0012
```

机器语言可以称为计算机工作者的“专用语言”，即不了解具体计算机指令系统和有关结构就无法用机器语言编程。同时读者不难看出用机器语言编写的程序难读、难修改、工作量大，不利于计算机的推广应用。

50 年代初期，计算机工作者采用“符号”表示计算机的存储地址、操作符和操作数。如用字符“＋”表示加法操作，“←”表示取数操用，“→”表示存数操作，则上述指令序列，可用符号表示为

```
←  R  3     3→R
＋  R  5     (R)＋5→R
→  R  SUM  (R)→SUM
```

这种符号指令序列称为汇编语言程序，汇编语言的诞生解决了程序设计中人工存储分配和修改程序的困难，减少了部分编程工作量，但程序员未能完全摆脱了解计算机细节

的困境。受汇编语言的启发，J. Backus 于 1956 年提出了第一个高级程序设计语言 FORTRAN，上述简单例子若用 FORTRAN 语言编程，只需一个语句 SUM＝3＋5（若用 Pascal 来写为 SUM:＝3＋5），它与数学上的等式基本一致（其区别在第 2 章中讲述）。显然，高级程序设计语言的诞生解脱了程序员了解计算机细节的困境，程序员不仅无需关心与程序执行有关的细节，而且大大提高了编程效率（值得注意的是，计算机不能直接执行汇编语言程序和高级语言程序，需要有相应的汇编程序和编译程序的处理才能执行，汇编程序、编译程序不属本书内容，在此不讨论）。

FORTRAN 语言的出现开辟了程序自动化的新纪元，也兴起了软件工作者研究高级程序设计语言的热潮，并且不断追求程序设计语言的表述能力，把语言的表述能力看作是衡量程序设计自动化程度的高低。在这种动力的激发下，自 50 年代末期到 60 年代末期，世界各国的软件工作者研究了上千种高级程序设计语言，最著名并得到广泛应用的几种程序设计语言是 FORTRAN、ALGOL、BASIC、APL、COBOL、PL/I、LISP 等。

60 年代末期，一方面由于提供了强有力的高级程序设计语言，使计算机的应用日益广泛，逐步涉透到各行各业乃至人们的日常生活中，开发的程序越来越庞大和复杂，并且对大型复杂软件的开发又缺乏科学的方法作指导；另一方面，由于软件工作者过分追求语言的表述能力，使语言中引入了不甚合理、难于理解的成分。上述原因往往造成所开发程序的极不可靠性，程序的不可靠性成为 60 年代末期最突出的问题。

为解决上述问题，软件工作者一方面研究程序设计的科学方法和程序正确性的验证方法。（程序设计方法以 E. W. Dijkstra 提出的“结构化程序设计法”为代表，在后续章节的各实例中，我们采用的正是这种设计方法）；另一方面针对计算机语言有些成分的不合理性，对语言各成分逐一进行分析和批判，力求设计出更简明、更适合于写大型程序的高级程序设计语言，典型的代表就是 Pascal 语言。Pascal 这个词是 17 世纪数学家和宗教徒 Blaise Pascal 的姓，它不是由一组词的第一个字母组成的（没有别的含义），故只有第一个字母大写。Pascal 语言是瑞士的 Niklaus Wirth 教授 1971 年提出的，1973 年给出了修订版，1974 年公布了 Pascal 用户手册和修订报告，Wirth 用 Pascal 命名是为了纪念他最早发明实用计算器的贡献。Wirth 建立 Pascal 语言主要有两个目的：

1. 提供一种教学语言，这种教学语言具有其它语言共有的概念，而避免不相容和不必要的细节。
2. 定义一个真正的标准语言，它在任何计算机上容易实现并且造价便宜。

从某种意义上讲，Pascal 是一种程序设计的混合语（但不混乱）或公用语言。它很容易掌握并能为学习其它语言打下良好的基础。凭我们的经验，学了 Pascal 之后，只用一个下午就可学会 BASIC，用一、二周就可学会 FORTRAN、C 等。同时，该语言表述能力强，写出的程序简明、直观、易读易懂，对错误易于检查和修改，对较大的程序系统编写也比较方便。它不仅适合于编制科学计算的程序，也适合于系统软件设计、数据处理软件设计乃至人工智能软件设计。一经问世便引起人们极大的兴趣并倍受重视，这也许是世界各国主要大学均采用它作为讲授程序设计的基本语言工具的真正原因。

当今程序设计语言的发展渗透到各个应用领域，如数据处理语言、分布并行处理语言、图形、图象处理语言、人工智能语言等，当你学好了 Pascal 语言之后，你可自由翱翔在计算机语言世界里！

§1.3 问题求解过程

这是一本关于如何用Pascal语言进行程序设计的书，然而，在介绍Pascal语言各成分之前，我们得强调一下计算机求解问题的过程，即程序设计过程。当我们使用程序设计这个术语时，所考虑的意义可能和别人完全不同。我们在使用计算机程序设计这个术语时意思是指：在计算机上解决一个问题所包含的全部步骤列。然而也常常有人把“程序设计”仅看作编码，即认为对于在计算机上求解问题的最好技术是一拿起铅笔就开始编程序（更有甚者是直接在计算机上编程序），直到写完这个程序为止。然后他就希望能产生一个正确结果，但这样做却毫无把握性。因为任何一种可能的解决办法在进行实际编码之前都应做好大量的准备工作。这种准备工作包含许多步骤，比如确切地定义要达到的目的；消去问题说明中的任何模棱两可的或不确切的东西；确定问题的解法并且用某种方便的记号粗略地概述这种解法的轮廓等。事实上，如果这些准备工作做好了，看起来很重要的编码阶段对任何人都变得相当简单明了了，因为它变成了这样一种工作：用某具体语言的正确语句对问题的求解进行简单机械的翻译。

我们再次告诫读者，在你编码之前应进行大量的准备工作，弄清问题的组织结构并描述你的解决方法。当你学习计算机程序设计时，没有理解这个原则是第一且最大的错误！下面我们更详细地说明一下所包含的步骤。

1. 定义问题

在解决一个问题之前，显然必须了解它的准确意义。但是这个很显然的阶段也常常被程序员忽视和省略，他们往往在所定义的问题还充满二义性并且不精确的情况下，就开始他的写程序代码工作，这是十分不明智的做法。为了建立一个可工作的解法，理清求解问题的含义无疑是非常必要的。

2. 概述解法

除非非常简单的问题，一个问题的求解程序不可能是一个单一的任务，而是由许多有联系的任务组成。在一个大型软件系统中，涉及到许多程序和程序员，于是指定每一任务的职能和任务间的关系和接口就变得极其重要了。我们在本书的过程和函数这一章将详细地讨论这个问题。

3. 选择和描述算法

当我们已指明了求解问题所需要的各种任务和子任务，并且对每个子任务已知道将提供什么信息，希望产生什么结果后，只是解决了“做什么”的工作，选择算法并把算法清晰地描述出来是解决“如何做”的问题。什么是算法，Kunthy教授对计算机科学领域的算法概念是这样定义的：

定义1.1 一个算法是一个有穷规则的集合，其规则规定了解答某个特定类型问题的运算序列。

此外一个算法还有以下五个特性：

（1） 有穷性：算法必须在有穷步内终止；

（2） 确定性：算法的每一步操作都有确切的定义；

(3) 能行性:算法的每一步操作是计算机能行可计算的;

(4) 有 0 个或多个输入;

(5) 至少有一个输出。

根据上述定义,有穷性、确定性、能行性是算法的主要特征。算法可以是已经研制成功的并且在文献上发行过的方法,也可以是我们自己研制和设计的方法。直接把算法的非形式说明作为编写程序的依据并不是好想法。关于算法的具体描述方法将在本书的第 4 章中讨论。

4. 编码

只有精确地定义了问题,组织了一个解决方法,并且画出了算法各步细节草图之后,才可能考虑开始编码。编码即为选择合适的程序设计语言,根据设计好的算法写出机器可处理的程序清单,本书的后续各章就是教会你如何用 Pascal 语言编写程序。

5. 调试

缺乏经验的程序员新手,在程序已经编好和运行时,问题还远远没有解决,还会出现语法和语义方面的各种错误。必须查明和改正所有不可避免的错误,这是很费时且烦闷的工作。

6. 资料整理

实用程序的资料整理也是一个十分重要的工作。资料整理工作是贯穿在整个程序设计过程中的连续工作,第 1 步程序的问题定义、第 3 步的算法描述、第 4 步程序本身都可以作为程序资料的一部分。然而,为了成功地完成程序,必须保证资料的完整性的可用形式,所谓完整性是指在编码和调试修改过程中若发现程序与原算法有不一致处,应随时修改算法,使算法与程序保持一致,还包括程序员使用的各种技术资料和用户使用的用户级资料。

7. 程序维护

实用价值越高的程序维护工作越重要,程序运行几个月甚至几年还可能发现新的错误,由于新功能要求的增加,设备和软件环境的改变或升级,都会发生维护要求,前述所强调的资料完整性和实用性对维护是至关重要的。

在我们介绍 Pascal 语言之前就和读者讨论程序设计的步骤问题,其出发点在于你学习 Pascal 语言的目的是为了用该语言设计计算机求解问题的程序,请你切记程序不是程序设计而只是程序设计的结果,不遵循程序设计的步骤和方法,你永远不会成为一个合格的程序员。

§1.4 Pascal 语言的基本符号

任何一种程序设计语言都有它自己的基本符号集,由这些基本符号按其语法规则构成该语言语句,由合适的语句序列组成程序。Pascal语言的基本符号由以下三部分组成:

1. 基本字符

包括下列三类字符:

(1) 字母：A,B,C…,X,Y,Z;a,b,c,…,x,y,z。按标准 Pascal 规定，一个字母可以是大写，也可以是小写，都被看作是相同的字符，但是在字符串中两者不能互换。

(2) 数字：0,1,2,3,4,5,6,7,8,9

(3) 特殊符号：由运算符、分隔符和保留字三部分组成。

运算符：+ − * / = > < <= >= <> :=

分隔符：() [] {} : . , ; “ ’

这些基本符号集，实际上是 ASCII 字符集(见附录 A)的一个子集。

保留字：由字母拼成的词(字)，它们在 Pascal 语言中有固定含义，不能作为用户定义的标识符使用。保留字有：

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	OTHERWISE

按标准 Pascal 的规定，保留字和标识符的字母均不分大小写，本书中为区别保留字和标识符，保留字采用大写字母，标识符采用小写字母。

2. 标识符

标识符是用来表示程序、常量、变量、过程、函数和类型等名称的符号。标识符必须是以字母开头的字母数字串。标识符的长度不受限制，但受机器存储的限制。因此，在 Pascal 中，通常把标识符前 8 个字符看作为有效字符。也就是说，用来表示不同对象的标识符应当在前八个字符上加以区别。标识符可分为标准标识符和用户定义的标识符。

标准标识符是 Pascal 语言预先为标准函数、标准过程、标准类型、标准常量以及输入、输出标准文件所定义的标识符，如三角函数、字符函数、标准读写过程等，用户在程序中可直接使用而无需定义。下面给出不同类标准标识符的名称，在实例中用到它们时再说明其含义。

标准常量： false true maxint.

标准类型： integer real boolean char text.

标准文件： input output.

标准函数： odd ord pred round sin sqr sqrt succ trunc.

标准过程： read readln write writeln page pack
unpack reset rewrite new dispose get put.

用户定义的标识符是根据标识符的定义方法，常常选用能说明标识符的意义或功能的助记忆的标识符名。用户定义的标识符必须在程序的说明部分加以定义才能在程序中使用。

下列是合法标识符：

```
hours
filename
a
xl
b1234567
```

下列不是合法标识字符：

```
product  on          {在标识符中不能出现空格符}
labc                 {标识符不能以数字开头}
y－m－d              {标识符不能含非字母非数字字符}
AND                  {保留字不能作标识符}
```

值得注意的是，除上述的分隔符外，Pascal 语言中还把空格符、行结束符作为分隔符，即任何两个相邻的保留字、标识符、数或两个相邻的符号之间，或程序中第一个符号之前都必须至少插入一个空格符，语句行之间由行结束符结束。但在一个保留字、标识符或数的内部不允许出现空格或行结束符。

3. 注解

注解是关于程序或程序段含义的文字说明(一串字符)，它在程序中不被执行，仅作说明用。好的程序风格是在程序的相应位置加适当的注解。注解可出现在任意两个语句之间，标准 Pascal 用{…}括号定界注解，而有的 Pascal 系统用/ * … * /定界注解。

§1.5 Pascal 语言的程序结构

一个用 Pascal 语言编制的程序通常由描述求解问题的数据和相应的操作两部分组成，即 Pascal 程序是由字母、数字和特殊符号按照其一套语法规则组成字，由字组成的语句序列。先通过一个简单的 Pascal 程序来初步认识一下完整程序的结构。

```
PROGRAM  example(input,output);{程序首部}
VAR  y,square,cube:integer;{说明部分,变量说明}
BEGIN
    read(y);{语句部分}
    square:=y * y;
    cube:=square * y;
    writeln('square of',y,'is',square);
    write  ('cube of',y,'is',cube)
END.
```

从这个简单例子中可看出，一个 Pascal 程序由程序首部、说明部分和语句部分三部分组成。说明部分与语句部分合起来可称为分程序。

1. 程序首部

程序首部必须以 PROGRAM 开头，它指示程序名(如本例的 example)和标准输入/

输出文件名。标准输入/输出文件名用圆括号括起来，两个文件名之间由逗号隔开。标准输入/输出文件用来表示程序与外界的联系，input 和 output 与具体运行环境有关。在微型计算机环境中，input 通常是键盘，而 output 通常是显示器屏幕或行式打印机。

2. 程序说明部分

程序中的说明(或称定义)部分是对数据的描述。Pascal 语言中除一些标准常量、标准类型、标准过程和标准函数不需在说明部分中定义而可直接使用外，其它用户定义的标识符、常量、类型、变量、函数和过程均要在说明部分加以说明。说明部分由五个部分组成，当你的程序不需要某一部分时可以为空(如上例中只有变量说明)，但说明部分必须按照如下顺序书写：

〈标号说明部分〉
〈常数说明部分〉
〈类型说明部分〉
〈变量说明部分〉
〈过程与函数说明部分〉

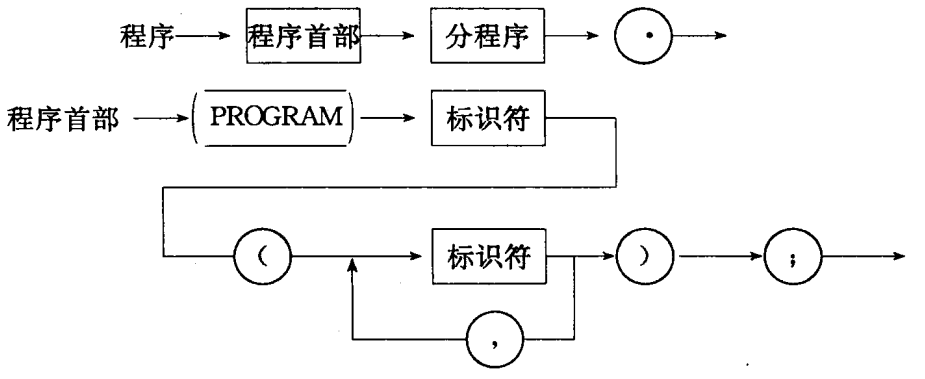
3. 语句部分

程序中语句部分是对实现操作的描述，它是程序的执行部分，由一系列语句组成，每个语句之间用分号(;)隔开。语句部分以保留字 BEGIN 开头，以 END. 结尾，(在 END 后必须加上句号(.)，表示整个程序结束。

如上例程序的目的是读一个数，计算其平方和立方值。程序的第二行变量说明定义了 y、square、cube 三个变量为整型变量，由 BEGIN 与 END. 括起的五个语句为程序的语句部分。第一个语句为读语句，它读入一个数存入变量 y 中；第二个语句把 y 值自乘，其积存储在变量 square 中；第三个语句计算 y 的立方值并存储在变量 cube 中，第四和第五两个语句分别输出 y 的平方值和立方值。注意程序的最末一个语句(第五个语句)和 END. 之间可以无分号(;)，若加分号相当于增加了一个空语句。

现有计算机语言的语法可以用 BNF 范式(Backus Normal Form)或语法图描述。BNF 范式在编译程序原理，形式语言与自动机著作中均有详细论述，本书将以直观的语法图描述 Pascal 的语法，在附录 C.1 中将给出 Pascal 语言的全部语法图。

由上述程序实例，我们可以把 Pascal 语言程序的组成部分描述为图 1.3 所示的语法图。



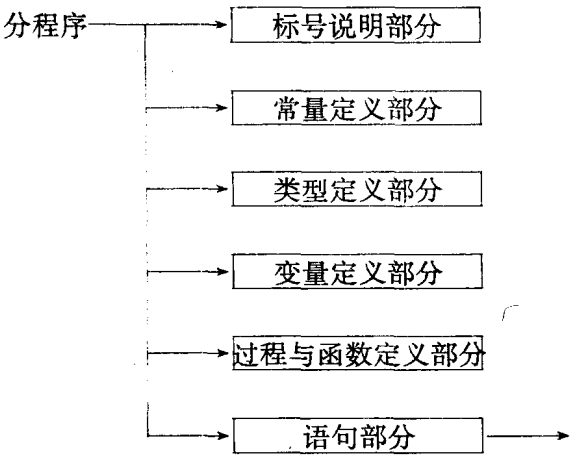


图 1.3　程序结构语法图

从名称“程序”的图形开始，一条通过该图的路径定义了一个语法上正确的程序。语法图给出 Pascal 语言各语法单位的语法规则——即它们必须遵循的格式。图中矩形框内是语言语法单位的名称，该名称引出该语法单位语法图，读者在后续章节中会学到其它语法单位的语法图。椭圆形框书写的是保留字，圆形框是 Pascal 程序中实际书写的符号。请读者熟悉这种语法图的表示法，以便理解其它语法单位的语法图。

§1.6　Pascal 源程序的编译和运行

一旦写好程序，它必须输入计算机，早期用于程序输入的工具是简单行编辑程序（如 edit），它提供了按行的程序输入、修改、插入、删除等功能。而今编辑程序的功能更强，且使用更方便，它可使用户在全屏幕环境下自由的输入修改程序（如 WPS 等）。正如 §1.2 所提及的，计算机不能直接执行用 Pascal 写的源程序，必须由 Pascal 的编译程序处理后才能执行。当用编辑程序把源程序输入到计算机后，调用 Pascal 的编译程序，一般经过两趟扫描，第一趟扫描检查 Pascal 源程序的语法错误，第二趟扫描把源程序翻译成目标程序。计算机执行目标程序时，输入它所要求的数据。程序运行完毕则输出结果。这一过程如图 1.4 所示。

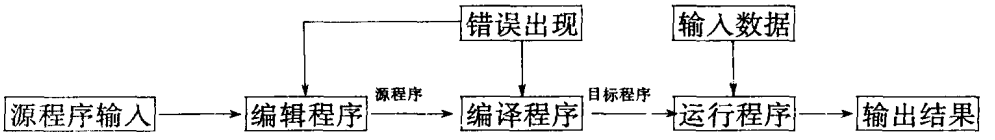


图 1.4　Pascal 源程序编译执行过程

近十年来，采用集成化技术，把编辑、编译、运行、文件管理等支撑 Pascal 程序执行的