第一日　绪　　论

1.1　集合与关系

俗话说:“物以类聚”。世界上万事万物,大至宇宙、星系,小至原子、电子,在物质及其运动的各个层次上,各种事物无一不可分门别类。人们把同类的事物放在一起考虑,就组成了所谓的集合。在我们日常生活中,集合的例子比比皆是。例如,“太阳系中行星的全体”是一个集合;“某所学校所有在校学生”是一个集合。总之,当某些具有共同性质的事物汇合在一起,就形成了一个集合。

一、集合

1. 集合的概念及其表示法

我们把具有某种共同性质的事物的全体称为集合,简称集。组成集合的每个事物称为这个集合的元素。习惯上用大写字母 A、B、…等表示集合,用小写字母 a、b、…等表示集合的元素。当然,在一些特殊集合中,元素往往有既定符号。在本日中,自然数集用 N 表示;整数集用 Z 表示;有理数集用 Q 表示;实数集用 R 表示;复数集用 C 表示。

若有一个集合 A,当事物 a 是 A 中的一个元素时,就称 a 属于 A,记为 $a\in A$;当事物 a 不是 A 中的一个元素时,就称 a 不属于 A,记为 $a\notin A$。例如:$-2\in Z$,$-2\notin N$。

假定一个集合 A 中包含 n 个元素 $a_1,a_2,\cdots,a_n(n>=0)$,则称集合 A 是一个有限集,记为 $A=\{a_1,a_2,\cdots,a_n\}$。

例如:设 P 是小于 12 的质数集。因为 P 的元素为 2、3、5、7、11,

所以可记为

$$p=\{2,3,5,7,11\}$$

集合既然是由具有某种共同性质的事物组成的，那么用这种性质同样也能限定集合由哪些元素组成。这样就得到了集合的另一种表示方法，其一般形式如下：

$$A=\{x|P\}$$

其中 P 是指某种性质，x|P 就是指元素 x 具有性质 P，而 $A=\{x|P\}$ 则表示集合 A 是由所有具有性质 P 的那些 x 组成的。

例如：方程 $x^2-4x+3=0$ 的解集 S，可表示成

$S=\{1,3\}$ 或 $S=\{x|x^2-4x+3=0,x\in R\}$

当集合 A 中不包含任何元素时，则称 A 是一个空集，空集用符号 Φ 表示。如果集合 A 中包含的元素是无限的，则集合 A 称为无限集。在今后的讨论中我们将主要是在有限集上进行，因此如果未作说明，所说的集合都是指有限集。

2. 集合之间的关系和运算

(1) 集合的之间的关系

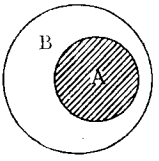


图 1-1 集合包含关系图示

设 A 和 B 是两个集合，如果 A 的每一个元素都属于集合 B，则称集合 B 包含集合 A，或称集合 A 包含于集合 B，记为 $A\subset B$。这时也把集合 A 称为集合 B 的子集，把集合 B 称为集合 A 的扩张集。用 Venn(文氏)图来表示 $A\subset B$ 是非常直观的，如图 1-1 所示。

如果集合 A 是集合 B 的子集，并且 B 中至少有一个元素 a 不属于 A，则称 A 是 B 的真子集。

例如：设集合 A 为 6 的正因数集，B 为 12 的正因数集，则

$A=\{1,2,3,6\}$，

$B=\{1,2,3,4,6,12\}$

因为集合 A 的元素都是集合 B 的元素，所以 $A\subset B$，即 A 是 B 的子集，且是真子集。

如果给定两个集合 A 和 B，它们满足 A⊂B 和 B⊂A，则称集合 A 和 B 相等，记为 A=B。也就是说两个集合中含有相同的元素（次序不一定一致）。

例如：C={2,3,1}　　D={1,2,3}

显然，由于集合 C 中的元素和集合 D 中的元素相同，即 C=D。

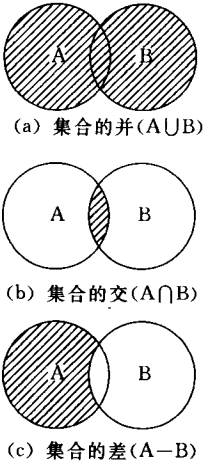
(2) 集合之间的运算

对于集合 A、B，定义它们的并集 A∪B、交集 A∩B 和差集 A−B 如下：

(1) 给定两个集合 A、B，由集合 A 和 B 中的所有元素构成的集合称为 A 和 B 的并集，记为 A∪B。图 1-2(a)为其文氏图表示。

(2) 给定两个集合 A、B，由同时属于集合 A、B 的所有元素构成的集合称为 A 和 B 的交集，记为 A∩B。图 1-2(b)为其文氏图表示。

(3) 给定两个集合 A、B，由所有属于集合 A 而不属于集合 B 的那些元素构成的集合称为 A 和 B 的差集，记为 A−B。图 1-2(c)为其文氏图表示。



(a) 集合的并(A∪B)

(b) 集合的交(A∩B)

(c) 集合的差(A−B)

图 1-2　集合的运算

例如：A={1,2,3,4}　　B={1,3,5,7}

则　A∪B={1,2,3,4,5,7}

A∩B={1,3}

A−B={2,4}

如果集合 A 和 B 满足 A∩B=Φ，就称集合 A 和 B 是不相交的。按照交集的定义，这也就是说，集合 A 和 B 没有公共的元素。

二、关系

在此我们首先介绍有序对（序偶）的概念。在初等数学中，最常见最典型的序偶，就是平面直角坐标系中点的坐标〈x,y〉。例如：点〈1,−2〉、点〈−3,5〉等。如果 a_1,a_2 是两个元素，按先后顺序将它们排列在一起，并且作为一个整体来看待，则称它为一个序偶，记为〈a_1,a_2〉。

注意$\langle a_1,a_2\rangle$和$\langle a_2,a_1\rangle$是不同的，因为它们的排列顺序不同。例如，在平面直角坐标系中$\langle 1,-2\rangle$和$\langle -2,1\rangle$是不同的，它们代表两个不同的点。

如果给定两个集合A和B，则定义它们的笛卡尔积如下，记为$A\times B$。

$$A\times B=\{\langle a,b\rangle \mid a\in A,b\in B\}$$

例如：设$A=\{a,b\}$，$B=\{1,2,3\}$，那么A和B的笛卡尔积为：

$$A\times B=\{\langle a,1\rangle,\langle a,2\rangle,\langle a,3\rangle,\langle b,1\rangle,\langle b,2\rangle,\langle b,3\rangle\}$$

$$B\times A=\{\langle 1,a\rangle,\langle 1,b\rangle,\langle 2,a\rangle,\langle 2,b\rangle,\langle 3,a\rangle,\langle 3,b\rangle\}$$

一般地，对于任何两个有限集A和B来讲，如果A有n个元素，B有m个元素，那么，$A\times B$和$B\times A$都有$n\times m$个元素。但是，正如上例中所表明的，$A\times B$和$B\times A$的元素个数虽然一样，其元素却不同，它们通常是两个不同的集合，即$A\times B\neq B\times A$。在$A\times B$中，如果B和A相等，即$B=A$，则笛卡尔积自然应为$A\times A$。

集合$A\times B$的每个子集R都称为从A到B的一个关系，或者称R是$A\times B$的一个关系。当$A=B$时，就称R是定义在A上的一个关系。如果R是定义在集合A上的一个关系，则$\langle a,b\rangle\in R$时，就称元素a和b有关系R，记作aRb。此时元素a称为元素b的前缀（或前驱），元素b称为元素a的后继。

设R是非空集合A上的一个关系，如果对于A中的任何元素a，都有$\langle a,a\rangle\in R$，则称关系R是自反的。如果对于A中任何元素a，都没有$\langle a,a\rangle\in R$，则称关系R是反自反的。如果对于A中的任何元素a和b，当$\langle a,b\rangle\in R$时，必有$\langle b,a\rangle\in R$，则称关系R是对称的。如果$\langle a,b\rangle\in R$且$\langle b,a\rangle\in R$时，必有$a=b$，则称R是反对称的。如果对于A中的任何元素a、b、c，当$\langle a,b\rangle\in R$，并且$\langle b,c\rangle\in R$，必有$\langle a,c\rangle\in R$时，则称关系R是传递的。

例如：整数集Z上的关系“$=$”是自反的，因为对于任意$x\in Z$，总有$x=x$；而整数集Z上的关系“$<$”是反自反的。因为对于任何$x\in Z$，$x<x$都不成立。整数集Z上的关系“$=$”是对称的，而关系“$>$”、“$\geqslant$”、“$<$”、“$\leqslant$”是反对称的。整数集Z上的关系“$<$”是传递的，因

为对于任何 a、b、c∈Z，如果 a<b 且 b<c，则必有 a<c。

当集合 A 上定义的关系 R 是自反的、对称的和传递的，这时就称 R 是一个等价关系。此时，如果〈a,b〉∈R，就称元素 a 和 b 是等价的。例如，整数集 Z 上的关系“=”是一个等价关系。

当集合 A 上定义了一个等价关系 R 时，可以根据这个关系将 A 中的元素分成若干部分，让它们分别属于 A 的若干个子集，这些子集互不相交，并且使同一个子集的任何两个元素都具有关系 R，而任何两个子集中的任何两个元素，都不满足关系 R，即这些子集构成了集合 A 的一些划分，而这些子集都称为关系 R 的等价类。

当集合 A 上定义的关系 R 是自反的、反对称的和传递的，则称 R 是集合 A 上的一个部分序（或偏序）关系。如果在集合 A 上定义了一个偏序关系 R，则称集合 A 为偏序集，记为(A,R)。显然，对于一个有限偏序集来说，至少有一个元素没有前缀，至少有一个元素没有后继。例如：自然数集 N、整数集 Z、有理数集 Q 和实数集 R 上的关系“≤”就是一个偏序关系。又如：在自然数集 N 上定义关系“/”如下：〈x,y〉∈/当且仅当 x 整除 y。不难证明，/是自然数集 N 上的一个偏序关系。

如果 R 是定义在集合 A 上的偏序关系，即(A,R)是一个偏序集，并且关系 R 满足下面条件：对于 A 中的任何元素 a、b，〈a,b〉∈R 和〈b,a〉∈R 两者至少有一个成立，这时就称 R 是集合 A 上的一个序（或完全序）关系，A 在这个序关系下称为有序（或完全序）集，仍记为(A,R)。例如：自然数集 N、整数集 Z、有理数集 Q 和实数集 R 在关系“≤”下都是完全序集。而自然数集 N 在如上定义的关系“/”下就不是完全序集。

1.2 数据结构概念

“数据结构”是一门随着计算机科学的发展而逐渐形成的学科，自 1946 年美国宾夕法尼亚大学的工程师和科学家发明了第一台电子计算机以来，计算机科学经历了将近半个世纪的蓬勃发展，其发展

速度远远超出了人们的预料，可谓日新月异。计算机硬件技术，软件技术不断提高，使计算机价格越来越便宜，功能越来越强大，也就使得其应用领域越来越广泛。计算机的应用早已不再局限于科学计算，而更多地用于过程控制、数据处理、信息管理，以及计算机辅助设计(CAD)和计算机辅助制造(CAM)等非数值数据的处理工作。与此相应，计算机加工处理的对象——数据也从单纯的数值数据发展到字符、表格、图像以及声音等各种非数值数据。为了更有效地使用计算机，设计出高效、可靠的程序，需要对数据的组织、数据元素之间的关系、数据在计算机中的表示(包括数据元素的表示和数据元素之间关系的表示)以及对数据的操作进行深入研究。这样，就促进"数据结构"这门学科的形成和发展。

一、什么是数据结构

在介绍数据结构这一概念之前，首先要了解一下数据的概念。所谓数据就是指对客观事物的符号表示。在计算机科学中我们把所有能输入到计算机中，并能被计算机处理的符号总称为数据。换言之，它就是计算机程序加工的原料。例如，一个代数方程求解程序，它处理的对象(数据)就是实数；又如：一个文字处理程序(如 WPS)，其处理的对象就是字符文字。对于计算机科学而言，数据的含义极其广泛，图像、声音等都可以通过编码而归之为数据的范畴。数据由数据元素组成，数据元素是数据的基本单位，通常在计算机程序中作为一个整体进行考虑和处理。而数据元素之间往往又不是孤立无关的，数据结构就是相互之间存在一种或多种特定关系的数据元素的集合，数据元素之间存在的相互关系就叫结构。根据数据元素之间关系的不同特性，通常有下列四类基本结构：

1. 集合：结构中的数据元素之间除了"同属于一个集合"的关系外，别无其它关系；

2. 线性结构：结构中的数据元素之间存在一个对一个的关系；

3. 树形结构：结构中的数据元素之间存在一个对多个的关系；

4. 图状结构(网状结构)：结构中的数据元素之间存在多个对多

个的关系。

图 1-3 为上述四类基本结构的关系图。其中，圆圈表示数据元素，圆圈之间的连线表示数据元素之间的关系。在本书中将主要讨论线性结构、树形结构和网状结构。“集合”，由于元素之间的关系极为松散，所以不作讨论。

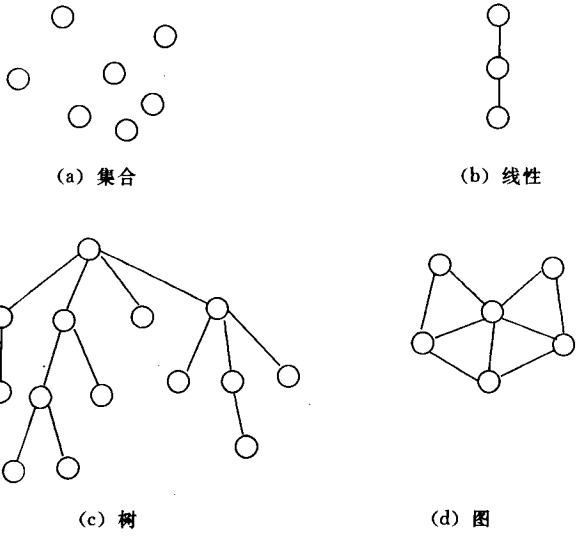


图 1-3 四类基本结构关系图

数据结构的形式定义可以用一个二元组表示：

$$\text{Data}-\text{Structure}=(D,R)$$

其中，D 为具有相同特性的数据元素的有限集，R 是 D 上数据元素之间关系的有限集。例如：复数在计算机科学中可作如下定义：

$$\text{Complex}=(C,R)$$

其中，$C=\{c_1,c_2 \mid c_1,c_2 \in \text{实数}\}$，$R=\{\langle c_1,c_2\rangle \mid c_1 \text{ 为实部}, c_2 \text{ 为虚部}\}$。又如：今欲编制一个企业管理程序，其中需要管理工厂中各车间的工人，则首先要为计算机处理的对象——车间工人设计一个数据结构，设一个车间由一个车间主任，三个班组的小组长以及 12 个工人组成，且这些成员之间的关系为：车间主任负责管理三个小组长，一

小组长，管理四个工人，则可以如下定义数据结构：

$$Grop=\{P,R\}$$

其中，$P=\{T,G_1,G_2,G_3,W_{11},W_{12},\cdots,W_{34}\}$

$$R=\{R_1,R_2\}$$

$$R_1=\{\langle T,Gi\rangle \mid 1\leqslant i\leqslant 3\}$$

$$R_2=\{\langle G_i,W_{ij}\rangle \mid 1\leqslant i\leqslant 3,1\leqslant j\leqslant 4\}$$

上述数据结构的定义仅是对操作对象的一种数学描述，而没有对操作对象定义具体的操作，换句话说，即从操作对象抽象出的数学模型。结构定义中的“关系”描述的是数据元素之间的逻辑关系，又称逻辑结构。

对数据结构的讨论，最终是为了在计算机中实现对数据元素的操作。为此我们不仅要讨论数据的逻辑结构及其运算，而且还要讨论数据结构在计算机中表示——数据的物理结构（又称存储结构），它包括数据元素的表示和元素之间关系的表示。数据元素之间关系的表示有两种不同的方法：顺序映象和非顺序映象。由此可得到两种不同的存储结构：顺序存储结构和链式存储结构（非顺序存储结构）。其中，顺序存储映象是借助元素在存储器中的相对位置来表示数据元素之间的逻辑关系；非顺序存储映象是借助在一个数据元素（a）中保存另一数据元素（b）在存储器中的相对位置来表示两个元素（a 和 b）之间的逻辑关系。

程序设计语言中，我们可以借助“数据类型”来描述数据的存储结构。例如：在 PASCAL 语言中，可以用“一维数组”描述顺序存储结构；用“指针”描述链式存储结构。

数据类型在程序设计语言中用以描述（程序）操作对象的特性。在高级语言中，每个变量都要属于一个确定的数据类型（整型、实型、字符型、布尔型等），不同的数据类型规定了在程序执行期间不同的变量的取值范围和允许的操作，所以，数据类型是一个值的集合和定义在这个值集上的一组操作的总称。例如，在 PASCAL 语言中的整数类型，它定义了其值集为区间[－maxin…maxin]上的整数（maxin 是依赖于具体计算机的最大整数），并规定了在这个值集上的一组操

作为：+、-、*、/、DIV、MOD 等。

与数据类型相关的还有一个概念称为数据对象。数据对象是某种数据元素的集合，是数据的一个子集。例如 PASCAL 语言中，布尔类型的数据对象是集合{TRUE(真)，FALSE(假)}，字符的数据对象为集合{x | x∈所有 ASCII 字符}。

二、数据结构的发展概况

六十年代初期，国内外的大学中都没有独立的“数据结构”课程，但数据结构的有关内容已散见于操作系统、编译原理和表处理语言等课程之中。1968 年，美国一些大学的计算机科学系的教学计划中明确规定“数据结构”为一门课程，但没有明确规定该课程的内容范围。当时，数据结构几乎和图论，特别是表和树的理论是同义语。随后，数据结构这个概念被扩充到包括网络、代数、集合论、关系等现在称之为“离散数学结构”的那些内容，它们同现在的“数据结构”的某些内容混在一起，总称为“数据结构”。由于数据必须在计算机中进行处理，因此，不能局限于研究数据本身的数学概念，还必须考虑数据的物理结构。这就进一步扩大了数据结构的内容。自从 1968 年美国研究计算机科学的著名教授 D. E. Knuth 所著的“计算机程序设计技巧”(The Art of Computer Programming)问世以后，才逐渐把数据的逻辑结构、物理结构以及每种结构所定义的运算作为组成“数据结构”课程的主要内容。近年来，由于数据库系统的不断发展，在数据结构课程中又增加了文件管理，特别是大型文件的组织等方面的内容。

数据结构与数学、计算机硬件、特别是计算机软件有着密切的关系。它是计算机专业的一门核心课程，是编译原理、操作系统、数据库、人工智能等课程的基础。同时又广泛用于信息科学、系统工程、应用数学以及各种工程技术领域。

数据结构在计算机科学中有着十分重要的地位。它有自己的理论、研究对象和应用范围，而且其研究的内容正在不断扩充和深化。而作为一门课程、一本教材，因受特定的对象和时期的限制，因此不能全面地反映数据结构的全貌。但值得注意的是，数据结构是新兴的

学科，正值方兴未艾，蓬勃发展的阶段。一方面，面向各专门领域中特殊问题的数据结构得到研究和发展，如多维图形数据结构等；另一方面，从抽象数据类型的观点来讨论数据结构，已成为一种新的趋势，越来越被人们所重视。

习　题

1. 指出下列集合的元素是什么？
   (1) 我国四大发明的集合。
   (2) 小于 24 并且是 5 的倍数。
   (3) $B=\{w \mid |w|<3, w\in Z\}$。
   (4) 方程 $y^2-5y+4=0$ 在实数范围内的解。
2. 下列每一对集合是否相等？
   (1) $\{-1,1\}$，$\{1,-1\}$
   (2) $\{1,2,3\}$，$\{2,1,4\}$
   (3) $\{a,b,c\}$，$\{b,c,d,e,a\}$
   (4) $\{x \mid x^2-4x+3=0, x\in R\}$，$\{1,3\}$
3. 若集合 $A=\{a,b,c\}$，下面的记法哪些是正确的？哪些是不正确的？
   $a\in A$；$b\subset A$；$\{c\}\subset A$；$A\subset A$；$\Phi\subset A$；$\{c\}\in A$；$\Phi\in A$
4. 已知：$A=\{0,2,4,6,8\}$，$B=\{1,3,5,7,9\}$，$C=\{2,5,6,9\}$，求：
   $A\cap B, A\cap C, B\cap C, A\cup B, A\cup C, B\cup C, A-B, B-C, C-A$
5. 设 $A=\{2,3,4,6\}$、$B=\{a,b,c\}$，求 $A\times B$ 和 $B\times A$。
6. 简述下列术语：数据、数据元素、数据对象、数据结构、存储结构和数据类型。
7. 数据结构研究的主要内容是什么？

第二日　算法的描述与分析

数据结构是一门实践性很强的学科，通过该课程的学习，读者能运用数据结构的技巧更好地进行算法和程序设计，所以我们在讨论各种数据结构的基本运算时，都给出了相应的算法。对于算法的描述，我们力求做到通俗易懂，适于自学，所以采用文字框图进行描述。读者在掌握和理解了框图所示的设计思想后，可以较方便地使用自己熟悉的算法语言来编制程序。另外，考虑用 PASCAL 语言来编写程序可以较好地体现程序的结构，并且简明易学，具有实用价值，所以本书中大部分算法在给出文字框图的同时还给出了用 PASCAL 语言编写的源程序片断。下面，我们将对本书框图中使用的符号及 PASCAL 语言进行介绍。

2.1　PASCAL 语言初步

一、程序结构

首先我们看一个 PASCAL 语言程序的例子，该程序要求：输入两个数，并且计算两数之和，然后再输出结果。

```
PROGRAM example(INPUT,OUTPUT);
  {This is used for s=x+y}
VAR x,y,s:ingeger;
BEGIN
  read (x,y);
  s:=x+y;
```

```
    writeln('s=',s)
END.
```

以上是一个用 PASCAL 语言编写的简单程序，它体现了 PASCAL 语言程序的结构由程序首部和分程序组成，而分程序又分为说明部分和语句部分。

1. 程序首部

程序首部用于指定程序的名字和列出程序中用到的文件。它由程序标识符、程序名、程序参数所组成。其中，程序标识符是 PASCAL 语言的标志，是每个 PASCAL 语言程序都有的，用 PROGRAM 表示。程序名事实上是一个标识符，是用户为程序安排的名字，就像一个人有一个人名一样，每个程序都有一个程序名为标记。如上例中的 example 就是一个程序名。在 PASCAL 语言中，标识符是一个字母开头的后跟若干字母或数字的字符串。例如：x，x12，x3yz 等都是标识符，而 123x 就不是标识符。程序参数用来表示该程序同外界的联系。它们一般是文件名，最常用的文件为 INPUT、OUTPUT，表示标准的输入/输出文件。

程序首部有时还带有程序的注释部分，用于对程序的名称、类型、功能、编写日期等进行描述。如上例中的{This is used for s=x+y}。

2. 程序的说明部分

在 PASCAL 程序中允许用户自己定义标号、常量、类型、变量、过程和函数等，这些都必须首先在程序的说明部分加以说明，然后才能在程序的执行部分引用。程序的说明部分应遵循如下次序：

(1)标号说明部分；

(2)常量定义部分；

(3)类型定义部分；

(4)变量说明部分；

(5)过程与函数说明部分。

下面予以逐一介绍。

■ 标号说明部分

在 PASCAL 语言中提供了 GOTO 语句。GOTO 语句使程序不再顺序地执行程序的下一个语句，而转去从指定标号的语句开始执行。标号规定为四位以内的无符号整数。标号和冒号“:”一起放在一个语句的前面，构成一个带标号的语句。例如：

```
99: Writeln('ERROR DETECTED');
```

GOTO 语句的例子为：GOTO 99

PASCAL 语言规定，语句前的标号以及 GOTO 语句的标号都必须在标号说明部分中说明。标号说明的一般形式为：

```
LABEL    标号表;
```

其中，标号表由一系列标号组成，标号之间用逗号隔开。例如：

```
LABEL 12 ,100;
```

说明了两个标号 12 和 100。

■ 常量定义部分

在程序中我们常会用到一些具体不变的数据，称为常量。例如，数学常数 π 的近似值 3.1415926。在程序中允许在哪里用到一个常量，就在那里写出这个常量的值。但是，为了程序描述清晰，书写方便，修改容易，程序员宁愿在用到常量的地方使用一个标识符代替常量的值。常量定义就是用来引入一个标识符作为一个常量的同义词。凡是在程序中出现这个标识符，就等价于在那里直接写上相应的常量值。如果要修改某一常量，只要修改一下它的常量定义即可。常量定义的一般形式为：

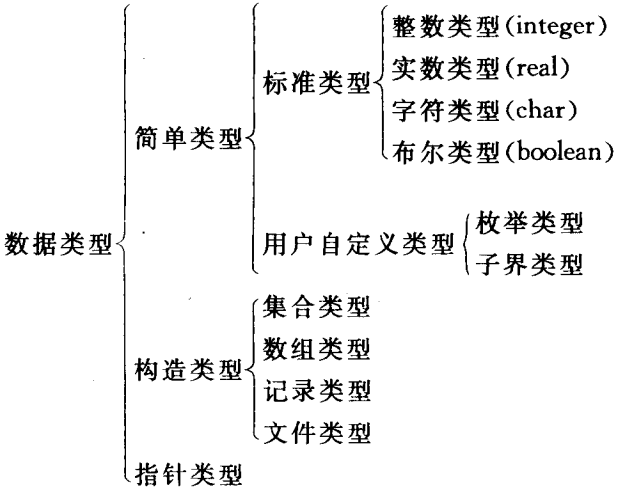
```
CONST    标识符 1=常量值 1;
         标识符 2=常量值 2;
                ⋮
         标识符 n=常量值 n;
```

下面是一个常量定义的实例，它定义了三个常量：Pi、Epsilon、Max：

```
CONST    Pi=3.1415926 ;
         Epsilon=1E-6 ;
         Max=100;
```

■ 类型定义部分

PASCAL 语言的数据类型可以图示如下：



对四种标准类型，程序员不用预先定义就可以引用。此外，程序员也能根据需要自己定义类型，并给一个类型标识符。类型定义的一般形式为：

```
TYPE  类型标识符 1=类型 1;
      类型标识符 2=类型 2;
            ⋮
      类型标识符 n=类型 n;
```

其中，类型标识符是指所定义类型的名称，类型指已定义的各类型，具体定义后面介绍。

■ 变量说明部分

在程序中除了常量外还经常会用到另外一种数据，它的值在程序执行期间可以根据需要而变化，我们称之为变量。每个变量必须有一个标识符，并属于某一类型。变量说明就是把变量标识符和它的类型联系起来。变量说明的形式为：

```
VAR   标识符表 1:类型名 1;
      标识符表 2:类型名 2;
```

⋮

标识符表 n:类型名 n;

其中,标识符表由一个或多个标识符组成,标识符之间用逗号分隔。

例如:

```
VAR i,index:integer;
    root1,root2:real;
    ch:char;
    found:boolean;
```

在这个例子中,把 i,index 说明成整型变量;root1,root2 说明成实型变量;ch 说明成字符型变量;found 说明成布尔型变量。

■ 过程和函数定义部分

在 PASCAL 语言中,不严格地说,过程(或函数)说明是定义了逻辑上相关的语句序列,用于描述一组复合的动作。并且给这个语句序列取一个名字,即过程(或函数)名。过程(或函数)定义的结构和程序类似。

如下是一个过程定义的例子:

```
PROCEDURE add(x,y:integer; VAR sum:integer);
BEGIN
  sum:=x+y;
END;
```

它体现了 PASCAL 语言中过程定义的结构由过程首部和分程序组成。其中分程序的定义和程序中的分程序一样分为说明部分和语句部分。过程首部用于指定过程的名字和列出过程中用到的参数,它由过程标识符、过程名、形式参数所组成。其中,过程标识符为 PROCEDURE,是每个过程都有的;过程名是一个标识符,是用户为过程起的名字,上例中的 add 就是一个过程名;形式参数用来表示该过程同外界的联系,上例中定义了三个参数 x、y 和 sum ,它们都为整型数。其中,x 和 y 称为值参,它只能从外界把数据送进来,而不能把数据送出去;sum 称为变参,它不仅可从外界把数据送进来,而且通过它能把数据送出去。

下面是一个函数定义的例子：

```
FUNCTION eq(x,y:integer):boolean;
BEGIN
  IF x=y then return TRUE
        ELSE return FALSE
END;
```

函数的结构和过程类似也由函数首部和分程序组成。其中分程序的定义和程序中的分程序一样又分为说明部分和语句部分。函数首部同过程首部不同之处在于函数的标识符是 FUNCTION；并且函数需要指明返回值的类型。上例中函数 eq 返回值为布尔类型。在函数中要使用 return 语句返回函数值。

二、PASCAL 语言的语句

PASCAL 程序的语句部分由如下形式定义：

```
BEGIN
  语句 1;
  语句 2;
    ⋮
  语句 n;
END
```

其中，语句是指如下所述的执行性语句。

1. 赋值语句

一般形式：

变量：＝表达式：

其作用就是计算右端的表达式，并把结果赋给左端的变量。其中表达式可以是算术表达式、关系表达式和布尔表达式。

PASCAL 语言的算术表达式是整型量或实型量与算术运算符的合法组合。其中算术运算符为：

＋(加)、－(减)、＊(乘)、DIV(整除)、MOD(求余)、/(实数除)。

它的规定如下：

(1) +(加)、-(减)、*(乘)是三种常用的运算,当两个操作数都是整型数时,运算结果是整型数;其中,若有一个实数或两个都是实数,则结果为实数。

(2) /(实数除)不管操作数是实数还是整数,结果均为实数。DIV(整数除)和 MOD(求余)运算的两个操作数必须都是整型数,结果也是整数。

PASCAL 语言的关系表达式是算术表达式与关系运算符的合法组合。关系运算符主要有下列六种:

=(等于),<>(不等于),<(小于),>(大于),<=(小于等于),>=(大于等于)关系运算的结果为布尔型数值:真(TRUE)值和假(FALSE)值。一个关系表达式实际上表示了一个判定条件,若判定条件满足,则关系表达式的结果为 TRUE,否则为 FALSE。

PASCLA 语言的布尔表达式是由布尔型数据和布尔(逻辑)运算符组成。其中布尔型数据可以由产生布尔型数值的关系表达式代替;布尔运算符主要有下列三种:NOT、AND 以及 OR 运算。其运算法则为:

A	B	A AND B	A OR B	NOT A
TRUE	TRUE	TRUE	TRUE	FALSE
FALSE	TRUE	FALSE	TRUE	TRUE
TRUE	FALSE	FALSE	TRUE	FALSE
FALSE	FALSE	FALSE	FALSE	TRUE

需要注意的是:

(1) 数学上的表达式:A≥B≥C≥D,在 PASCAL 语言必须写成:

(A>=B) AND (B>=C) AND (C>=D)

(2) 逻辑运算必须写成诸如:

A AND B,A OR B,NOT A;

(3) 运算优先次序为:

① 圆括号,由内向外逐层展开;

② NOT ;