

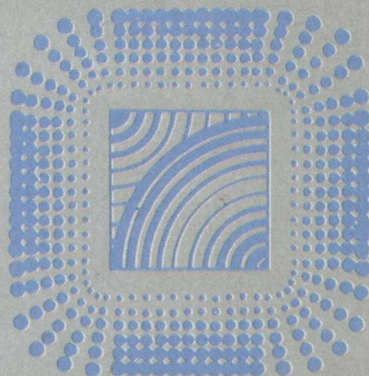


中央广播电视大学教材

PASCAL 语言程序设计

YU YAN CHENG XU
SHE JI

主 编 王秀坤



中央广播电视大学出版社

PASCAL 语言程序设计

王秀坤 主编

中央广播电视大学出版社

(京)新登字 163 号

图书在版编目(CIP)数据

PASCAL语言程序设计/王秀坤主编. —北京:中央广播电视大学出版社, 1994
ISBN 7-304-01045-2

I .P... II .王... III .PASCAL语言-程序设计 IV .TP312
PA

中国版本图书馆CIP数据核字(94)第10064号

PASCAL 语言程序设计

王秀坤 主编

*

中央广播电视大学出版社出版

新华书店总店科技发行所发行

北京密云胶印厂印刷

开本 787×1092 1/16 印张 16.75 千字 383

1994年2月第1版 1994年5月第1次印刷

印数 1—27000

定价 9.40 元

ISBN 7-304-01045-2/TP·52

内 容 简 介

本书系统地介绍了标准 Pascal 语言的基本语法结构、各种数据类型、各种语句的语法及相应功能,同时兼顾了 MS Pascal 和 TURBO Pascal 的一些扩充内容。全书始终贯穿了结构化程序设计思想,注重并强调程序设计风格。内容深入浅出,通俗易懂,例题来源于实践,短小精炼而又具有代表性。

本书作为中央广播电视大学计算机专业的教材,也可作为各类普通高等学校计算机有关专业的教学用书及从事计算机应用人员的参考书。

前 言

Pascal 语言是 70 年代初由瑞士苏黎世联邦技术学院恩·沃思(N. Wirth)教授设计的。它以历史上著名的数学家 B. Pascal 的名字命名,是一种通用的高级程序设计语言。

自 Pascal 语言问世以来,就以其严谨的结构、丰富的数据类型、灵活的表达能力、可移植性好等特点而深受欢迎,已相当广泛地用于科学计算、数据处理和各种系统软件的设计。尤为突出的是它特别适于教学,非常有利于讲述结构化程序设计的思想和方法,有利于使学生养成良好的程序设计风格和习惯,有利于程序设计基本技能方面的训练。正因为如此,绝大部分高等院校的计算机专业在众多的语言中都选择了 Pascal 作为主要的专业基础课。在许多数据结构、程序设计方法学等教材中,也常以 Pascal 为工具来描述算法。

本书是为中央广播电视大学计算机专业编写的教材。其目的是使学生从一开始就养成一个良好的程序设计风格和习惯,为后续课程的学习打下一个坚实的基础。

本着这一目的,本书系统地介绍了标准 Pascal 语言的基本概念、程序结构、各种数据类型、各种语句的语法及相应的功能。同时也适当地介绍了目前国内外广泛使用的 MS Pascal 和 TURBO Pascal。全书始终贯穿了结构化程序设计的思想,注重并强调程序设计的风格,并在 § 4-9 中集中讨论了这方面的有关内容。各章的例题取材于实际应用,大小适量,难易适度,生动易读,典型而又说明问题。凡是完整的程序均已在 IBM 微机上使用 MS Pascal 调试通过。

书中带有“*”的章节和例题为选学内容。

本书由王秀坤主编,其中绪论、四、十、十一章由王秀坤编写;一、二、三、七、九章由陶龙芳编写;五、六、八章由朱嵬编写。

王光兴教授、高复先教授、姚国治副教授、龚杰民教授认真审阅了全稿,并提出了许多宝贵的意见。

本书在编写过程中,得到了中央广播电视大学、大连广播电视大学、大连理工大学计算机系有关领导的大力支持与帮助,在此一并致谢。

由于水平所限,疏漏和不妥之处恳请同行和读者不吝指教。

编 者

1993 年 7 月

目 录

绪论	(1)
一、计算机简介	(1)
二、计算机软件	(3)
三、Pascal 语言简介	(4)
四、元语言符号	(5)
第一章 基本概念	(7)
§ 1-1 Pascal 程序的组成	(7)
§ 1-2 语法图及 Pascal 程序	(7)
§ 1-3 字符和符号	(9)
一、Pascal 语言中的字符	(9)
二、Pascal 语言中的符号	(9)
§ 1-4 数据类型	(13)
一、整数类型	(14)
二、实数类型	(15)
三、字符类型	(16)
四、布尔类型	(17)
§ 1-5 常量定义和变量说明	(19)
一、常量定义	(19)
二、变量说明	(21)
§ 1-6 标准函数	(22)
一、算术函数	(22)
二、转换函数	(23)
三、进退函数	(24)
四、逻辑判断函数	(24)
§ 1-7 Pascal 表达式	(25)
一、算术表达式	(25)
二、字符表达式	(26)
三、布尔表达式	(27)
习题一	(29)
第二章 程序基本结构和 Pascal 基本语句	(31)
§ 2-1 程序基本结构和 N-S 结构流程图	(31)
一、程序的三种基本结构	(31)
二、N-S 结构流程图	(32)
§ 2-2 Pascal 语言的语句	(33)
§ 2-3 赋值语句	(34)
§ 2-4 输入语句	(35)
一、标准过程 READ 语句	(36)

二、标准过程 READLN 语句	(36)
§ 2-5 输出语句	(38)
一、标准过程 WRITE 语句	(38)
二、标准过程 WRITELN 语句	(38)
三、输出格式	(39)
§ 2-6 顺序结构的程序设计	(42)
习题二	(43)
第三章 控制语句	(46)
§ 3-1 选择性语句	(46)
一、IF 语句	(46)
二、CASE 语句	(54)
§ 3-2 重复性语句	(57)
一、WHILE 语句	(58)
二、REPEAT 语句	(60)
三、FOR 语句	(63)
四、多重循环	(68)
§ 3-3 标号说明与 GOTO 语句	(71)
一、标号说明	(71)
二、GOTO 语句	(71)
§ 3-4 初步介绍“自顶向下,逐步求精”的方法	(73)
习题三	(74)
第四章 函数与过程	(82)
§ 4-1 标准函数与标准过程	(82)
一、标准函数	(82)
二、标准过程	(82)
§ 4-2 函数说明和函数调用	(83)
一、函数说明	(84)
二、函数调用	(85)
三、函数说明与函数调用的结合——程序组装	(85)
四、函数与主程序的动态结构关系	(86)
§ 4-3 过程说明和过程调用	(88)
一、过程说明	(89)
二、过程调用	(90)
三、过程说明与过程调用的结合——程序组装	(91)
四、过程与主程序的动态结构关系	(92)
§ 4-4 值参数和变量参数	(94)
一、值参数	(95)
二、变量参数	(96)
§ 4-5 标识符的作用域	(100)
§ 4-6 子程序的嵌套	(103)
§ 4-7 递归调用	(108)

一、内层对外层的调用(包括自身调用)	(109)
二、向前引用	(113)
* § 4-8 函数参数与过程参数	(116)
一、函数参数	(116)
二、过程参数	(118)
§ 4-9 程序设计方法简介	(119)
一、关于程序质量的评价问题	(119)
二、自顶向下逐步求精的方法	(121)
三、程序设计风格的若干问题	(125)
§ 4-10 应用举例	(128)
习题四	(136)
第五章 枚举与子域类型	(138)
§ 5-1 类型定义	(138)
§ 5-2 枚举类型	(139)
一、枚举类型定义	(139)
二、变量说明	(140)
三、枚举类型的运算	(141)
四、枚举类型的函数	(141)
五、枚举类型的值不能读写(输入/输出)	(142)
§ 5-3 子域类型	(145)
一、子域类型的定义	(145)
二、变量说明	(145)
三、子域类型的运算	(146)
四、有关函数	(146)
五、有关读写	(146)
§ 5-4 类型相容与赋值相容	(147)
一、类型相容	(147)
二、赋值相容	(147)
习题五	(148)
第六章 集合类型	(149)
§ 6-1 集合类型的概念	(149)
一、集合类型定义	(149)
二、变量说明	(149)
三、集合类型的值	(150)
四、集合变量的赋值	(150)
五、集合的输入与输出	(150)
§ 6-2 集合运算	(151)
§ 6-3 集合应用举例	(153)
习题六	(155)
第七章 数组类型	(157)
§ 7-1 一维数组	(157)

一、基本概念	(157)
二、应用举例	(159)
§ 7-2 二维数组	(165)
一、基本概念	(165)
二、应用举例	(167)
§ 7-3 紧缩数组	(171)
一、紧缩数组概念	(171)
二、标准过程 PACK 与 UNPACK	(171)
§ 7-4 字符串变量	(174)
* § 7-5 可调数组参数	(176)
习题七	(178)
第八章 记录类型	(181)
§ 8-1 记录概念	(181)
一、记录的定义	(181)
二、变量说明	(182)
三、记录变量的引用	(182)
四、记录变量的运算	(183)
五、记录变量的输入与输出	(183)
§ 8-2 WITH 语句	(184)
* § 8-3 记录变体	(188)
§ 8-4 应用举例	(192)
习题八	(195)
第九章 指针类型	(196)
§ 9-1 指针和动态数据结构	(196)
一、指针类型和指针变量	(196)
二、标准过程 NEW 和 DISPOSE	(197)
三、动态存储单元的引用和动态变量	(197)
四、对指针变量和动态变量的运算	(198)
§ 9-2 指针应用	(200)
一、链表	(200)
* 二、二叉树	(210)
习题九	(214)
第十章 文件类型	(217)
§ 10-1 文件的基本概念	(217)
§ 10-2 顺序文件	(218)
一、文件类型的定义与文件类型变量的说明	(218)
二、文件缓冲变量	(219)
三、文件的基本操作	(219)
§ 10-3 文本文件	(228)
一、文本文件的特点	(228)
二、文本文件的操作	(229)

* § 10-4 直接存取文件 (233)

 一、文件分类 (233)

 二、直接存取文件 (234)

习题十..... (237)

第十一章 Pascal 语言应用举例 (239)

 § 11-1 科学计算举例 (239)

 § 11-2 非数值应用举例 (244)

附录一 ASCII 码表 (251)

附录二 Pascal 语言语法图表 (252)

参考书目..... (257)

绪 论

Pascal 语言是人与计算机进行交流的工具之一。若想利用 Pascal 语言指挥机器完成预定的工作,不仅要掌握 Pascal 语言及程序设计的思想方法,还必须适当地了解有关计算机的基本组成、简要工作流程以及 Pascal 语言与机器的关系等。为此,我们首先介绍这些必备的基本知识。

一、计算机简介

1. 计算机的用途

计算机是一种在人的指挥下能够自动地、高速度地计算的电子机器,它的发明和不断发展是 20 世纪科学技术的卓越成就之一。自 1946 年第一台计算机问世以来,仅仅几十年的时间,就经历了数代更新。它由初期的单纯用于科学计算发展到广泛用于数据处理、情报检索、企事业管理、生产制造、过程控制、数据通信、编辑排版、图形绘制、图象处理、人工智能、医疗卫生、文化教育等各个领域,有力地推动着生产、科学技术与人类文明的发展。正因为如此,它又有电脑之美称,意即其功能可模拟人脑。

2. 计算机的基本组成及工作流程

以用算盘算题为例,参加算题的有纸、笔、算盘和人。算盘是运算部件;纸和笔是用来保存和记录原始数据、中间结果、最终结果的存储部件;人是控制部件,负责指挥与协调各部件有节奏性地工作。

计算机和人用算盘算题所用器具相类似,它是由运算器、存储器和控制器这三个主要部件组成的,我们称它们为主机部分。除了主机部分外,还需有计算机和人进行通信的纽带——输入、输出设备及机器停止工作之后仍能保存数据的外存储器,统称之为辅机部分或外围设备。

(1) 存储器

存储器是存储数据和指令的部件,且所有数据和指令必须以二进制的形式进行存储。

存储器分为内存储器和外存储器。内存储器与运算器直接相连。可以发送代码到运算器参与运算,并能从运算器接收运算结果。外存储器与运算器一般没有直接的联系,但有较大的存储量,并且可以和内存储器成批地进行数据交换。适于长期、大量地保存数据。

衡量存储量大小的基本单位是字节,它是由 8 个二进制位组成的。

存储器的一大特点是:从某一存储单元“取”出数据后,数据并不丢失,“存”入一个新的数据时,该单元原有的内容才会被新的内容取代。因而把数据的“存”和“取”称为“读出”和“写入”才更确切。

(2) 运算器

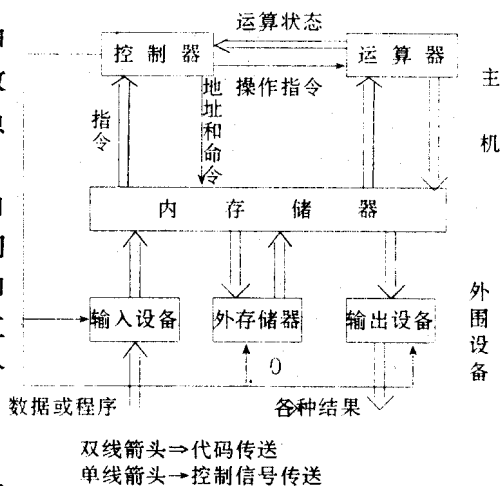


图 1 计算机组成示意图

运算器是完成各种算术运算、逻辑运算和其它操作的部件。运算过程中,运算器在控制器的控制下,不断从内存储器取得数据,经过运算后又把结果送回内存储器保存,这一切是通过执行存放在内存储器中的指令序列来完成的。

(3) 控制器

控制器是控制计算机各个部分协调地进行工作的部件。其主要功能是:

- ①控制运算器和内存储器及有关部分自动协调地执行指令系统规定的运算动作;
- ②控制计算机外围设备和主机之间进行联系。

(4) 输入输出设备

输入输出设备是人和计算机进行通信的纽带。输入设备用来输入程序和原始数据;输出设备用来输出计算结果。常用的输入设备有终端键盘、光笔、鼠标器、图形输入仪等。常用的输出设备有终端屏幕、打印机、绘图仪等。

计算机各组成部分间的联系如图 1 所示。

3. 计算机与二进制

前面谈到,指令和数据都是以二进制代码的形式进行存储,什么是二进制数?这还得从十进制数谈起。

日常生活中,普遍采用十进制计数,十进制计数的特点是使用 10 个数码 0~9,采用“逢十进一”的进位制。如通常所说的某人工资 140 元,工龄 31 年,年龄 53 岁等都是以十进制计数。在有些情况下也经常采用非十进制计数,如,每 12 英寸为 1 英尺(十二进制),每 60 秒为 1 分钟、每 60 分钟为 1 小时(六十进制)等。

计算机为什么采用二进制,只要分析一下计算机的构造及二进制的特点便答案自明。

(1) 二进制仅使用 0 和 1 两个数码,采用“逢二进一”进位制。由于技术上的原因,制造具有两个稳定状态的元件比制造具有多个稳定状态的元件容易得多。因而在计算机内,仅用一个具有两个不同稳定状态的元件来表示一位数码。比如,用一种稳定状态表示数字 0,另一种稳定状态表示数字 1 即可。这正是二进制。

(2) 二进制算术运算比较简单。例如:

加法 $0+0=0, 1+0=1, 0+1=1, 1+1=10$ (逢二进一)

乘法 $0\times 0=0, 0\times 1=0, 1\times 0=0, 1\times 1=1$

在计算机内使用二进制,而在人们日常生活中却广泛使用十进制,它们之间如何相互转换?这正是下面要介绍的内容。

4. 二一十进制之间的转换

(1) 数的分解表示

一个十进制数从小数点起,自右向左的数字依次表示个位、十位、百位、千位……;自左向右的数字依次表示十分之一位、百分之一位、千分之一位……。如十进制数 $2047.85_{(10)}$ 可分解为:

$$2047.85 = \underset{\substack{\uparrow \\ \text{千位}}}{2} \times 10^3 + \underset{\substack{\uparrow \\ \text{百位}}}{0} \times 10^2 + \underset{\substack{\uparrow \\ \text{十位}}}{4} \times 10^1 + \underset{\substack{\uparrow \\ \text{个位}}}{7} \times 10^0 + \underset{\substack{\uparrow \\ \text{十分之一位}}}{8} \times 10^{-1} + \underset{\substack{\uparrow \\ \text{百分之一位}}}{5} \times 10^{-2}$$

同理,一个二进制数从小数点起,自右向左的数字依次表示 2^0 位、 2^1 位、 2^2 位、 2^3 位;自左向右的数字依次表示 2^{-1} 位、 2^{-2} 位、 2^{-3} 位。如: $11010.101_{(2)} = 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}$

(2)二—十进制转换

将一个二进制的数转换为十进制数是非常容易的,观察上式,只要将等式右边的各项按照十进制的计算方法求出其值,再相加,其结果就是相应的十进制数。因为:

$$1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} = 16 + 8 + 0 + 2 + 0 + 0.5 + 0.125 = 26.625$$

所以有:

$$11010.101_{(2)} = 26.625_{(10)}$$

(3)十—二进制转换

将一个十进制的整数转换为二进制的数也是非常容易的,可采用除二取余法。即将十进制数除以2得到一个商和余数,再将商除以2,又得一新的商和余数,如此继续下去,直到商等于零为止。然后,将所得的各项余数按先后顺序自右向左排列,就是所求的二进制数。

例1 将 $26_{(10)}$ 转换成二进制数。

采用除2取余法有:

2	26		右
2	13	余 0	
2	6	余 1	
2	3	余 0	
2	1	余 1	↓
	0	余 1	左

再按所得余数的先后次序自右向左排列得11010。这就是 $26_{(10)}$ 所对应的二进制数,即:
 $26_{(10)} = 11010_{(2)}$ 。

十进制纯小数转换成二进制小数可用乘2取整法。即每次用2乘纯小数部分,将所得整数(0或1)依次自左向右排列,直到满足所要求的精度或纯小数部分等于零为止,即得到所求的二进制小数。

例2 将 $0.625_{(10)}$ 转换成二进制数。

采用乘2取整法有:

	纯小数部分	整数部分
$0.625 \times 2 = 1.25$	0.25	1
$0.25 \times 2 = 0.5$	0.5	0
$0.5 \times 2 = 1$	0	1

于是 $0.625_{(10)} = 0.101_{(2)}$

例3 将 26.625 转换成二进制数。

由例1及例2可得:

$$26.625_{(10)} = 26_{(10)} + 0.625_{(10)} = 11010_{(2)} + 0.101_{(2)} = 11011.101_{(2)}$$

通常这种转换都是事先编制一个程序,由机器执行相应程序自动完成。机器与用户接口均为人们习惯的十进制数。

二、计算机软件

软件种类很多,大致可分为三个层次,如图2所示,上层利用下层所提供的功能而构成。

1. 操作系统(最下层)担负起组织和管理各种软、硬件资源,使之协调地、高效地对数据进行处理,使用户较方便地完成各项任务。

2. 公用程序(第2层)包括各种编辑软件、工具类程序等。

(1)翻译程序(第2层)

计算机能直接识别的只有机器语言,它与自然语言相差甚远,不易掌握。用机器语言编程序难,修改维护更难,因而人们陆续研制了一些比较接近自然语言的适用于解决不同领域问题的各种计算机高级语言。如:适于系统软件开发的C、适于科学计算的FORTRAN、适于商业应用的COBOL、适于逻辑推理的PROLOG、适于初学的BASIC、适于教学及系统软件开发的Pascal等等。高级语言的种类繁多,这些语言机器都不能直接识别,均需要有各自的翻译官——翻译程序。比如,Pascal翻译程序负责把用Pascal语言编写的源程序翻译成等效的机器语言,即所谓目标程序。对目标程序的执行,就实现了源程序中所描述的计算工作。

(2)数据库管理系统

数据库管理系统是60年代末产生的一种能对大量数据进行集中统一管理的软件,它有着十分广泛的应用。

第2层的软件还有很多,如AUTOCAD可用于辅助设计、各种表处理软件、各种排版软件等。这层软件越丰富,计算机的能力也就越强。

3. 应用程序(第3层)指的是用各种高级语言为科学计算、辅助教育、工业、商业、经济管理、图象处理、图形设计等领域编写的实用程序。

从以上的介绍我们可以看出,硬件必须靠软件进行管理,而软件必须靠硬件支持,二者互为依存、互相作用。

三、Pascal语言简介

1. 产生背景

60年代中期,曾一度出现了软件危机。其原因是软件系统越来越大,错误越来越多,软件产品的交付日期越来越得不到保证。人们逐渐认识到,构造大型软件是一项尚未掌握的领域。在1968年召开的NATO会议上,提出了“软件工程”的概念,强调把软件开发作为工程来对待,要有严格的组织和开发规范,以避免产生软件危机。

从方法论方面,荷兰计算机学家Turing奖获得者E. W. Dijkstra提出了“结构程序设计”思想,为的是使程序具有合理的结构,易读、易维护。

在这样的历史背景下,N. Wirth教授于60年代末在瑞士苏黎世联邦工业大学设计了Pascal语言,1970年实现了第一个Pascal编译程序,翌年公布了Pascal语言的最初版本。1974年发表了Pascal用户手册和修改报告,1982年国际标准化组织(ISO)对它进行了标准化工作。我国已于1987年公布了与之等效的国家标准,这就是我们所说的标准Pascal。

2. Pascal语言的特点及用途

(1)结构性好。它提供的语言设施体现了结构程序设计原则,程序易读、易维护、易移植。适于多人分工协作编程。

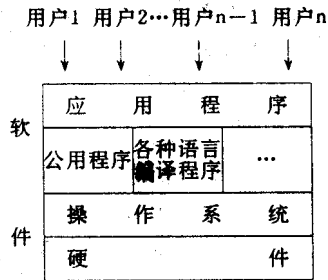


图2 计算机软硬件系统示意图

(2)内容丰富,表达力强。它提供了枚举、子域、数组、记录、集合、文件、指针等众多的用户自定义数据类型,能够用来描述复杂的数据对象,便于书写系统程序和应用程序。

(3)编译简单,目标代码效率高。

(4)有简明通用的语句,可提高应用程序员的软件生产率。

(5)适用面广,可用于科学计算、程序设计教学、数据处理等。

3. Pascal 语言的发展

由于 Pascal 语言具有自己的特色,而且自产生以来深受好评,应用也越来越广泛,因而也就必然地促进了它的不断发展。目前,在 IBM PC 机上广泛流行的是 MS Pascal 和 TURBO Pascal,它们都在标准 Pascal 的基础上作了少量修改与某些扩充,但仍不失结构性好的风格,这些修改与扩充都非常有益于用户的编程。通过 Pascal 版本的不断升级,我们深信,功能更强的 Pascal 版本将会不断产生。

四、元语言符号

任何程序设计语言均有其严格的语法规则。语言的使用者和语言的实现者都是非常关注形式的语法规则的,因为有了它我们便可机械地识别一个程序是否合法。用于描述语法规则的语言,我们称之为元语言,它有一套特定符号,称之为元语言符号,常见的元语言有两种,一种是语法图;另一种是巴克斯—诺尔范式(BNF)。

本书对 Pascal 的每一语法实体均采用语法图进行描述,为加深读者理解,个别语法实体也同时给出了类似于 BNF 的描述。

1. 语法图中采用的符号及其画法

(1)方框 用于表示语法实体,方框中写上语法实体的名字。语法实体是需要进一步再定义的,按形式语言学的术语叫做非终结符。

(2)端圆框或圆框 用来表示语言的基本词汇,如保留字、运算符、分隔符等,它们是不需要再定义的,按形式语言学的术语叫做终结符。

(3)箭头 用来表示语法路径。

如:我们可用图 3 的语法图来描述什么是正整数,什么是数字。

显然,由图 3(a)我们知道正整数是由数字组成的,但数字被框在方框内,它不是非终结符,若知它到底是什么,我们还必须再查看有关数字的语法图。由图 3(b)可知,数字可以是 0,或 1,或 2,...或 9。由此可知,正整数是由 0~9 这 10 个数字任意排列组合而成。如:635 便是一个合法的正整数。

Pascal 语言的全部语法图均在附录一中详细给出。

2. 巴克斯—诺尔范式(BNF)的主要符号及写法

(1)尖括号< > 用来表示一个语法实体,其作用同语法图中的方框。

(2)方括号[] 用来表示方括号中的内容是可选的,即这项内容有、无均符合语法。

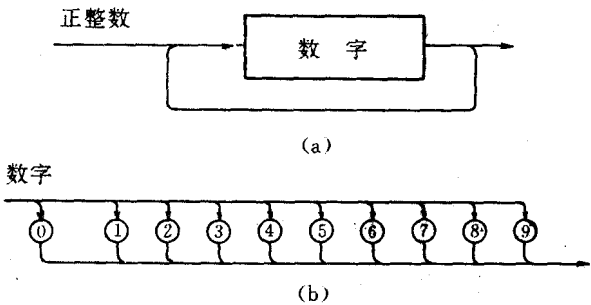


图 3 正整数与数字的语法图

(3) 竖线 | 表示或者的意思。

(4) $::=$ 该符号的左面必须是一个用尖括号括起的语法实体,右面是对该语法实体的描述。

我们可用巴克斯—诺尔范式来描述什么是正整数。

$\langle \text{正整数} \rangle ::= \langle \text{数字} \rangle | \langle \text{正整数} \rangle \langle \text{数字} \rangle$

$\langle \text{数字} \rangle ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9$

语法图看起来比巴克斯—诺尔范式更直观、更易理解,但后者简洁、严谨,便于语法理论研究,二者无本质上的差别。很多高级语言的语法也常用 BNF 来描述。

第一章 基本概念

§ 1-1 Pascal 程序的组成

在系统学习 Pascal 语言程序设计之前,先通过一个简单的 Pascal 程序来认识程序的基本组成。

例 1-1-1 计算半径为 r 的圆的面积 s 。已知计算公式为: $s = \pi r^2$ 。

```
PROGRAM AreaOfCircle(INPUT,OUTPUT);           程序首部

CONST      pi=3.14159;           常量定义
VAR        s,r:REAL;           变量说明      } 说明部分
BEGIN
    READLN(r);                  {输入 r 值}
    s:=pi*SQR(r);
    WRITELN('s=',s)             {输出 s 值}
END.
```

} 程序体

从上面的例子不难看出:

1. 一个 Pascal 程序是由程序首部和程序体两部分组成的。程序首部以分号结束,程序体以圆点结束。

2. 程序首部包括三项内容,按照书写顺序,它们是:

(1) PROGRAM Pascal 程序标志,写在程序的开头。

(2) 程序名 由程序设计者依据“标识符”规则(详见本章 § 1-3)自行定义。

(3) 参数表 即本程序所用到的文件列表。写在程序名后面的括号内,文件之间用逗号分开。

3. 程序体是程序的主体,也称“分程序”。程序体由说明部分和语句部分组成。有时说明部分的某些内容可以全部或部分地缺省。说明部分用来描述程序中所用到的数据的属性;语句部分是程序的执行部分,用来描述程序所执行的算法和操作。

§ 1-2 语法图及 Pascal 程序

Pascal 语言的语法规则可以用语法图进行描述。例如,图 1-2-1 所示的语法图描述了程序的组成。

图中最左边的文字“程序”是被定义的对象,它是由“程序首部”、后面跟分号“;”、“程序体”和圆点“.”组成的。图中矩形框围起来的内容在别处另行定义,而端圆框(此图内为圆框)围起来的内容是必须出现的实际字符,它们是不必进一步定义的对象。图中的箭头只表示先后次序,不是语法规定的内容。