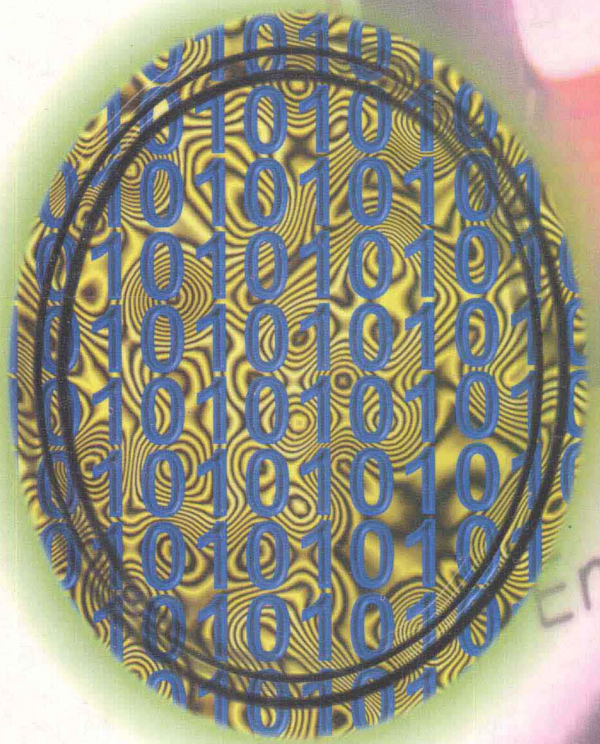
# 数据结构

左春荣　主编

中国物资出版社

# 数 据 结 构

左春荣 主编

中国物资出版社

**图书在版编目(CIP)数据**

数据结构/左春荣主编. —北京:中国物资出版社,1997.12

ISBN 7-5047-1343-0

Ⅰ.数… Ⅱ.左… Ⅲ.数据结构 Ⅳ.TP311.12

中国版本图书馆 CIP 数据核字(97)第 23303 号

**责任编辑**:印　丽
**特约编辑**:沙　金
**装帧设计**:木　贞
　　王　磊
**责任校对**:马永开

中国物资出版社出版发行
(北京市西城区月坛北街 25 号　邮编:100834)
全国新华书店经销
北京科发文化交流有限公司激光照排
安徽省蚌埠市红旗印刷厂
787×1092 毫米　1/16　印张:13.25　字数:331 千字
1998 年 1 月第 1 版　1998 年 1 月第 1 次印刷
印数:00001—10000 册
**ISBN 7—5047—1343—0/TP·0016**
**定价:14.80 元**

# 总　序

从1946年第一台计算机诞生至今已历经半个多世纪。计算机的出现和广泛应用，标志着人类社会的一次大飞跃，信息时代的一次大转折，生存方式的一次大变革，现代文化的一次大融汇。随着计算机技术的飞速发展和广泛普及，其应用已遍及社会生活的各个领域。由于计算机技术已进入到我们生活中的方方面面，人类社会的生活方式、思维方式以及时空观念等各个方面都已经发生了深刻的历史性的巨变。

随着信息化社会的发展，人们对信息交流的要求越来越高。世人已普遍公认：哪个国家的信息化程度高，其经济竞争力就越强，科技发展就越快，办事效率就越高，对下一代的教育条件就越好。信息化的进一步深入需要我们培养一大批高素质人才。当今社会，熟练掌握计算机应用技术已成为高素质人才的必备条件。因此，一个十分艰巨的任务，就是要使受教育者具有使用计算机的能力和与之相适应的计算机文化素质。如果我们的知识结构和文化修养准备不足，就不能适应时代和社会发展的需要。

一本好书，是人生旅途的一掬甘泉；一套好教材，是教学成功的必要条件。广大学子和读者殷殷所望，无非是博得一艺，学有所用。本着对读者负责的精神，我们组织北京电子科技学院、集美大学、合肥工业大学、安徽大学、安徽财贸学院、江苏广播电视大学等高等院校和部分大中专学校教学经验丰富的教师，以及一些具有较高理论基础和软件开发经验的计算机技术人员共同合作编写了这套计算机及应用专业教材。为保证教材的质量，我们还聘请了一批学术造诣较深的专家、教授作为本套教材的主审和顾问。本套教材具有以下几个方面的特点：

首先，作为一套计算机专业教材，必须保证整个计算机知识体系的完整性。本套教材包括必修课17种，选修课和配套教材3种，基本上涵盖了目前大中专院校计算机及应用专业所必修或选修的课程内容。各种教材在编排上，既注意到内容上的连贯性，又保证了教学上的相对独立性。

其次，在教材内容的组织上，注重介绍和汲收当今计算机领域的一些新技术和新知识，摒弃了传统教材中一些过时的内容，这些变化在各本教材中都得到程度不同的体现。本套教材编写时既参照了有关部委计算机及应用专业教学大纲，又参考了“程序员考试大纲”和“全国计算机水平等级考试大纲”的内容。因此本套教材既适合作为各级各类院校计算机及应用专业教材，亦可作为计算机水平等级考试学习用书。个别教材之间内容上的重复，是为了照顾部分读者单独选用单本教材的需要，敬请广大读者予以谅解。

再次，考虑到各校教学的特点，本着学以致用的原则，在本套教材编写中我们始终贯彻“由浅入深，理论与实践相结合”的原则，以阐明要义为主，辅之以必要的例题、习题和上机实习，以便使读者尽快领悟和掌握。

在本套教材编写过程中，各位作者付出了艰辛的劳动，教材编委会的各位专家和教授对各本教材的内容进行了认真的审定和悉心的指导。在本套教材出版过程中我们自始至终得到中国物资出版社领导和编辑以及印制单位的大力支持和帮助。本套教材承蒙中国科学院计算技

术研究所、国家智能计算机研究开发中心王川宝、高文、中国机械科学研究院江波等同志进行了较为细致的终审终校工作。正是由于各方面的通力配合，才使得本套教材得以顺利出版和发行。书中参考、借鉴了国内外同类教材和专著，在此一并表示感谢。

近年来，计算机技术发展日新月异，异彩纷呈，许多新的概念和内容都在不断扩展之中，囿于编者学识和水平，书中疏漏、错误之处还望广大读者不吝批评指正，以便对本套教材不断修订完善。

计算机及应用专业教材编委会

1998 年 1 月

# 附：计算机及应用专业教材编委会名单

## 顾　　问

（以姓氏笔划为序）

| | |
|---|---|
| 王仲文 | 北京电子科技学院院长、教授 |
| 韦　穗 | 安徽大学副校长、教授 |
| 张全寿 | 铁道部电子计算中心主任、北方交通大学教授 |
| 李文忠 | 全国计算机基础教育学会副理事长、东南大学教授 |
| 杨善林 | 合肥工业大学副校长、教授、博士生导师 |
| 辜建德 | 集美大学校长、教授 |
| 魏余芳 | 西南交通大学教授 |

## 编　　委

| | |
|---|---|
| 鄂大伟 | 集美大学副教授 |
| 李树德 | 北京电子科技学院教授 |
| 刘　锋 | 安徽大学副教授 |
| 王川宝 | 中国科学院计算技术研究所、国家智能计算机研究开发中心硕士研究生 |
| 高　文 | 中国科学院计算技术研究所、国家智能计算机研究开发中心博士研究生 |
| 江　波 | 中国机械科学研究院硕士研究生 |
| 屈道良 | 上海铁路局蚌埠分局高级工程师 |
| 蒋翠清 | 上海铁路局蚌埠分局高级工程师 |

# 前　言

“数据结构”是计算机及应用专业的一门核心课程，它是“操作系统”、“数据库”、“软件工程”、“编译原理”等课程的基础。

作为一本大中专教材，本书选材基本上涵盖了有关数据结构的主要内容。全书共分八章。第一章为“绪论”，介绍了数据结构的基本概念和本书采用的算法语言及评价算法的一些原则等；第二章为“线性表”，介绍了线性表、栈、队列、串、广义表等基本数据结构以及有关的算法实现和应用举例；第三章为“树”，介绍了有关树和二叉树的基本概念，描述了二叉树的遍历算法，给出二叉树的几种物理表示和主要算法的实现以及树的应用举例；第四章为“图”，给出了图的多种表示方法，并着重介绍了若干重要的算法，如深度优先遍历、广度优先遍历、最短路径问题以及拓扑排序等；第五章和第六章分别为“检索”和“排序”，包含了在数据处理领域几种主要的检索方法和内排序方法；第七章为“文件”，介绍了有关文件的概念及其逻辑特性，简述了几种常见的文件组织方法；第八章为“存贮管理”，给出了几种存贮分配策略，介绍了两种存贮回收方法。

本书中的算法用类 PASCAL 语言描述，各章之后附有习题。考虑到大中专学生的接受特点，本书在描述各种数据结构和有关算法时注重实例讲解，着重于实际应用，分析算法时，略去了一些理论推导和证明。需要进一步学习的读者可以查阅书末参考文献中所列的书目。

本书编写得到哈尔滨工业大学、合肥工业大学、安徽财贸学院等有关高校专家教授的帮助，书中参考引用、借鉴了国内外同类著作，全书在终审终校过程中得到马永开、鄂大伟、屈道良、梁昌勇、薛萍等同志的大力帮助，在此一并致谢！

由于编者学识水平和教学经验不足，书中错误和不足之处在所难免，敬请有关专家和广大读者不吝批评指正，以便不断修订完善。

编　者

1998 年 1 月

# 目　录

# 第一章

# 绪 论

## §1.1 数据结构的基本概念

### 1.1.1 什么是数据结构

当前，计算机已深入到人类社会的各个领域。计算机不仅应用于科学计算，而且更多地应用于控制、管理及数据处理等非数值计算的处理工作。与此相对应，计算机加工处理的对象也由纯粹的数值发展到字符、表格、图像、声音等各种类型的数据。对计算机科学稍有了解的人都知道，计算机必须通过程序才能进行信息处理。要使计算机具有较高的工作效率，就需要设计出较好的程序。有人曾概括成这样一个公式：程序＝算法＋数据结构，这就是说，要设计出好的程序，除了要设计出好的算法之外，还必须采用合适的数据结构。那么，什么是数据结构呢？我们先举几个例子。

【例 1】 某汽车配件销售公司的配件库存表形式为：

| 配件名称 | 型号规格 | 价格 | 库存量 | 供应商 |
|---|---|---|---|---|
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |

若利用计算机进行库存管理，则计算机处理的便是描述汽车配件的各项信息。我们可以根据不同的要求来组织这张表的结构，访问表中的数据。例如，随意检索某种配件的库存量，可建立按配件名称排序的索引表，若检索库存量在给定值以上的配件，可建立按库存量排序的索引表。

该例表明，对数据进行不同的组织，形成不同的结构，可以直接影响到处理问题的效率和算法的设计。这是一个数据结构的问题。

【例 2】 企业的物资供应部制定物资采购计划时，通常要按类别列出每种物资的计划。如何组织好众多的类别和物资的明细数据，使人能一目了然，这也是一种数据结构问题。通常，人们都是以树形结构表来组织这些数据。如图 1—1 所示。

这种结构，类别分明。若检索∅10 管钢的采购计划，根据它所属的大中小类，很快就能找到∅10 管钢的计划数。

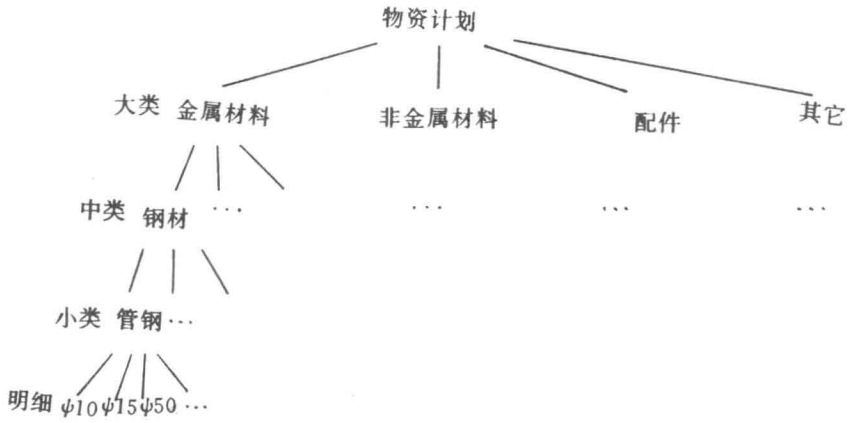



图 1—1

【例 3】 图 1－2 所示的交叉路口有 5 条通路：A，B，C，D，E。其中 C 和 E 是单行路，因而共有 13 个“拐弯”。有些“拐弯”如 A→B 和 E→C 可以同时进行，而象 A→D 和 E→B 就不能同时进行。

问题：需设几种颜色的交通灯才能既使车辆相互之间不碰撞，又能达到车辆的最大流通？

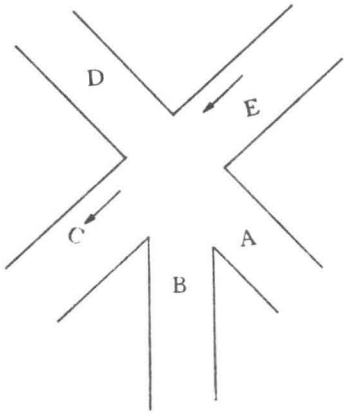



图 1－2　一个交叉路口

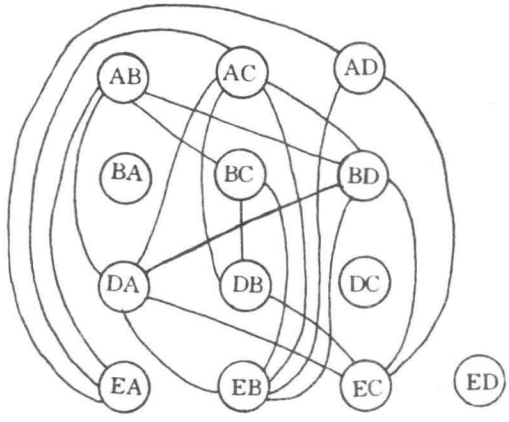



图 1－3　交叉路口的数学模型

通常，这类问题的数学模型是一种称之为“图”的数据结构。每个“拐弯”对应图中的一个顶点，若两个“拐弯”不能同时进行，则它们对应的顶点是相邻的，即它们之间有一条边相连接。图 1－3 所示交叉路口的数学模型，如何用最少数目的指挥灯实现交叉路口的安全管理问题就可化为图论中的着色问题：用最少数目的不同颜色对图中的每个顶点着色，使得任何两个相邻顶点的颜色都不同。

上述几例告诉我们什么是数据结构的概念。可见，描述非数值计算问题的数学模型不再是数学方程，而是诸如表、树、图之类的数据结构。如何存入、取出这些数据，如何去组织数据的逻辑结构和物理结构，对某个实际问题如何去选择合适的数据结构、设计算法、编制程序等问题，就是数据结构这门课程所要研究的内容。由此不难看出，数据结构是主要研究数据逻辑关系和数据表示的一门学科。

### 1.1.2 一些概念和术语

下面我们介绍一些概念和术语，它们在以后的章节中将会多次出现。

**数据**(data)是对客观事物及其活动的符号的描述。在计算机领域，是指能够被计算机输入、存储和处理的一切信息。因此，对计算机科学而言，数据的含义极为广泛，象声音、图象等也可以输入到计算机中进行处理，所以它们也可归为数据的范畴。

**数据元素**(data element)是一个数据整体中相对独立的单位。如对于一个文件来说，每个记录就是它的数据元素；对于一个数组来说，每一个单元就是它的元素。有时，一个数据元素可由若干个数据项(data item)组成，例如，例1中表的每一行信息为一个数据元素，而每一行都包含若干数据项(如配件名称、价格等)。数据项是数据的不可分割的最小单位。

**数据结构**(data structure)是相互之间存在一种或多种特定关系的数据元素的集合。从前面几例可以看出，描述客观事物及其活动的数据之间必然存在着联系，由于这种联系是内在的或根据需要人为定义的，所以被看作是逻辑上的联系，因此，又把数据结构称为数据的逻辑结构，把数据结构在计算机中的存储表示称为数据的物理结构，又称存储结构，它包括数据元素的表示和关系的表示。由于存储表示的方法有顺序、链接、索引、散列等多种，所以一种数据结构可表示成一种或多种物理结构。

根据数据元素之间关系的不同特性，通常有下列四种基本结构。

1. 集合　结构中的数据元素之间除了"属于同一个集体"的关系之外，别无其它关系。
2. 线性结构　结构中的数据元素之间存在一对一的关系。
3. 树结构　结构中的元素之间存在一对多的关系。
4. 图型结构　结构中的元素之间存在多对多的关系。

为了更确切地描述数据结构，通常采用二元组表示：

$$B=(K,R)$$

B是一种数据结构，它由数据元素的集合K和K上二元关系的集合R所组成。其中：

$$K=\{k_i \mid 1\leqslant i\leqslant n, n\geqslant 0\}$$

$$R=\{r_j \mid 1\leqslant j\leqslant m, m\geqslant 1\}$$

$k_i$ 表示第i个数据元素，n为B中数据元素的个数，特别地，若n=0，则K是一个空集，因而B也就无结构而言，或者说它具有任何结构；$r_j$ 表示第j个二元关系(以后均简称关系)，m为K上关系的个数。

在以后所讨论的数据结构中，一般只讨论m=1的情况，即R中只包含一个关系(R={r})的情况。对于包含有多个关系的数据结构，可分别对每个关系进行讨论。

K上的一个关系r是序偶的集合，对于r中的任一序偶<u,v>(u,v∈K)，把u叫做序偶的第一元素，把v叫做序偶的第二元素，称序偶的第一元素为第二元素的直接前驱，简称为前驱，称第二元素为第一元素的直接后继，简称后继。如在<u,v>的序偶中，u为v的前驱，v为u的后继。

数据结构还能够利用图形形象地表示出来，图形中的每个结点对应着一个数据元素，两结点之间的带箭头的连线对应着关系中的一个序偶，其中序偶的第一元素为有向边的起始结点，第二元素为有向边的终止结点。下面根据表1－1构造一些典型的数据结构。

表 1－1　　财务处人员一览表

| 职工号 | 姓名 | 性别 | 职务 | 基本工资 | 科室 |
|---|---|---|---|---|---|
| 01 | 赵利 | 女 | 处长 | 520 | |
| 02 | 王刚 | 男 | 科长 | 380 | 计划科 |
| 03 | 方敏 | 男 | 科长 | 460 | 财务科 |
| 04 | 刘旭 | 男 | 科员 | 410 | 计划科 |
| 05 | 杨芳 | 女 | 科员 | 290 | 计划科 |
| 06 | 马玉 | 女 | 科员 | 340 | 计划科 |
| 07 | 胡立 | 男 | 科员 | 350 | 计划科 |
| 08 | 李晓 | 男 | 科员 | 500 | 财务科 |
| 09 | 周华 | 男 | 科员 | 480 | 财务科 |
| 10 | 钟伟 | 男 | 科员 | 470 | 财务科 |
| 11 | 李建 | 男 | 科员 | 320 | 财务科 |
| 12 | 肖虹 | 女 | 科员 | 450 | 财务科 |

表中有 12 条记录，每条记录由六个数据项组成，由于每条记录的职工号各不相同，所以可把每条记录的职工号作为该记录的关键字，在下面的例子中，将用记录的关键字代表整个记录。

【例 4】　一种数据结构 linearity＝(K,R)，其中：

K＝{01,02,03,04,05,06,07,08,09,10,11,12}

R＝{r}

r＝{<01,08>,<08,09>,<09,10>,<10,03>,<03,12>,<12,04>,<04,02>,
<02,07>,<07,06>,<06,11>,<11,05>}

对应的图形如图 1－4 所示。

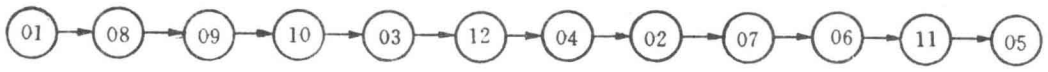



图 1－4　线性结构示意图

结合表中数据不难看出，r 是按职工基本工资从大到小排列的关系。

在数据结构 linearity 中，每个数据元素有且只有一直接前驱元素(除其中第一个元素 01 外)，有且只有一个直接后继元素(除其中最后一个元素以外)。这种数据结构的特点是数据元素之间的一对一联系，即线性关系，我们把具有这种特点的数据结构叫做线性结构。

【例 5】　一种数据结构 tree＝(K,R)，其中：

K＝{01,02,03,04,05,06,07,08,09. 10,11,12}

R＝{r}

r＝{<01,02>,<01,03>,<02,04>,<02,05>,<02,06>,<02,07>,<03,08>,<03,09>,<03,10>,<03,11>,<03,12>}

对应的图形，如图 1－5 所示。

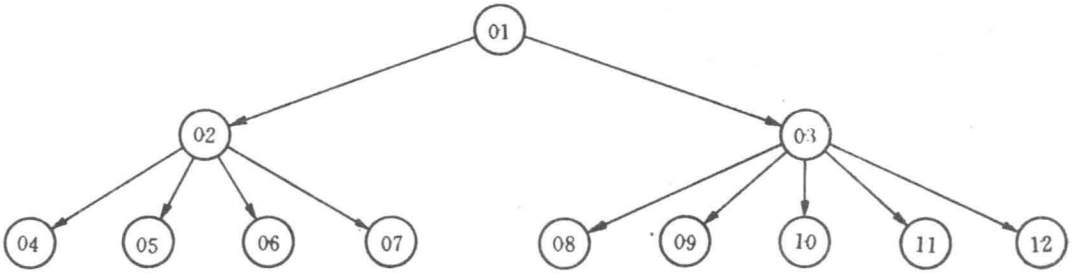



图 1－5 树结构示意图

结合表中数据不难看出，r 是职工之间领导与被领导的关系。

图 1－5 象一棵倒画的树，在这棵树中，最上面的一个没有前驱只有后继的结点叫做树根结点，最下面一层的只有前驱没有后继的结点叫做树叶结点，除树根和树叶之外的结点叫做树枝结点。在一棵树中，每个结点有且只有一个前驱结点（根结点除外），但可以有多个后继结点（树叶结点可看作为具有 0 个后继结点）。这种数据结构的特点是数据元素之间的一对多联系，我们把具有这种特点的数据结构叫做树结构。

【例 6】 一种数据结构 graph＝(K,R)，其中：

K＝{01,02,03,04,05,06}

R＝{r}

r＝{<01,02>,<02,01>,<02,03>,<03,06>,<06,03>,<06,02>,<02,06>,<02,05>,<05,02>,<05,04>,<04,05>,<04,01>,<01,04>}

对应的图形如图 1－6 所示。

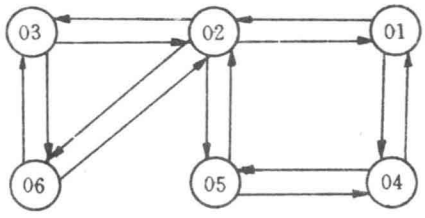



图 1－6 图型结构示意图

从图 1－6 可以看出，r 是 K 上的对称关系。为了简化起见，我们把<x,y>和<y,x>这两个对称序偶用一个无序对(x,y)或(y,x)来代替；在图中，把 x 结点和 y 结点之间两条相反的有向边用一条无向边来代替。这样关系 r 可改写为：

r＝{(01,02),(02,03),(03,06),(06,02),(02,05),(05,04),(04,01)}

对应的图形如图 1－7 所示。

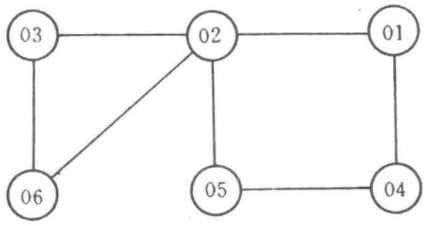



图 1－7　图型结构示意图

从图 1－6 和图 1－7 可以看出，结点之间是多对多的联系，也就是说，每个结点可以有任意多个前驱结点和任意多个后继结点。我们把具有这种特点的数据结构叫做图型结构。

从图型结构、树结构和线性结构的定义可知，树结构是图型结构的特殊情况，线性结构是树结构的特殊情况。为了区别于线性结构，我们把树结构和图型结构统称为非线性结构。

**【例 7】**　一种数据结构 B=(K,R)，其中：

$K=\{k_1,k_2,k_3,k_4,k_5,k_6\}$

$R=\{r_1,r_2\}$

$r_1=\{<k_3,k_2>,<k_3,k_5>,<k_2,k_1>,<k_5,k_4>,<k_5,k_6>\}$

$r_2=\{<k_1,k_2>,<k_2,k_3>,<k_3,k_4>,<k_4,k_5>,<k_5,k_6>\}$

若用实线表示关系 $r_1$，虚线表示关系 $r_2$，则对应的图形如图 1－8 所示。

从图 1－8 可以看出数据结构 B 是一种非线性的图型结构。但是，如果只考虑关系 $r_1$ 则为树结构，若只考虑关系 $r_2$ 则为线性结构。

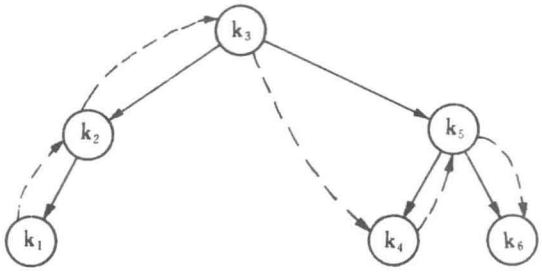



图 1－8　带有两个关系的数据结构示意图

**数据类型**(data type)　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 PASCAL 语言中的整数类型，其值集为区间[－maxint，maxint]上的整数(maxint 是依赖于计算机的最大整数)，定义在其上的一组操作为：加、减、乘、除等。

**抽象数据类型**(Abstract Data Type－ADT)　是指一个数学模型及定义在该模型上的操作集合。抽象数据类型的定义取决于它的逻辑特性，与其在计算机内部如何表示和实现无关。抽象数据型的实现是将其中的数学模型用一种程序语言中适当的数据结构来表示，并且操作集合用一组相应的过程或函数来代替，而数据结构是利用语言中现存的基本数据类型和数据构造工具构成的。例如数组和记录是 PASCAL 语言中两个主要的数据构造工具。

**算法**(algorithum)　是对特定问题求解步骤的一种描述。在计算机科学中，一个算法实质

上就是根据处理问题的需要，在数据的逻辑结构和存储结构的基础上进行的运算过程。由于数据的逻辑结构和存储结构不是唯一的，所以处理同一个问题的算法也不是唯一的。而且，即使对于相同的逻辑结构和存储结构而言，算法的设计思想和技巧不同，编写出的算法也大不相同，在计算机上用程序实现时执行的效率也有差别。我们学习数据结构这门课的目的，就是要能够针对数据处理问题的需要，选择合适的逻辑结构和存储结构，从而设计出效率比较高的算法。

## §1.2 算法的描述

算法需要用一种语言来描述。本书在描述算法时，采用一种与PASCAL 语言相仿，但较其简单，并适当加入一些中文语句的语言，即类PASCAL 语言作为描述工具。类PASCAL 语言与标准PASCAL 语言有所不同。它忽略了PASCAL 语言中语法规则的一些细节，同时增加了一些功能较强的语句，这样使得描述出来的算法可读性大大增强。读者不难根据这些算法编写出符合任一种PASCAL 语言或其它任一种程序语言语法规则的算法来。下面对本书中采用的算法描述语言中常用的一些语句作一简要说明。

### 1.2.1 过程和函数

本书中介绍的算法都以如下所示的过程或函数的形式表示：

```
procedure  过程名称(参数表);        function  函数名称(参数表):类型;
{算法说明}                          {算法说明}
{变量说明}                          {变量说明}
begin                               begin
  语句组                              语句组
end;{过程名称}                      end;{函数名称}
```

其中过程名称可以用英文字符串或中文来写，以尽量使读者易于看出此过程的功能为原则；参数表可含有若干值参和变参；语句组由一个或一个以上的语句组成，两语句之间用“;”作分隔符。主程序调用某过程或某过程调用另一个过程时，在需调用处只写被调用过程名称或用中文写“调用某某过程”即可，一般在过程名称后的括号中写上调用时相应的实参名称。

### 1.2.2 一般语句

赋值语句有：

简单赋值　变量名:=表达式

交换赋值　变量名↔变量名

成组赋值　(变量名 1,…,变量名 n):=(表达式 1,…,表达式 n)——顺序执行 n 个简单赋值语句。

记录名:=记录名

记录名:=(值 1,…,值 n)——分别对记录的 n 个域赋值。

输入语句写成 read(变量列表)的形式；输出语句写成 write(变量列表)的形式，有时也简单地用中文写成“输出某某量”的形式。

加、减、乘、除算术运算，分别采用＋、－、＊、/四种符号。div 运算符表示整除，mod 表示取余运算。例如(17 div 5)等于 3，(17 mod 5)等于 2。

关系运算符有＝、<>、<、>、<＝、>＝，分别表示等于、不等于、小于、大于、小于或等于、大于或等于。

### 1.2.3 条件语句

条件语句有

if　条件　then　语句 1

if　条件　then　语句 1　else　语句 2

case

　条件 1：语句 1；

　⋮

　条件 n：语句 n；

(else 语句 n＋1)

end

条件通常是由关系运算符构成的布尔表达式。语句 1 和语句 2 可以是任意的语句组，以下说明中出现的“语句”类同。

### 1.2.4 循环语句

以后的算法里用到 while 语句和 for 语句两种循环语句。while 语句形式为

while　条件　do　语句

表示只要条件成立，就执行 do 后面的语句，一直到条件不成立时才结束，for 语句一般用于已确定了循环次数的情况，其形式为

for　变量名：＝初值 to 终值 do 语句（终值大于或等于初值，增量为 1）

for 变量名：＝初值 downto 终值 do 语句（终值小于或等于初值，增量为－1）。

while 语句和 for 语句这两种循环语句都可以有多重循环。

### 1.2.5 其它语句

error(字符串) 出错处理语句，其功能是结束本次算法过程并带回表达出错信息的文字串。

| | |
|---|---|
| max(变量表) | 求最大值 |
| min(变量表) | 求最小值 |
| abs(变量) | 求绝对值 |
| eof(　) | 判文件结束 |
| eoln(　) | 判行结束 |

例如，对一维数组 A[1..n]，其中 1 和 n 分别表示该数组下标的下界和上界，若要求该数组中元素的最大值和最小值，并把它们的下标分别赋给变量 vmax 和 vmin，则相应的算法描述如下：

```
  procedure SEARCH(A:array[1..n] of integer;n:integer;var vmax,vmin:
  integer);
var i:integer;
begin
  vmax:=1;
  vmin:=1;
  for i:=2 to n do begin
    if A[i]>A[vmax] then vmax:=i;
    if A[i]<A[vmin] then vmin:=i
  end {for}
end;{SEARCH}
```

## §1.3 算法评价

### 1.3.1 评价原则

对于解决同一个问题，往往有多种不同的算法。评价算法的目的，一方面是从解决问题的多种算法中选出较为合适的一种，另一方面是有助于对现有的算法进行改进。一般从以下四个方面评价一个算法。

(一)正确性

算法应当满足具体问题的要求，在合理的输入数据下，能在有限的时间内得出正确的结果。这是算法最起码应该达到的要求，也是设计和评价一个算法的首要条件。要从理论上证明一个算法是否正确并不是一件容易的事，也超出了本书范畴，在此不再赘述。一般情况下，算法对于精心选择的典型、苛刻而带有刁难性的几组输入数据能够得出满足要求的结果就认为是正确的。

(二)运行时间

对于同一种问题如果有多个算法可以解决，运行时间短的算法效率高。影响运行时间的因素是多方面的，如机器的运行速度、编译程序产生的目标代码的质量、程序的输入等。通常运算时间是指运行所需的一些基本操作的次数。例如所需的比较和移动次数，所需的加法和乘法次数等。显然，一个算法所需的运算时间与所解决问题的规模大小有关。关于算法的运行时间的度量将在后面章节重点讨论。

(三)占用存储空间

一个算法所占用的存储空间，包括存储算法本身所需的存储空间、算法的输入输出数据所占用的存储空间和算法在运行过程中临时占用的存储空间三个方面。算法的输入输出数据所占用的存储空间是一定的，不随算法的不同而改变。存储算法本身所需占用的存储空间与书写算法的长短成正比。一般情况下，分析算法占用的存储空间是分析算法在运行过程中所需要的最大存储空间，被定义为算法的空间复杂性，一般以数量级的形式给出。

(四)简单性

最简单最直接的算法往往不是最有效的,其运行时间可能较长,但算法的简单性使得程序的可读性强,易于调试,容易维护,可以节省人的时间。如果程序只使用一次或很少几次,简单性是人们所追求的主要目标。然而,当一个程序要被多次重复使用时,该程序在计算机上运行时的时间代价就变成主要矛盾了。

### 1.3.2 算法的时间复杂性

前已述及,影响一个算法运行时间的因素很多,这表明使用绝对时间单位衡量算法的效率是不合适的,撇开这些与计算机硬件、软件有关的因素,可以认为一个算法"运行工作量"只依赖于问题的规模,不论一个算法是简单还是复杂,最终都是被分解成基本操作来具体执行的,因此,每个算法都对应着一定的基本操作的次数。显然,一个算法所包含的基本操作的次数越少,其运行时间也就相对地较少。所以,通常把算法中所包含的基本操作次数的多少叫做算法的时间复杂性,它是一个算法运行时间的相对量度。这里,我们把一个程序的运行时间定义成函数 $T(n)$,$n$ 是该程序输入数据量的规模,而不是某一个具体的输入。$T(n)$ 的单位是不确定的,一般把它看成是在一个特定计算机上执行的指令条数。

当我们讨论程序的运行时间 $T(n)$ 时,考虑的不是 $T(n)$ 的具体值,而是它的增长率。称函数 $f(n)$ 是 $T(n)$ 增长率的上界,是指存在一个正的常数 $C$ 和非负整数 $n_0$,当 $n \geqslant n_0$ 时,$T(n) \leqslant Cf(n)$,记为 $T(n)=O(f(n))$。

【例 1】 函数 $f(n)=3n^3+2n^2$,证明 $T(n)=O(n^3)$。

证明:取 $n_0=0$,$C=5$,则有当 $n \geqslant 0$ 时

$$3n^3+2n^2 \leqslant 5n^3$$

所以 $T(n)=O(n^3)$

我们还可以证明函数 $3^n$ 不是 $O(2^n)$,即 $3^n \neq O(2^n)$。用反证法,假设 $3^n=O(2^n)$,则存在常数 $C$ 和 $n_0$,当 $n \geqslant n_0$ 时,$3^n \leqslant C2^n$,即 $C \geqslant (\frac{3}{2})^n$,由于 $(\frac{3}{2})^n$ 是一个任意大的数,因而常数 $C$ 不存在。

一个程序运行时间的增长率将最终决定该程序在计算机上所能解决的问题的规模。因此,我们说一个运行时间为 $O(n^2)$ 的程序较之运行时间为 $O(n^3)$ 的程序具有更好的时间复杂性。至于常数因子,通常可以忽略不计。图 1—9 表明了 4 个具有不同时间复杂性的程序的运行时间。可见,对于运行时间为指数函数的程序,当输入规模达到一定程度之后,它的增长率是极快的。