李启炎 / 陈福生 / 宋秋杰 / 编著

# PASCAL 程序设计语言

（第二版）




同济大学出版社

# 序　言

PASCAL 语言是在本世纪 60 年代末 70 年代初由瑞士 N. Wirth 教授提出的。它是在 ALGOL 60 的基础上发展起来的一种结构程序设计语言(语句结构化和数据结构化)。具有功能强、数据类型丰富,编写的程序简洁易读等特点。适用于数值和非数值问题的描述,既能用于应用程序设计,也能用于系统程序设计。因此,它已成为当前最流行的几种程序设计语言之一。由于 PASCAL 语言的程序逻辑结构的严密性,在欧美国家中越来越多的大学选用 PASCAL 语言作为教学的第一计算机语言。

鉴于 PASCAL 语言的上述特点,我国很多高等院校的计算机专业或非计算机专业也都选用了 PASCAL 语言作为计算机的教学语言,使学生一开始就培养良好的程序设计风格和掌握较全面的程序设计技巧。本书是编者在我校多年的 PASCAL 语言程序设计课程教学实践的基础上编写的。编写方法采用了子集扩充法,即每学完一章都能编写出完整的程序实例进行上机实习。由简到繁,循序渐进,以利于学习。

任何一种高级语言必须有一个编译程序将它翻译成机器指令程序,才能在特定的计算机系统中进行运行。而编译程序本身也是在一个特定的环境下工作的(即特定的计算机硬件及操作系统的支持)。故在特定的工作环境下,每一种高级程序设计语言就有一种具体的实现方案。一个实用的 PASCAL 实现方案除了包括标准的 PASCAL 语言的语法外,总还包括某些扩充的语法成分,以增强语言的功能。考虑到 TURBO PASCAL 语言功能的先进性以及使用的广泛性,因此我们在取材时,除了标准 PASCAL 语言的成分外,还适当地将目前较流行的 TURBO PASCAL 的图形功能和面向对象程序设计方法,作为扩充部分编入本书。

全书共分为十一章和六个附录。第一章简单叙述了计算机系统的组成和部分功能,程序设计和编译、执行的基本概念,PASCAL 源程序的结构及 PASCAL 基本字符集和语法单位;第二章概述了 PASCAL 语言中的数据类型以及变量、表达式等基本概念,介绍了赋值语句和输入、输出语句;第三章和第四章介绍了 PASCAL 语言中的流程控制语句,包括 IF 语句、REPEAT 语句、WHILE 语句、FOR 循环语句、GOTO 语句和 CASE 语句;第五章对 PASCAL 子程序——函数和过程作了较详尽的叙述;第六章介绍了用户定义的数据类型:枚举类型、子界类型,并介绍了结构化数据类型——数组类型以及字符串变量;第七章详细介绍了另外两种结构化数据类型——集合和记录;第八章叙述了文件的概念,对如何建立文件、更新文件、合并文件等常规文件操作通过实例作了详细介绍,并对文本文件的概念和应用进行了讨论;第九章叙述了指针和动态数据结构的概念,重点介绍了链表和树等动态数据结构以及对它们所进行的遍历、插入、删除等常规操作;第十章介绍了 TURBO PASCAL 中单元的概念,并重点叙述了其中的 Graph 标准单元以及在图形功能程序设计中的使用;第十一章介绍了面向对象的基本概念以及 TURBO PASCAL 语言中面向对象的程序设计方法。在整个叙述中,从实例引出概念,由浅入深,对基本语法的叙述力求准确完整,但并未给出严密的语法定义范式。在附录 A 中给出了 PASCAL 语法结构的语法图;附录 B 中介绍了 TURBO PASCAL 对标准 PASCAL 的扩充等;附录 C 中介绍了 TURBO PASCAL 语言集成开发环境的使用;附录 D 中给出了

TURBO PASCAL 语言系统中出错信息表；附录 E 中列出了 TURBO PASCAL 语言中图形模式以及标准 Graph 单元中的图形功能的一些过程和函数；最后附录 F 给出了 ASCII 码表。

本书列举的一些颇有实用价值的例题全部调试运行过，并在一些例题中力图使用“自顶而下”的程序设计方法，以体现出结构化程序设计的良好风格。读者通过本书各章的学习，参阅附录就能编写程序并可在计算机上进行上机调试运行。

借本书出版之际，对同济大学和兄弟院校曾经关心、支持和帮助过编者的许多先生和女士们表示衷心的感谢，并致以 21 世纪的良好祝愿。

编者

1998 年 10 月 28 日

# 目　录

# 第一章　计算机系统及程序设计概念

## 1.1　计算机系统概述

计算机可高速地处理数据信息，它在现代经济建设和科学技术领域中的作用越来越为人们重视。要使用计算机首先有必要概括地了解计算机。一个计算机系统包括系统硬件和系统软件。

系统硬件包括：中央处理机，主存储器和外围设备(图 1-1)。

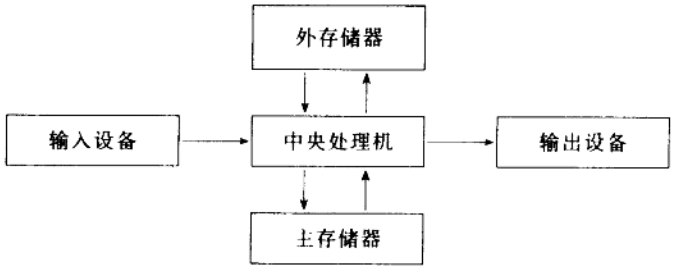


图 1-1　计算机硬件配置

中央处理机由运算器和控制器组成，它发出控制命令并按一定的顺序执行各种操作，如算术运算和逻辑运算。

主存储器是用来存放系统的常驻信息和当前被中央处理机所处理的信息和数据的，它速度快但容量较小。外存储器是用来存储当前暂不处理的信息和数据的，它速度较慢但容量大。存储器是按字节或字进行编址的。每个字节为 8 个二进位(即 8 bit)。一个字的长度对不同的计算机有不同的定义：微型或小型机一般是 8 bit 或 16 bit；中大型机一般是 32 bit，也有 48 或 64 bit。主存储器的容量从几十千字节到几兆字节。外存储器通常是磁盘、磁带，其容量可大到几百兆、几千兆字节。在外存储器中，信息一般以文件形式存放。

输入输出设备是用来向计算机输入数据信息和由计算机输出结果的。常用的输入设备是终端键盘。常用的输出设备是宽行打印机及终端显示设备。

系统软件主要包括操作系统及其所支持的一些实用程序和语言编译程序。其作用是统一管理计算机系统的软硬件资源，包括输入输出设备的管理，存储器管理，文件管理，作业调度及编译源程序等。它是用户和计算机硬件之间的接口，是整个计算机系统中的指挥部分。因此只有系统硬件而没有系统软件的计算机是不能有效地工作的。

## 1.2　源程序的编译和执行

对于任一特定的计算机来说，它只能识别特定的机器语言(即指令系统)。它的工作过程总

是用机器指令编成的程序来处理输入数据，然后得到输出数据。如图 1-2 所示。

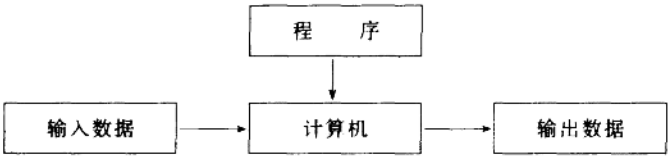


图 1-2　计算机工作过程

图 1-2 中的程序是按数据处理的要求用机器语言编写的指令串。解决各种不同的问题需要各种不同的程序。由于机器指令繁琐难懂，对计算机的推广使用是一大障碍。PASCAL 语言是一种高级语言，由于接近自然语言，语法规则简单清晰，故易于各类专业人员掌握使用。但计算机是不"懂得"这种高级语言的，因此用高级语言写的程序（源程序）必须有一个编译程序将它编译成机器指令组成的程序（目标程序），才能在一个特定的计算机系统中执行。翻译 PASCAL 源程序的程序称为 PASCAL 编译程序。

综上所述，在计算机上运行一个 PASCAL 源程序的过程分两步进行。

第一步：编译

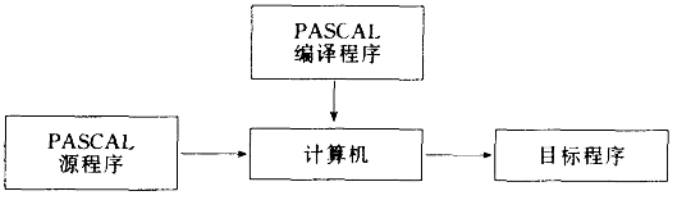


图 1-3　编译过程

第二步：运行

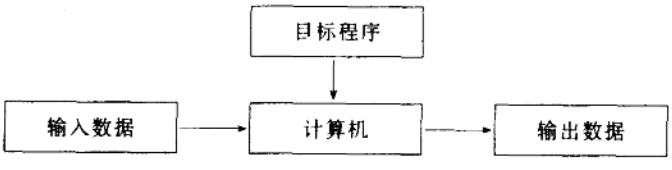


图 1-4　运行过程

由图 1-3 和图 1-4 可知，计算机被使用了两次。第一次使用时，所用的程序是 PASCAL 编译程序，输入数据是 PASCAL 源程序，得到的输出数据是目标程序。第二次使用计算机时，所用的程序是目标程序，对输入数据进行处理后得到所需的输出数据。

一般而言，在一个源程序中难免存在一些错误。这些错误可分为三类：

（一）在编译时发现的错误，称为编译错误。例如，对于一个 BEGIN 漏掉相应的 END 等。

（二）执行目标程序时发现的错误，称为运行错误。例如，在执行函数 sqrt(x)时，x 的值为负值。

（三）在编译和运行时，计算机不能发现的错误。例如，由于程序员的笔误，将 sqrt(x)写成 sqr(x)，则计算机将把 $\sqrt{x}$ 错误地计算成 $x^2$。

当发生编译错误时，计算机将报告出错信息。严重的编译错误会导致不能产生目标程序。当发生运行错误时，计算机也会报告出错信息。

由于程序设计中难免发生错误，需要对程序进行调试，找出语法上和逻辑上的错误，以便进行修改。这样的过程往往要进行多次，才能得到一个较为完善的可执行程序。进行程序设计应力求使程序具有下列特点：

(1) 正确性。这要求程序员要熟悉所使用的程序设计语言，避免语法、语义上的错误；还要设计简单易行的算法来达到预期的目的。有些问题本身可能是很复杂的，这就要求程序员采用有效的程序设计方法，譬如流行的自顶向下(TOP-DOWN)逐步求精的方法，使复杂问题简明化。

(2) 易读性。要调试一个程序就要求程序的结构清晰易读，这样就易于查找错误。将程序提供给他人阅读时，就更要求程序的易读性。

(3) 运行效率高。这里指的是运行时间短，占用存储空间小。这是人们力求达到的目标，需在不断地学习和实践中提高程序设计水平。

## 1.3 PASCAL 程序结构

在系统地学习 PASCAL 语言之前，我们先通过一个简单的 PASCAL 程序来初步认识完整的程序结构。虽然程序的书写格式对编译程序来说是不起作用的，但为了易于阅读，本书所有的程序实例中都用紧缩对齐格式书写。并用大写字体表示保留字，如 BEGIN，END，REPEAT，UNTIL 等。用小写字体表示标识符。

程序 minimax 读入一组整数，对它们计数，并记录其最小值和最大值。在尚未学习语法知识之前，我们可用英文词义来理解程序在做什么。

```
PROGRAM minimax (input, output);                    程序首部
CONST
    min=-32767;  }
    max= 32767;  }  常量说明                        }
VAR                                                 } 说明部分
    reading : boolean;                              }
    number, minimum, maximum, count : integer;  }  变量说明
BEGIN
    reading := true;  count := 0;                   }
    minimum := max; maximum := min;                 }
    WHILE reading DO                                }
      BEGIN                                         }
        read(number);                               } 语句部分
        IF number=0 THEN reading := false           }
                    ELSE BEGIN                      }
                      count := count+1;             }
                      IF number<minimum THEN        }
```

```
                minimum : =number;
              IF number>maximum THEN
                maximum : =number;
            END;
    END; {WHILE}
    writeln(count, 'number read');
    writeln('The smallest was', minim m);
    writeln('The largest was', maximum)
END.
```

一个程序是由首部、说明部分和语句部分组成的。有时把说明部分和语句部分合称为程序块。一个程序总是以句点结束。首部是保留字 PROGRAM 后接程序名及程序参数表(即此程序所用到的文件列表)。说明部分包括各类说明,本程序中包括常量说明和变量说明,其主要作用是告诉编译程序一些必要的信息,以便安排存储空间及确定各变量的数据类型。语句部分是程序的执行部分,由一系列的语句组成,它们描述程序所执行的算法和动作。

由程序 minimax 可知,在首部、说明部分及语句部分之间均以分号隔开,说明部分中各说明之间及语句部分的各语句之间也均以分号隔开。程序中括在花括号中的字符串是程序的注解。注解无语法意义,编译程序将忽略注解。注解可出现在任何空格可出现的地方。在本书后面一些章节的某些较复杂的程序例中,我们还插入了一些括在花括号中的中文注释,这仅仅是为了便于阅读。该程序首先将记数器 count 置零;将计算机允许存放的最大整数(32767)送到存放最小值的结果变量 minimum 中去;将计算机允许存放的最小整数-32768 送到存放最大值的结果变量 maximum 中去。在完成这些准备工作后,逐个读入数据,每读入一个数,记数器 count 加 1。若读入的数据小于 minimum 中的值或大于 maximum 中的值,则用新的数据代替 minimum 或 maximum 中的值。重复执行上述过程,直到读入一个零值为止。最后,该程序输出三个结果:总共读入的数据个数,它们中的最小值和最大值。

这一简单程序已包含了 PASCAL 程序的基本结构,从第二章开始,我们将对程序的每一组成部分进行详细的描述。

## 1.4 基本语法单位

上一节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,除了规定一套严密的语法规则外,还必须规定一套基本的语法单位,以便按照语法规则将它们构造成语言的各种成分(如常数、变量、表达式、语句、说明等)。本节将叙述 PASCAL 的基本语法单位。

PASCAL 基本语法单位有两类:字和专用符号(如:+,-,*,/,:=,;等)。它们是由 PASCAL 字符集中的字符组成的。PASCAL 字符集包括英文字母、数字及一些符号,实际上是 ASCII 字符集的一个子集。

字可分为两类:保留字和标识符。

保留字的作用是用来命名 PASCAL 语句,某些预定义的数据类型,某些操作符、说明段及程序首部的。现将所有的保留字分列如下:

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

除这些标准 PASCAL 的保留字外，在某些实现方案中还增加了保留字 OTHERWISE 来扩充 CASE 语句的功能(详见第四章中关于 CASE 语句的叙述)。读者不必死记硬背这些保留字，通过本书的学习，读者自然会掌握每个保留字的具体用法。

标识符是以字母开头的字母与数字的序列。有两类标识符：预定义的标准标识符及用户定义的标识符。

预定义的标准标识符是用来命名标准函数、标准过程、标准类型、常数及标准文件的。现分列如下：

常数：

false, true, maxint

标准类型：

integer, boolean, real, char, 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：

dispose, get, new, pack, page, put, read, readln,
reset, rewrite, unpack, write, writeln

用户定义的标识符是程序员根据需要按标识符的定义方法所定义的标识符。是用来为常数、变量、用户定义的类型、过程、函数及程序取名的。用户在选用标识符时需注意以下几个问题：

(1) 不能与保留字同名。

(2) 尽量避免与标准标识符同名，以免发生混淆。

(3) 必须严格按照标识符的定义来构成标识符。

(4) 必须遵照先定义后使用的原则，即一个用户定义的标识符必须首先出现在程序的说明部分，然后才能出现在程序的语句部分。

(5) 标准 PASCAL 对一个标识符只识别前 8 个字符。若两个标识符的前 8 个字符完全相同，则被认为是同一个标识符，如：abcdefgh1 与 abcdefgh2 是同一标识符 abcdefgh。

下面是一些合法的标识符：

a　　a1　　nameofcar　　line3

下面是一些非法的标识符：

1a　　不是以字母开头

begin　　与保留字同名

up.to　　出现非字母及非数字的字符

last name　空格不能出现在一个标识符的中间。

除注意上述五点外，为使程序清晰易读，在选用标识符时应适当取具有相应含义的英文名作为标识符。这样做既便于阅读又便于记忆。故在某些 PASCAL 实现方案中，把标识符的有效识别长度从 8 个字符扩大到更长，还允许在一个标识符中嵌入符号'-'，'$'等，这样写出的标识符更清晰易读，如：type-of-name(详见附录 B)。

## 习　题　一

1. 计算机系统由哪几部分组成？计算机硬件系统包括哪几部分？各部分的作用是什么？

2. 什么叫机器语言、高级程序设计语言？各有什么特点？

3. PASCAL 语言的程序结构由哪几部分组成？试述各部分的构成形式和作用。

4. PASCAL 基本字符集由哪些成分构成？PASCAL 保留字是什么？举例说明保留字的作用。

5. PASCAL 语言中有哪些标准类型、标准文件、标准函数、标准过程以及标准常数。

6. PASCAL 语言中标识符是怎么构成的？下列符号中哪些是正确的 PASCAL 标识符？哪些是错误的？为什么？

A5B，5H4，PEL.1，A31，BEGIN，ABS，x*y

A+B，$500，G1(x)，CI，VAR，xzy，E-10

# 第二章　数据、表达式、赋值语句及输入输出

在 PASCAL 中，一个程序所用到的每个常数、变量和函数标识符都或隐或显地与一种数据类型（简称类型）相联系。每种数据类型是由一组值和能在这组值上进行运算的一组运算符组成的。例如在字长为 2 个字节的计算机中，预定义的标准类型 integer（整型）是由 −32767 到 32767 的所有整数及能在这些值上进行运算的运算符 +，−，*，DIV，MOD 组成的。因此，类型既规定了与该类型相联系的常数、变量和函数标识符的取值范围，也规定了对它们能执行的运算。

常数的类型是由该常数本身隐含的，如：5 的类型是标准类型 integer。3.14159 的类型是标准类型 real。变量标识符和函数命名符的类型是在程序的说明部分显式地说明的。

PASCAL 中类型可分为三大类：标量类型、结构类型及指针类型。具体划分如下：

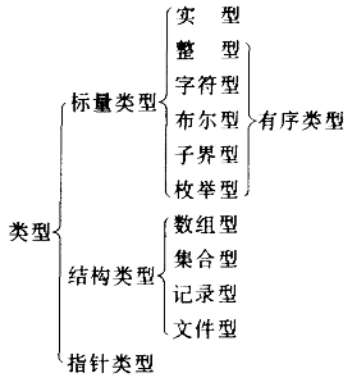


图 2-1　数据类型的分类

如图 2-1 所示，PASCAL 语言的数据类型极为丰富，共有十一种具体的类型。本章将介绍整型、实型、字符型和布尔型这四种预定义的标准类型。其他类型将在以后各章叙述。

## 2.1　常数及常数说明

在程序中可出现一些在程序执行前就已知的常数值，如：78，−5.63，'a'，true 等。下面叙述四种标准类型的常数。

### （一）整型常数

整型常数是带或不带符号（+或−）的数字串。如：

6　−13　+7800　0

都是整数，而

7.3　−0.0　15,000

都不是整数。前两个是实数，最后一个数中的逗号是非法的。

### （二）实型常数

实数有两种表示法。一种是习惯写法，如：

0.0　+0.965　−74.6　890

都是实数。实数的另一种写法是科学表示法（即指数形式），规则是在一个习惯写法的实数后面接字母 E 再后接一个整数。如：

| | |
|---|---|
| 6.25E + 08 | $(= 6.25 \times 10^{8})$ |
| − 51E − 1 | $(= -51 \times 10^{-1})$ |
| 0E0 | $(= 0 \times 10^{0})$ |
| 0.6378E2 | $(= 0.6378 \times 10^{2})$ |

都是合法的实数。

必须注意的是：在小数点前后必须都有数字；在一个实数中除了正负号、数字、小数点及字母 E 外不能有其他字符；在科学表示法中，字母 E 前必须有一个习惯写法的实数，E 后必须有一个整数；一个整数也可看成是一个实数（如：890）。下面是一些非法的实数例：

| | |
|---|---|
| 0. | 小数点后面没有数字 |
| .736 | 小数点前面没有数字 |
| 2,765.63 | 出现一个逗号 |
| E6 | 字母 E 前缺一个实数 |
| 5E1.3 | 字母 E 后接一个非整数 |
| .6E3 | 字母 E 前的实数出错 |

### （三）字符型常数

一个字符型常数是括在两个单引号之间的 PASCAL 字符集中的一个字符。如'a'表示字母 a，' '表示空格字符，'7'表示数字 7，等等。不同的实现方案对 PASCAL 字符集有不同的定义，但一般都包括：26 个大小写的英文字母，10 个数字及一些标点符号、运算符号及专用符号。

除了字符型常数外，在程序中还可使用字符串常数。字符串常数是括在引号中的一串 PASCAL 字符。如：

'His name is Li Min'

字符串常数可由字符型数组来表示，用法见第六章。

### （四）布尔型常数

布尔常数只有两个：false 和 true。

由于 PASCAL 中除上述四种标准类型的常数外，还有其他类型的常数，如：子界型、枚举型等常数。这些类型的常数的定义将在以后各章中讲到相应的类型时介绍。

在 PASCAL 中，还可由用户对常数取名，即用一个标识符来代表一个常数。这种常数是在程序的常数说明中定义的。如：

CONST

```
pi = 3.141593;
biggest = 1000;
smallest = -biggest;
blank = ' ';
date ='monday, january 16';
```

此常数说明规定 pi 为实型常数 3.141593；biggest 为整型常数 1000；smallest 为整型常数 -biggest(即-1000)；blank 为字符型常数' '；date 为字符串常数'monday, january 16'。有了这些常数说明后，在程序的后面部分需要出现这 5 个常数的地方可以用这些相应的常数标识符来代替。

由上例知，常数说明的语法规则是：

CONST

〈常数标识符〉=〈常数〉；

⋮

〈常数标识符〉=〈常数〉；

其中保留字CONST 表示开始一个常数说明段。每个常数标识符是程序员选择的标识符。等号后面是该标识符代表的常数值。常数说明之间用分号隔开。一个常数标识符不能与程序中其他任何标识符同名。等号后的常数可以是带有符号的前面已说明过的常数标识符(如上例中的-biggest)。在程序执行中不能改变常数标识符的值。

由于字长的限制，对于一个特定的计算机，只能表示整数的有限子集，故 PASCAL 有一个不必说明的标准整型常数 maxint，此常数标识符表示计算机所能接受的最大整数。以 16 位字长的机器为例，所能接受的最大整数为

$$\text{maxint} = 2^{15} - 1 = 32767$$

即该计算机的整数取值范围为：

$$-\text{maxint} \leqslant n \leqslant \text{maxint}$$

有了常数说明使程序易于修改。我们可在程序中将多次出现的同一个常数(如 1000)在常数说明中定义为一个常数标识符(biggest)，然后在用到此常数的地方用该常数标识符来代替。此后，若程序员要修改此常数时(如从 1000 改为 500)，只需修改常数说明中的那个常数就可以了(即改为 biggest = 500)。

## 2.2 变量及其类型

程序中的每一项数据不是常数就是变量，上面讨论的是常数。变量的值在程序执行中是可改变的。变量是以标识符来取名的。每个变量都具有一个相应的类型，这样就规定了该变量的取值范围及它所能执行的运算操作。程序中所用到的变量必须首先在变量说明段中加以说明，然后才能使用。变量说明段的语法规则是：

VAR

〈变量标识符表〉:〈类型〉；

⋮

〈变量标识符表〉:〈类型〉；

其中保留字 VAR 表示开始一个变量说明段。每个变量标识符表是一个或多个由逗号隔开的变量标识符组成的。冒号后面的类型是类型标识符或类型说明。每个变量说明之间用分号隔开。如：

```
VAR
    age, index, count : integer;
    y, value : real;
    condition : boolean;
    ch1, ch2 : char;
```

此说明段规定了 age，index，count 为整型变量；y，value 为实型变量；condition 为布尔型变量；ch1，ch2 为字符型变量。由图 2-1 知，PASCAL 中除了这四种类型的变量外，还可以有其他类型的变量，如子界型、枚举型、记录型、数组型、集合型、文件型、指针型等。关于这些类型的变量将在以后章节中讲到相应的类型时介绍。

### （一）整数类型

integer 类型的变量的取值范围与整型常数相同，利用前述的变量说明，我们有

$$-\text{maxint} \leqslant \text{age} \leqslant \text{maxint}$$

整型量的运算符有：

$$+(\text{加}),\ -(\text{减}),\ *(\text{乘}),\ \text{DIV}(\text{除}),\ \text{MOD}(\text{取余})$$

后三种运算符优先于前两种。其中+，－，＊的运算法则与一般的算术运算加、减、乘相同，即两个整型量运算得到一个新的整型量。DIV 的运算法则是两个整型量相除，略去余数得到整型量的结果，如：

$$6\ \text{DIV}\ 4 = 1$$
$$3\ \text{DIV}\ 5 = 0$$

MOD 的运算法则是两个整型量相除取余数，如

$$6\ \text{MOD}\ 4 = 2$$
$$8\ \text{MOD}\ 5 = 3$$
$$(-11)\ \text{MOD}\ 2 = -1$$
$$13\ \text{MOD}\ (-4) = 1$$
$$(-9)\ \text{MOD}\ (-2) = -1$$

将整型量作为自变量的标准函数列表如下，其中 x 为整型量。

| 函数名 | 函数值 | 例 |
|---|---|---|
| pred(x) | x − 1 | pred(5) = 4 |
| succ(x) | x + 1 | succ(− 3) = − 2 |
| abs(x) | \|x\| | abs(− 10) = 10 |
| sqr(x) | $x^2$ | sqr(− 6) = 36 |
| odd(x) | x 为 奇/偶 数时得 true/false | odd(11) = true<br>odd(16) = false |
| chr(x) | 序号为 x 的 PASCAL 字符 | chr(65) = 'A' |

上表中字母 A 所对应的序号为 65(见字符类型),true 和 false 为布尔常数。

### (二) 实数类型

与整数类型一样,计算机由于受字长的限制,在表示实数时有范围和精度的限制。如对于一个字长为 16 位的计算机,能接受的实型量的绝对值在 1E－38 和 1E＋38 之间。当绝对值超过 $10^{38}$时,发生溢出错误;当绝对值小于 $10^{-38}$时,则机器认为是零或发生下溢错误。此计算机中的实型量约有 7 位十进制有效数字。

实型量的运算符有:

＋(加), －(减), ＊(乘), /(除)

运算法则与一般的算术运算相同。乘除优先于加减。若一个操作数是实数,另一个操作数是整数时,则在操作前自动地将整数转换成实数,如计算

(6 ＋ 4) ＊ (1 ＋ 0.1)

时,先计算左边括号中的表达式

(6 ＋ 4) ＝ 10(整数)

再计算右边括号中的表达式

(1.0 ＋ 0.1) ＝ 1.1(实数)

最后计算乘法

10.0 ＊ 1.1 ＝ 11.0(实数)

除法的两个操作数可以是整数或实数,若为整数时与整除 DIV 不同,其运算结果仍为实数,如:

13 DIV 5 ＝ 2

而 13/5 ＝ 2.6

将实型量作为自变量的标准函数列表如下,其中 x 为实型量。

| 函数名 | 函数值 | 例 |
|---|---|---|
| abs(x) | $\|x\|$ | abs(－ 13.2) ＝ 13.2 |
| sqr(x) | $x^2$ | sqr(1.2) ＝ 1.44 |
| sin(x) | sinx | sin(1.570796) ＝ 0.999999 |
| cos(x) | cosx | cos(3.1415926) ＝ － 0.99999 |
| arctan(x) | $tg^{-1}x$ | arctan(1.0) ＝ 0.785398 |
| ln(x) | lnx | ln(2.1) ＝ 0.7419373 |
| exp(x) | $e^x$ | $e^{0.7419373}$ ＝ 2.1 |
| sqrt(x) | $\sqrt{x}$ | sqrt(1.44) ＝ 1.2 |
| trunc(x) | x 的整数部分 | trunc(7.12) ＝ 7, trunc(－ 7.6) ＝ － 7 |
| round(x) | 最接近 x 的整型量 | round(7.48) ＝ 7,round(－ 7.61) ＝ － 8 |

前八个函数的自变量也可是整型量。

### (三) 布尔类型

boolean 型量只能取两个值: true, false。

有三种布尔运算符,按它们操作的优先次序排列为