

中等专业学校计算机专业教材

PASCAL 程序设计实践

陈卫卫 王庆瑞 康凯 编著

馆

科学出版社

458588

中等专业学校计算机专业教材

PASCAL 程序设计实践

陈卫卫 王庆瑞 廉 凯 编 著



458588

科学出版社

2000

内 容 简 介

本书用生动实用的示例、简明通俗的语言,系统讲述了PASCAL的基本语法概念和PASCAL程序设计技术。主要内容包括PASCAL程序基本概念、简单程序设计、PASCAL的控制结构、数组类型、过程和函数、其它自定义类型、指针和链表、文件类型等。本书以基本程序结构作为重点,以训练程序设计能力为主要目的。

本书主要用作中等专业学校计算机系列课程的教材,也可作为广大电脑爱好者学习程序设计方法的参考书。

图书在版编目(CIP)数据

PASCAL 程序设计实践/陈卫卫等 编著. -北京:科学出版社, 2000

(中等专业学校计算机专业教材)

ISBN 7-03-007379-7

I. P... II. 陈... III. PASCAL 语言-程序设计-专业学校-教材

IV. TP312

中国版本图书馆 CIP 数据核字 (1999) 第 31702 号

科学出版社出版

北京东黄城根北街16号

邮政编码:100717

北京双青印刷厂印刷

新华书店北京发行所发行 各地新华书店经售

2000年1月第 一 版 * 开本: 787×1092 1/16

2000年1月第一次印刷 印张: 11

印数: 1-3 500 字数: 243 000

定价: 16.00 元

(如有印装质量问题,我社负责调换(环伟))

中等专业学校计算机专业教材

编 委 会

主 编 江起中

副 主 编 谢希仁 王元元 肖经建 徐一冰

责任副主编 王元元

编 委 江起中 谢希仁 王元元 肖经建

徐一冰 贺 乐 李秀琴 张正杰

程自强 贺雅娟 王庆瑞 周华英

王景玉 张锦涛

出版说明

在计算机科学技术飞速发展、广泛应用、深入普及的今天，计算机科学技术图书的出版发行轰轰烈烈、规模空前。但是，在浩瀚的计算机出版物中，人们很难寻觅到适合当前中等专业技术学校使用的计算机专业及专业基础教材，更难发现适合具有高中文化程度的计算机爱好者、“发烧友”系统学习计算机基础知识的图书。因此，我们组织了部分具有丰富中等专业技术教育经验的优秀教师和部分计算机技术专家，编写了这套中等专业学校计算机专业教材。

“中等专业学校计算机专业教材”大致可以分为以下三个模块：

(1) 专业基础知识模块，包括：

- 计算机应用基础
- 计算机数学初步
- 模拟和数字电路基础
- 计算机英语阅读
- 数据结构与算法入门

(2) 专业知识模块，包括：

- PASCAL 程序设计实践
- 数据库技术基础
- 计算机操作系统基础
- 数据通信基础
- C 语言程序设计
- 微型计算机原理入门
- 计算机网络技术基础
- 多媒体技术基础

(3) 实用技术模块，包括：

- 视窗系统及办公应用软件
- 微型计算机系统维护与故障诊断
- 因特网应用入门
- 计算机绘图

我们衷心感谢南京市中等专业（走读）学校的教师在教材编撰过程中所给予的大力支持和关心指导，衷心感谢中国人民解放军理工大学计算机与指挥自动化学院专家、教授们的精心组织和辛勤工作，衷心感谢南京同创计算机学校各级领导和老师们，感谢他们的通力协作和在部分书稿试用阶段的卓有成效的实践。

中等专业学校计算机专业教材

编委会

1999年7月

前 言

程序设计语言 PASCAL 是一种非常优秀的教学语言,其语法简单,易教易学,使用 PASCAL 语言从事计算机程序设计教学,能使學生尽快地掌握程序设计的一般方法和技术。PASCAL 特别适用于算法描述,用 PASCAL 编写的程序具有结构良好、框架优美、易读易交流的特点,所以 PASCAL 语言是训练结构化程序设计方法的首选语言。

本书旨在青少年读者群体中普及和推广计算机程序设计知识,培养广大电脑爱好者的程序设计兴趣,提高他们的程序设计能力,使他们对计算机程序设计原理有进一步的理解,学会如何“指挥”计算机为人类服务,使计算机做人们的好帮手。

本书力图用简朴的语言阐述 PASCAL 的基本概念,即便对某些严格的语法规则,也尽量用实例逐条叙述,避免枯燥乏味、长篇大论,以满足广大青少年读者的需要。

作者精心挑选的一些例题既生动有趣,又有一定的实用性,在编制示例程序时,特别注意程序结构和程序设计方法,希望读者细心体味。

本书共分八章,第一章介绍 PASCAL 的基本概念。第二章介绍仅使用赋值语句和输入输出语句的简单程序设计。第三章介绍如何使用条件语句、循环语句等设计带有控制结构的较为复杂的程序。第四章介绍数组的使用方法。第五章介绍过程和函数设计方法。第六章介绍枚举、子界、集合、记录等自定义类型的用法。第七章介绍链表和指针的用法。第八章介绍文件操作方法。

前三章是 PASCAL 最基本的内容,尤其是表达式、标准函数、简单数据类型和条件语句、循环语句等基本语句,以及输入输出格式应当作为教学重点。第四章和第五章是全书的另一个教学重点,应当花费比较多的精力和时间。在课时比较充足时,第六章中的集合类型、记录类型和第七章的链表也应列为教学内容,其中的部分内容(如链表)甚至也可以作为重点内容。

本书用作中等学校教材时,讲授本书全部内容,大约需要 100 课时(包括上机课时);完成前七章内容的教学,不应少于 80 课时;即使选择最低标准,也应完成前四章或前五章内容的教学,且不应少于 60 课时。书中带“*”的章节是选讲或略讲内容。

本书也可作为普通大专学生教材,讲授内容可适当增加,课时数也可以适当减少。

上机编程练习是学习程序设计语言最好的、必不可少的环节。在计划课时内,上机课时与理论课时比例不小于 1:2。根据现有条件,也可将此比例提高为 2:3。“少讲多练”是学习程序设计的捷径。

本书第三章由康凯编写,其余各章由陈卫卫编写。王庆瑞设计了全书的编写大纲,并对各章的初稿作了认真细致的修改,加配了部分例题。康凯还为前几章选配了部分习题。

由于成稿时间较短,难免有不当甚至错误之处,由衷地希望读者提出宝贵意见。

作 者

1999 年 7 月

目 录

第一章 PASCAL 程序基本概念	(1)
1.1 PASCAL 程序简介	(1)
1.1.1 程序与 PASCAL 语言的基本概念	(1)
1.1.2 PASCAL 程序结构及流程图	(2)
1.1.3 PASCAL 程序示例	(4)
1.2 PASCAL 的基本语法成分	(5)
1.2.1 词汇表	(5)
1.2.2 常量和变量	(7)
1.2.3 数据和数据类型	(9)
习题一	(14)
第二章 简单程序设计	(16)
2.1 赋值语句和表达式	(16)
2.1.1 赋值语句	(16)
2.1.2 表达式和运算符	(16)
2.1.3 标准函数	(19)
2.2 输入输出语句	(21)
2.2.1 输入语句	(22)
2.2.2 输出语句	(26)
习题二	(31)
第三章 控制结构	(35)
3.1 if 语句和复合语句	(35)
3.1.1 if 语句	(35)
3.1.2 复合语句	(37)
3.1.3 if 语句的嵌套	(38)
3.1.4 综合示例	(41)
3.2* goto 语句	(42)
3.3* case 语句	(44)
3.4 while 语句	(47)
3.5 for 语句	(49)
3.6 repeat 语句	(53)
3.7 多重循环	(55)
习题三	(57)
第四章 数组类型	(63)
4.1 一维数组	(63)
4.1.1 定义方式	(63)

4.1.2 下标表达式和下标变量.....	(64)
4.2 二维数组.....	(69)
4.3 字符数组 (字符串)	(74)
4.4 综合示例.....	(77)
习题四	(82)
第五章 过程和函数	(84)
5.1 过程的定义.....	(84)
5.1.1 过程说明.....	(85)
5.1.2 形参表.....	(86)
5.2 过程调用和参数传递.....	(87)
5.2.1 过程调用.....	(87)
5.2.2 参数传递.....	(88)
5.3 标识符作用域.....	(91)
5.3.1 全局量和局部量.....	(91)
5.3.2 参数的选择.....	(93)
5.4 函数.....	(95)
5.4.1 函数说明.....	(95)
5.4.2 函数调用.....	(96)
5.4.3 过程与函数的区别.....	(98)
5.5 递归程序设计.....	(99)
5.6 综合示例	(101)
习题五.....	(106)
第六章 其它自定义类型.....	(110)
6.1 类型说明	(110)
6.2 子界类型和枚举类型	(111)
6.2.1 子界类型	(111)
6.2.2 枚举类型	(112)
6.3 集合类型	(115)
6.3.1 集合的类型定义	(115)
6.3.2 集合的运算	(116)
6.3.3 集合应用示例	(118)
6.4 记录类型	(121)
6.4.1 记录的定义	(121)
6.4.2 记录的引用	(122)
6.4.3* with 语句	(125)
6.5* 类型的相容性	(126)
习题六.....	(128)
第七章 指针和链表.....	(131)
7.1 指针和指针类型	(131)
7.2 静态变量与动态变量	(132)

7.3 动态变量的引用	(133)
7.3.1 标准过程 new ()	(133)
7.3.2 动态变量的用法	(134)
7.3.3 p 与 p↑的关系	(134)
7.3.4 指针类型的运算	(135)
7.3.5 标准过程 dispose ()	(136)
7.4 简单链表	(136)
7.4.1 结点的定义方式	(137)
7.4.2 指针、结点和结点之间的关系	(139)
7.5 链表示例	(141)
习题七	(146)
第八章* 文件类型	(148)
8.1 文件的概念	(148)
8.2 文件类型和文件变量	(149)
8.3 文件的产生和使用	(150)
8.3.1 文件的打开和关闭	(150)
8.3.2 正文文件的读写	(152)
8.3.3 input 和 output	(157)
8.4 文件示例	(158)
习题八	(161)
附录	(162)
附录 A ASCII 码表	(162)
附录 B 保留字和预定义标识符	(163)
参考文献	(164)

第一章 PASCAL 程序基本概念

计算机是 20 世纪科学技术发展史上最卓越的成就之一。计算机的出现推动着当今科技的飞速发展,目前计算机应用领域已经涉及到人们生活的方方面面。为了让计算机为我们做更多的事,就必须事先编写好计算机程序,让计算机按照程序中的指令,一步步地为我们工作。那么什么是程序?怎样编写程序?能不能直接与计算机对话?这些正是我们学习的主要内容。

1.1 PASCAL 程序简介

1.1.1 程序与 PASCAL 语言的基本概念

自从计算机诞生以来,人们设计出很多种不同的程序设计语言,PASCAL 语言就是其中之一,它是由瑞士苏黎世联邦工业大学的沃思(N. Wirth)教授于 1971 年推出的,它的命名是为了纪念法国数学家 Pascal。PASCAL 语言问世以来,深受大家的喜爱,一度成为世界上使用最广的程序设计语言。直到现在,仍有强大的生命力。

下面从一个例子来引出程序与 PASCAL 语言的基本概念。

小明新买了一套组装玩具,他想组装一辆小汽车。于是,他打开说明书,按说明书的步骤操作起来:

第一步,找出所有与汽车有关的部件;

第二步,按图示用部件装成车身;

第三步,按图示用部件装成轱辘;

第四步,按图示将车身、轱辘装成汽车。

小明按照说明书中给出的组装“程序”一步步装成了小汽车。编写计算机程序和编写说明书相类似,程序中有控制计算机操作的步骤,人们把这些步骤称为指令,一个完整的计算机程序就是由众多指令组成的指令序列。

小明认识汉字,所以他能边看说明书边进行组装,如果他不识字或说明书是外文的,他要想自己动手组装,就得请别人给他讲解说明书中的安装步骤。人们编写计算机程序时,通常使用程序设计语言,编出的程序叫源程序,源程序中的指令又叫语句,由于计算机不能直接阅读源程序中的语句,因此要经过一个“翻译程序”把源程序翻译成由机器语言(这是一种计算机能懂的语言)组成的目标程序,再由计算机执行目标程序。

PASCAL 语言有很多优点,其中最主要的是:

(1) 数据类型丰富

除了整型、实型、字符型、布尔型等常用简单类型之外,PASCAL 还提供了枚举、子界、数组、记录、集合、文件、指针等众多的用户自定义类型,给人们编程提供了极大的方便。

(2) 程序设计结构化

PASCAL 的程序结构特别简单、紧凑、框架优美、表达能力强,体现出自顶向下的结构化程序设计风格,用 PASCAL 编写的源程序可读性极好。

(3) 优秀的教学语言

由于用 PASCAL 描述的算法直观、准确、易于交流,很多计算机课程中的算法都是用 PASCAL 描述的,所以很多学校都首选 PASCAL 作为程序设计的教学语言。

本书以标准 PASCAL 语言为蓝本,个别地方介绍 TURBO PASCAL(后文简称 TP)的特殊规定。

1.1.2 PASCAL 程序结构及流程图

我们知道,程序是由一条条语句组成的。一般情况下,执行程序时按照语句在程序中的先后次序逐条执行,但有时会根据需要改变语句的执行次序。程序中语句的实际执行次序叫做程序的流程。不同的程序结构有着不同的程序流程。PASCAL 语言有四种程序结构:顺序结构、分支结构、循环结构和子程序结构。

(1) 顺序结构

顺序结构是一种最简单的程序结构。处在顺序结构中的语句按其出现的先后次序逐条执行。如图 1.1 所示,首先执行语句 S1,执行完语句 S1 之后,再执行语句 S2,然后执行语句 S3。

(2) 分支结构

分支结构又称选择结构。在这种结构的开头,设置一个判断条件,根据判断条件成立与否,确定执行程序的哪个分支。如图 1.2 所示,若判断条件结果为真则执行语句 S1,否则执行语句 S2。图 1.3 是另一种分支结构,若判断条件为真则执行语句 S1,否则语句 S1 不被执行。

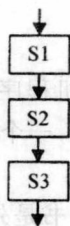


图 1.1 顺序结构

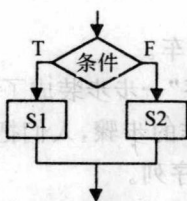


图 1.2 分支结构一



图 1.3 分支结构二

(3) 循环结构

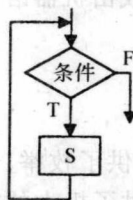


图 1.4 循环结构

循环结构又称重复结构。循环结构有好几种。图 1.4 中给出的是当型循环结构,当判断条件结果值为真时,则执行语句 S,并重复循环(即再次判断条件真否,若真,就再一次执行语句 S),直到条件不成立(即为假)才停止循环。

(4) 子程序结构

子程序结构的执行流程可由下例说明。

某公司承包修建一条高速公路,公司把修路工程分解成三个子任

务:A. 挖土方;B. 铺路基;C. 铺柏油路面。这三项子任务分别由其三个子公司完成。我们可以把 A,B,C 分别看成三个相互独立的具有一定功能的程序段,每个程序段称作一个子程序。若某处需要挖土方,公司(主程序)就调度 A;某处要铺路基,就调度 B;要铺路面就调度 C,如图 1.5 所示。

子程序的引入,减少了重复编程,容易排查错误,也使程序层次分明、结构清晰。图 1.6 表示调用子程序时的程序流程,主程序 main 执行到调用子程序 A 的语句时,则暂停 main 的执行,而转去执行子程序 A,执行完子程序 A 之后,返回主程序 main,继续执行调用语句下面的一条语句。



图 1.5 子程序结构

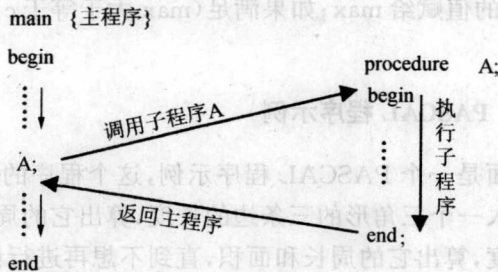


图 1.6 子程序调用示意图

一个大型的完整的 PASCAL 程序,其结构可以是这四种结构的叠加、嵌套,但就其整体结构来说,仍属顺序结构,所以说顺序结构是基本结构。

为了直观形象地表达程序设计者的解题思路,通常在编写程序时,先画出程序流程图。流程图含有以下六种图形符号(见图 1.7):

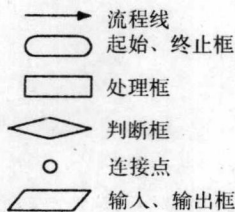


图 1.7 流程图图符号

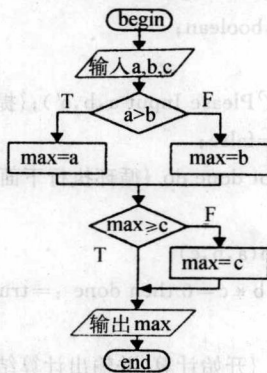


图 1.8 求三个数中最大值的流程图

- ① 流程线。一条带有箭头的线条,表示程序流程的方向和路径。
- ② 起始框和终止框。用椭圆代表程序的开始与结束。
- ③ 处理框。用矩形和矩形中的文字表示某种处理或运算步骤。
- ④ 判断框。一个菱形框和框里的测试条件,表示一个选择结构,用 T 或 Y 表示当条件为真时流程转向一个分支,用 F 或 N 表示条件为假时流程转向另一个分支。
- ⑤ 连接点。用一个圆圈和圆圈中的符号(通常选用数字或字母)将一个庞大流程图

的两部分连接起来,具有相同数字或符号的圆圈表示同一个流程点。

⑥ 输入、输出框。在一个平行四边形内写明输入或输出内容。

图 1.8 给出求三个整数 a, b, c 中最大值的流程图。第一个椭圆表示程序开始,接下来的平行四边形中注明要求输入 a, b, c 三个整数,再下面的菱形框写出判断条件“ $a > b$ ”,用“T”表示若 a 确实大于 b ,则流程转向左边的处理框,使 $\max$ 获得 a 的值;若上述条件不成立,即 a 不大于 b ,则流程转向右边的处理框,使 $\max$ 获得 b 的值。在完成这两个处理框(实际上只做其中之一)后两侧的流程线汇合, $\max$ 的值便等于 a, b 两数中的较大者。紧接着的菱形框中的判断条件“ $\max \geq c$ ”,如果不满足(c 大于 $\max$),则流程转向右侧的处理框,将 c 的值赋给 $\max$;如果满足($\max$ 大于等于 c),则流程向下。最后的输出框将 $\max$ 的值输出。

1.1.3 PASCAL 程序示例

下面是一个 PASCAL 程序示例,这个程序的功能是计算多个三角形的周长和面积。每次输入一个三角形的三条边的长度,算出它的周长和面积,再输入下一个三角形的三条边的长度,算出它的周长和面积,直到不想再进行计算为止。

为了简化程序结构,我们这里假设每次输入的三条边的长度都是正的实数,且满足三角不等式——任意两边之和均大于第三边。当某次输入的三条边的长度中有一个(或几个)为零,就表示不再进行计算,程序终止。

例 1.1 求三角形周长和面积的程序。

{这是一个输入三条边,求三角形周长和面积的 PASCAL 程序}

```
program ex11(input,output);           程序首部
var a,b,c,p,s,length,area:real;      说明部分
    done:boolean;
```

```
begin
```

```
    writeln('Please Input a,b,c'); {提示输入三条边长度}
```

```
    done := false;
```

```
    while not done do {循环执行下面的处理}
```

```
    begin
```

```
        readln(a,b,c);
```

```
        if a * b * c = 0 then done := true {设置停止工作标记}
```

```
        else
```

```
        begin {开始计算,并输出计算结果}
```

```
            length := a + b + c;
```

```
            p := length / 2;
```

```
            s := p * (p - a) * (p - b) * (p - c);
```

```
            area := sqrt(s);
```

```
            writeln('length=',length:10:2,'area=',area:10:3)
```

```
        end
```

```
    end;
```

```
    writeln('The end!') {程序工作结束}
```

```
end.
```

— 程序体

一个完整的 PASCAL 程序,大都包括程序首部、说明部分和程序体三大部分。

程序首部——以保留字 program 开始,后接程序名(上例中程序名为 ex11,程序名可以任意选用)和外部参数表(input,output),在 TP 中可以不带外部参数表,甚至整个程序首部都可以省略。

说明部分——用户在程序中使用的符号常量、变量、类型、标号、过程和函数等,都必须在程序的说明部分加以定义说明,然后才能在程序的下文中使用它们。习惯上,这些说明(如果使用的话)在程序中的先后次序是:

- 标号说明部分;
- 常量定义部分;
- 类型定义部分;
- 变量说明部分;
- 过程或函数说明部分。

上例中,只有变量说明部分(由 var 打头),定义了 a,b,c,p,s,length,area 等七个实型变量和一个布尔型变量 done,没有其它说明部分,所以它还只是一个简单程序。

程序体——以语句括号“begin,end”括起来的可执行语句序列构成程序体。主程序的程序体后面加一个小圆点表示整个源程序结束。

PASCAL 源程序的书写格式是自由式,即一行可写多条语句,一条语句也可占多行,语句和语句之间一定要用分号隔开,词和词之间要留有空格。为了便于阅读,人们喜欢采用层次感较强的嵌套缩进格式,就像上例那样。

在 PASCAL 源程序中的任何地方,都可以在一对花括号“{ }”内书写一段文字表示注解(又叫注释)。正如图 1.1 的示例程序那样。

注释不是源程序语句(它不会被计算机执行),加注释的目的完全是为了阅读程序方便。注释通常加在某段程序开头,说明本段程序的功能,或者加在某条重要语句之后,介绍该条语句的作用。任何内容都可作为注释,包括英文、汉字等。

1.2 PASCAL 的基本语法成分

1.2.1 字汇表

在上一节的示例程序 ex11 中,出现了 program,var,begin,end,read,a,b,c 等字母或词汇,它们都是 PASCAL 程序的基本词汇,PASCAL 语言中的所有基本词汇构成 PASCAL 的字汇表。字汇表有以下三类基本词汇:

(1) 基本符号

PASCAL 程序中可使用的基本符号有:

① 大小写英文字母:A~Z,a~z(注:PASCAL 中字母不分大小写,除非作为字符或字符串中的内容出现)

② 数字:0~9

③ 特殊符号,共有下列三类:

单一符号: + - * / = , ; ^ < > .

三对括号: () [] { }

组合符号: <= >= <> :=

(2) 保留字

保留字是 PASCAL 语言规定的具有专门语法含义的单词,不能随意使用。例如 program 表示程序首部的开始,begin,end 表示语句括号等。由于 PASCAL 语言规定字母不区分大小写,所以 Program,program,PROGRAM 等都表示同一个保留字。

不过为了醒目,以示与一般的标识符相区别,有的教材中规定保留字一律用大写字母书写。但为了程序录入方便,也可都用小写字母(本书都用小写字母)。各保留字的具体语法含义分别在有关章节中详细介绍,为读者查阅方便,本书将 PASCAL 语言规定的全部保留字列在附表 B 中。

(3) 标识符

标识符用来标记程序处理的常量、变量、类型、过程和函数、程序、文件等对象的名称,一个标识符可以是一个常量名、变量名、类型名或过程名等等,所以又统称为名字。

有两类标识符,一类是 PASCAL 系统预先定义的标准标识符(又称预定义标识符);一类是用户自定义标识符。例 1.1 中的 real,boolean,readln 等都是标准标识符,a,b,c 等都是用户自定义标识符。

每个预定义标识符都有特定的含义,标准 PASCAL 规定的全部预定义标识符都列在附表 B 中,至于它们的具体含义,我们将在后文中陆续介绍。注意,不同版本的 PASCAL,所含预定义标识符的个数不尽相同。

用户自定义标识符是用户为自己程序中引入的变量、类型等所起的名字,每个自定义标识符都必须在程序的说明部分加以定义。定义的同时,也就为标识符起了名字。

PASCAL 规定标识符必须以字母打头,后面可以再跟随几个字母或数字,不能含其它符号。有的版本规定标识符的长度(即字符个数)不能超过 8,有的版本允许长一些,比如 64 等,不过名字取得太长也麻烦。

虽然用户可以为自定义的标识符任意取名,但最好不与标准标识符同名,否则标准标识符就失去原来的含义。

标识符一定不能与保留字同名,否则系统会认为程序出现了错误。

从结构化程序设计的观点来看,标识符的命名应尽量体现它的具体含义,能够起到“见名知义”的作用,给阅读程序带来方便。例如,用 length 表示长度,用 area 表示面积等等。

无论是标准标识符,还是用户自定义标识符,其中的字母不分大小写,即认为 Real 与 REAL、real 是同一个标识符。

例如,下面是合法的标识符:

pascal y12 mf105 student node ptxt

下面是非法的标识符:

8pt 数字开头

a&5 “&”不是字母

p.txt “.”不是字母

a 8 a 和 8 之间不能有空格

注意,TP 中,标识符里可含下划线。如:ex_1,emn_6,_time 等都可用作标识符。

1.2.2 常量和变量

PASCAL 允许使用常量、变量表示数据。程序中的常量、变量与数学上的常数、变量或自变量有很多相似之处,但也不尽相同。程序中的常量、变量有类型的概念,而且占用一定的存储单元。使用时要加以注意。

1. 常量

常量有两种使用方式,一种是直接在程序中书写具体数值,比如,0,1,100,3.14 等等;另一种是在程序首部,用保留字 const 定义的符号常量。

例1.2 根据半径求圆周长和面积的 PASCAL 程序。

```
program ex12(input,output);
const pi=3.1416;
var r,c,s:real;
begin
    writeln('Input radius:');
    read(r);
    c:=2*pi*r;
    s:=pi*r*r;
    writeln('r=',r,' c=',c,' s=',s)
end.
```

这个程序中就使用了两个常量,一个是整数 2,另一个是圆周率 pi。其中 2 是直接写在程序中的常量,pi 是在程序首部用 const 定义的符号常量,它代表 3.1416。

常量的数据类型由其书写形式而确定。例 1.2 中,2 是整型,pi 是实型。PASCAL 语言引入符号常量,一是为了编写通用程序,二是为了增加程序的易读性。

程序中使用的符号常量必须在程序的说明部分加以定义。常量定义的一般格式为:

```
const
    <标识符 1>=<常量值 1>;
    <标识符 2>=<常量值 2>;
    .....
    <标识符 n>=<常量值 n>;
```

例如:

```
const ok='Y'; eps=1E-6;
      maxn=1000; city='Nanjing';
```

一共定义了四个常量,ok 是字符型,city 是字符串,eps 是实型,maxn 是整型。

常量的值在程序的执行期间是不能改变的,即使是符号常量,也不能用赋值语句给它赋值,不能用 read 语句给常量输入数值。

注意,在给常量命名时,代表常量的标识符不能重复使用,也不能一个标识符代表多个常量值。

例如,下面常量定义是错误的:

```
const
```

```
  p=587;
```

```
  m=30,20;
```

```
  p=3.14;
```

m 不能既是 30, 又是 20; p 前后定义了两次。

应特别注意, 在一个程序段中, 同一个标识符不能定义两次, 不管是常量定义, 还是变量定义, 或其它定义, 否则就是“重复定义”, 系统将报错。

各种类型的常量数值书写格式及用法, 将按类型逐一介绍。

2. 变量

顾名思义, 变量是在程序运行期间其值可以改变的量。几乎所有的程序都要使用变量, 正是引入了程序变量, 才使程序变得方便灵活。

下面的示例程序能够说明这一点:

```
program ex13(input,output);
```

```
begin
```

```
  writeln(3.14 * 5 * 5)    {计算半径为 5 的圆面积}
```

```
end.
```

这段程序只能用来计算半径为 5 的圆面积, 如果要计算半径为 10 的圆面积, 就必须修改源程序, 把程序中的 5 都改为 10, 如果要计算半径为 8 的圆面积, 又要把 10 改为 8。这显然太麻烦、太笨拙了。

我们不妨把程序改为:

```
program ex14(input,output);
```

```
const pi=3.1416;
```

```
var r,s:real;
```

```
begin
```

```
  read(r);
```

```
  s:=pi * r * r;
```

```
  writeln(r:10;2,s:20;4)    {计算半径为 r 的圆面积}
```

```
end.
```

用变量 r 表示圆半径, 运行程序时, 从键盘输入不同的 r 值, 就能计算出不同直径的圆面积。这样, 程序才具有通用性。

程序使用的变量必须在程序说明部分加以定义, 指定它所具有的类型。变量的定义格式如下:

```
var
```

```
  <变量名 1>:<数据类型名 1>;
```

```
  <变量名 2>:<数据类型名 2>;
```

```
  .....
```

```
  <变量名 n>:<数据类型名 n>;
```

例如:

```
var
```

```
  s:real;
```