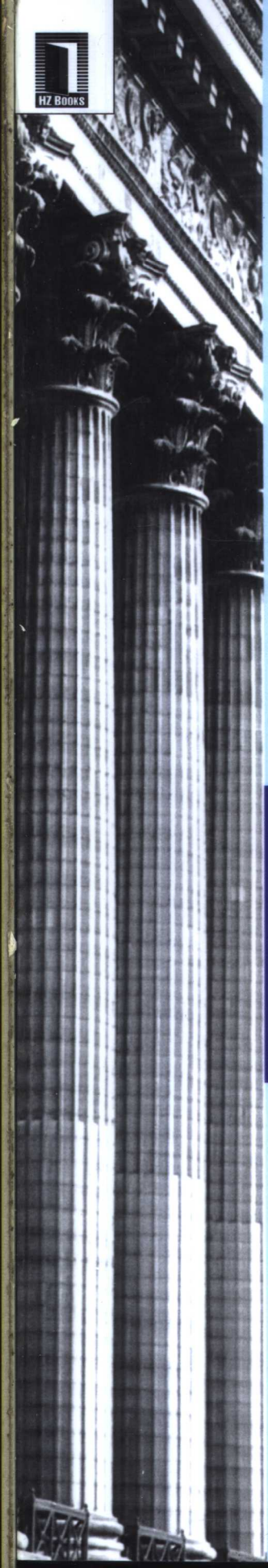


高等院校
计算机教材系列

范策　周世平　胡潇琨　等编著

算法与数据结构

(C语言版)

机械工业出版社
China Machine Press

高等院校计算机教材系列

算法与数据结构

（C语言版）

范策　周世平　胡潇琨　等编著

机械工业出版社
China Machine Press

本书以较通俗的语言，按照由易到难的原则，详细介绍了各种数据结构的基本概念、逻辑特性和物理特性，对各种结构定义了相应的抽象数据类型（ADT）以及相关的操作和算法。本书采用类C语言描述算法，并给出了各种算法的效率分析，以及这些结构在计算机科学及其他领域的应用。在各章末尾，还给出了几个算法设计的例子。

本书可作为高等院校计算机专业的教材，同时也可供计算机工程技术人员参考。

图书在版编目（CIP）数据

算法与数据结构（C语言版）/范策等编著. -北京：机械工业出版社，2004.9
（高等院校计算机教材系列）
ISBN 7-111-14620-4

Ⅰ.算… Ⅱ.范… Ⅲ.①数据结构-高等学校-教学参考资料 ②电子计算机-算法设计-高等学校-教材 Ⅳ.TP 311.12

中国版本图书馆CIP数据核字（2004）第057884号

机械工业出版社（北京市西城区百万庄大街22号 邮政编码 100037）
策划编辑：温莉芳
责任编辑：迟振春
北京中兴印刷有限公司印刷·新华书店北京发行所发行
2004年9月第1版第1次印刷
787mm×1092mm 1/16·18.25印张
印数：0 001-5 000册
定价：28.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换
本社购书热线：（010）68326294

前　言

“数据结构”是20世纪70年代开始设立的计算机科学与技术专业的基础课，现在是其重要的核心课程。随着计算机软、硬件技术的发展，计算机在各个学科和领域得到广泛应用，其当今应用的一个主要方面就是数据处理，如情报检索、数据采集、企业管理、决策分析和人工智能等。而应用到这些领域时所面临的首要问题就是对于信息量大、种类繁多、结构复杂的数据和数据关系的处理，由此必须设计高级的数据结构和数据组织，以便有效地实现数据采集、数据组织、数据存储、数据传输和数据处理。数据结构主要研究数据的逻辑结构、数据在计算机中的物理结构，以及处理不同结构数据的算法。我们研究数据结构的目的是为了学会编写更高效的程序，而高效、简洁的程序取决于数据结构和算法的设计。

现在，有关算法与数据结构的教材，一部分侧重于算法的基本思想和步骤，用伪代码描述；一部分用程序语言描述，侧重于算法的程序实现。前者侧重于对数据结构原理的阐述，不利于学生掌握算法的程序实现以及对算法的分析、比较。后者增加了对数据元素的描述，从而使得程序更难理解。

本教材对数据结构和算法用类C语言描述，数据结构和算法在设计本质上是属于底层的，在本书中并不提供可直接上机运行的源代码，而是提供一种类C语言伪代码来描述算法的基本思想和基本步骤。此外，用类C语言描述的算法容易转换为C语言程序。

本书系统全面地介绍了各种传统的数据结构，对每种数据结构及相关算法都进行了时间和空间效率的分析，强调算法与数据结构的密不可分性，引进抽象数据类型概念，将数据类型与其上的操作封装为一体，为面向对象的程序设计方法奠定了坚实的基础。此外，针对不同的抽象数据类型讨论不同的存储方法，并且研究不同存储方法的可能算法，从而体现了计算机学科方法论的理论、抽象和设计三个过程。

全书分为11章：第1章是概论，解释了从问题到程序的求解过程，以及在这一过程中抽象数据类型的表述和作用，介绍了程序的运行步骤及复杂度；第2章介绍了有关线性表的概念及其基本操作，该章是后续章节的基础；第3章在第2章的基础上讨论了操作受限的线性表——栈与队列的特点，并列举了几个应用示例；第4章简要给出了非数值处理——串的处理方法；第5章主要是关于程序设计中的一种常用数据类型——数组在计算机内部的表示和实现，同时介绍了广义表的概念；第6章描述了非线性复杂数据结构——树，它是解决决策和博弈等问题的基本结构；第7章是关于图的内容，包括有向图和无向图。图是一种比树更复杂的数据结构；第8章涉及存储管理中应用的基本方法；第9章以集合作为数据模型，讨论了查找的方法和技术；第10章介绍了一些常用的排序算法，包括内部排序和外部排序；第11章简要介绍了文件。

本书由教学一线的教授亲自主笔，根据作者多年的教学实践经验，针对算法与数据结构这门课程的特点撰写而成，目的是培养学生合理组织数据和进行优秀的算法设计的能力。本书尽量合理地安排内容顺序，教师可以根据需要自由地重新组织内容。

本教材可作为高等院校计算机科学与技术专业的教材、参考书和考研辅导，同时也可供

计算机工程技术人员参考。对于计算机科学与技术专业，可讲授64学时，对于其他专业，去掉带星号的章节，可讲授48学时。

本书的第1、2、3章由周世平同志编写，第4、10章由胡潇琨同志编写，第5、9章由孟佳娜同志编写，第6、8、11章由谭征同志编写，第7章由范策同志编写。全书由范策同志统稿，陈守孔同志编写了各章最后一节的算法设计并审阅了全书，孟佳娜同志编写了本书的电子教案。

由于作者水平所限，加上计算机科学技术的发展十分迅速，书中难免有不妥和挂一漏万之处，恳请广大读者赐教。

编　者

2004年4月于烟台大学

目　录

第1章 概 论

随着信息社会的到来，计算机的应用领域越来越广泛。信息的载体是数据，数据是计算机可以直接处理的最基本和最重要的对象。在科学计算或数据处理、过程控制以及对文件的存储和检索及数据库技术等计算机应用领域中，都需要对数据进行加工和处理。从存储空间和运行时间角度来看，程序要结构好、效率高。因此我们必须研究数据的特性和数据间的相互关系及其对应的存储表示，并设计出相应的算法，更好地实现程序。

1.1 引言

当我们使用计算机来解决具体问题时，一般需要经过下列几个步骤：首先从具体问题中抽象出一个适当的数学模型，然后设计或选择一个解此数学模型的数据结构和算法，最后编写程序进行调试、测试，直至得到最终的解答。

1.1.1 解决问题的步骤

1. 问题

问题就是从一组输入产生一组相应的解（输出）。对于计算机要解决的问题，总有一些资源限制。例如，任何计算机程序只能使用有限的空间资源，并且必须在合理的时间内完成运行。

2. 抽象

为了使程序有效和可靠，需要有建立程序的方法。研究表明，模块化方法是建立程序的有效方法。模块化方法是将程序分解成适当的小模块，分别对小模块进行编程、测试，然后再将其合成大模块、再进行测试。显然，将大任务分解成众多的单独的小任务，会使任务的管理变得容易。当将大任务分解成小任务时，就要使用抽象的概念了。

抽象的本质就是抽取反映问题实质的东西，而忽略非实质的细节。程序设计的发展过程，实际上是一个逐步抽象的过程，往往从较高层次看要处理的问题，而忽视低层次的细节，即更关注问题本身。

有两种形式的抽象：过程抽象和数据抽象。过程抽象是面向过程的程序设计方法经常使用的抽象方法，它将整个系统的功能划分为若干个部分，强调功能完成的过程和步骤。在面向对象程序设计方法中，所使用的抽象方法是数据抽象。所谓数据抽象，是指将要处理的数据以及这些数据上的操作进行捆绑，从而形成具有不同的功能和性质的抽象数据类型。

3. 数据结构

一旦完成了对要处理的问题的抽象，就要选取合适的数据结构来解决这个问题。为此，需要研究各种典型的数据结构。

4. 算法

确定了解决问题的数据结构后，就可以为解决这个问题进行算法设计，解决一个问题常

常会有多个算法，我们总是选择最适合的算法。这就需要对算法进行分析、评估，因此就进一步要求我们建立算法分析和评估的体系和方法。

5. 程序

对于要解决的问题，设计了算法并进行了算法分析之后，就可以选用一种计算机程序设计语言编写相应的程序，并运行这个程序，给出这个问题的解。

如果所得到的输出（解）符合问题的要求，则这个解是有效的。反之，若输出（解）不满足问题的要求，则需要再重新检查、修正上述步骤，直到解决问题。

1.1.2 一个例子

下面我们通过一个例子来看一看计算机是如何解决实际问题的。

例 1.1 快速送达疫苗。

已知有邻近的五个村子发生了疫情，见图 1-1。我们要用汽车快速地将疫苗送达到 5 个村子，目标是寻找一条耗时最少的路线。

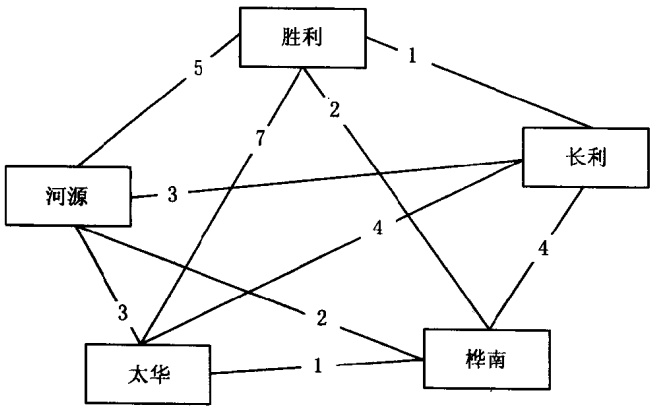


图 1-1 发生疫情的五个村子

1. 问题分析

用结点表示村子，结点之间的连线表示村子之间的汽车路线。两个结点之间的值称为权，表示两个村子之间汽车的耗时。这里假定两个村子之间的两个方向的耗时是相同的。这样，可以用图结构来表示这个问题，见图 1-2。图既是数学分支“图论”研究的对象，也是一种典型的数据结构。利用图结构表示实际问题，也就是用图结构对问题进行抽象。这种抽象将一个具体的问题转化为一类抽象的、一般的问题，从而使我们对问题的本质更清楚。对于这类问题，可以这样叙述：对于有 n 个结点的图，要访问图中的每一个结点，并仅访问一次，要求访问的代价最小。这个代价可以是时间、距离、耗油等等。

2. 算法

假设图有 n 个结点，将这 n 个结点放在一个数组里，称为结点数组。用结点之间的权创建一个矩阵，称为“耗时矩阵”。将这个矩阵放在一个二维数组中，见图 1-3。

首先考虑穷举法。所谓穷举法，是指将所有的可能情况列出来，求出每种情况的耗时，然后进行比较，找出代价最小的解。穷举法可以准确地求出代价最小的解，但是对结点少的问题，也就是规模小的问题可以用，而对于规模大的问题，其求解时间会随着问题规模的增

长而成指数增长，计算机将无法承受。下面采用贪心算法来解决这个问题。

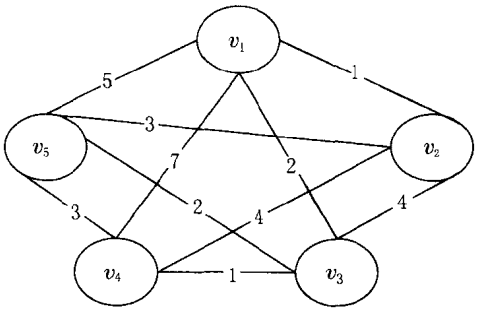


图 1-2　村子联系网络

$$\begin{bmatrix} 0 & 1 & 2 & 7 & 5 \\ 1 & 0 & 4 & 4 & 3 \\ 2 & 4 & 0 & 1 & 2 \\ 7 & 4 & 1 & 0 & 3 \\ 5 & 3 & 2 & 3 & 0 \end{bmatrix}$$

图 1-3　耗时矩阵

贪心算法是从问题的某一个初始解出发逐步逼近给定的目标，以尽可能快地求得更好的解。当达到算法中的某一步不能继续前进时，算法停止。这时就得到了问题的一个解，但是这个解不能保证是最优的。正因为这个缺点，所以限制了贪心法的应用范围。

用贪心法解决问题时，往往有 n 个输入，最后求得满足某些约束条件的解的子集。而任一个满足这些约束条件的解，我们称它是可行的。

用 $c(n, n)$ 二维数组来存放 n 个村子的耗时矩阵，用 *cost* 表示耗时量，用 *tour* 表示汽车行驶的途径，初始时 *tour* 为空（null），而 $<v, s>$ 表示从村子 v 到达任意一个没有经过的村子 s 最少耗时的边，$c(v, s)$ 则表示其耗时量。假定出发村子为 v_0，耗时矩阵要事先输入。

[算法 1.1]

```
Greedy(v0)
{//贪心算法
  tour=null;
  cost=0;
  v=v0;
  for(k=1,k<=n-1,k++)
   {tour=tour+<v,s>;
    cost=cost+c(v,s);
    v=s;
   }
 tour=tour+<v,v0>;
 cost=cost+c(v,v0);
}
```

对于快速送达疫苗的例子，若选择村子 1 作为出发点，执行本算法后，汽车行驶的路线为 1-2-5-3-4-1，其最少耗时为 14。然而它不是耗时最少的路线，例如，路线 1-5-3-4-2-1 的耗时量为 13。

Greedy 算法设计简单，运算快且直观。若有 n 个结点，仅生成耗时矩阵 c 就需要 n^2 次操作，因此解决这个问题的任何算法所需时间的下界是 n^2。

3. 解的精化

Greedy 算法的缺点在于，从某一固定村子出发，找到一条当前耗时最少的路线后，不可能再找到比这条路线耗时更少的路径，然而实际上存在着耗时更少的路线。可以对上述贪心算法进行适当的改进，以求得更满意的结果。一种方法是选择从 $p \leqslant n$ 个不同村子出发，执

行 Greedy 算法，然后从中选出耗时最少的路线。Greedy1 为修改后的算法：

[算法 1.2]

```
Greedy1
{//修改后的贪心算法
  cost1 = ∞;
  Best = null;
  for (k = 1;k < = p;k + +)
  {v0 = k;
    Greedy(v0);
    if(cost < cost1)
      {Best = tour;
        cost1 = cost;
      }
    }
}
```

1.2 数据结构

1.2.1 为什么要学习数据结构

计算机早期处理的问题主要是数值问题。数值问题所涉及的运算对象是简单的整型、实型或布尔类型数据。随着计算机的广泛应用，计算机处理的主要对象已由数值问题转化为非数值问题。这类问题一般无法用数学方程式加以描述，而需要考虑它们所涉及到的复杂的数据结构，需要考虑数据元素之间的相互关系。下面所列举的就是属于这一类的具体问题。

例 1.2 考生录取信息系统。

若某高校需要在报考该学校的考生中查找某个考生的有关情况，或者想查询报考某个专业的考生的有关情况，或者想统计某个分数段的考生的情况，则可以建立相关的数据结构，并且按照某种算法编写相关程序，从而实现计算机自动检索。由此，可以在考生录取信息系统中建立一张按考号顺序排列的考生信息表和分别按姓名、专业、成绩顺序排列的索引表，如图 1-4 所示。由这四张表构成的文件便是学生信息检索的数学模型，计算机的主要操作便是按照某个特定要求（如给定姓名）对考生信息文件进行查询。

诸如此类的表结构，还有电话自动查号系统、图书馆的书目检索系统和仓库库存管理系统等。在这类文档管理的数学模型中，计算机处理的对象之间通常存在着一种简单的线性关系，因此这类数学模型称为线性的数据结构。

例 1.3 八皇后问题。

为简便起见，这里只考虑四个皇后。在四皇后问题中，处理过程不是根据某种确定的计算法则，而是利用试探和回溯的探索技术求解。为了求得合理布局，在计算机中要存储布局的当前状态。而布局之间的关系是由比赛规则决定的。通常这个关系不是线性的，因为可以从一个布局演变成多个布局。从最初的布局状态开始，一步一步地进行试探，每试探一步形成一个新的状态，整个试探过程形成了一棵隐含的状态树。如图 1-5 所示。回溯法求解过程实质上就是一个遍历状态树的过程。这个问题中所出现的数据结构是树结构。树结构可以应用在许多非数值计算的问题中。

考号	姓名	性别	报考专业	成绩
2300411	李闽志	男	计算机科学与技术	658
1000472	于惠芳	女	英语	632
1506302	刘　红	女	应用数学	617
2105902	宋大明	男	英语	600
0934785	高大庆	男	计算机科学与技术	601
0600807	何文丽	女	英语	611
0878529	隋文涛	男	应用数学	612
1690834	崔秀海	男	英语	602
1710641	于众群	女	计算机科学与技术	619
1900162	魏人民	男	英语	671

a）考生信息表

崔秀海	8
高大庆	5
何文丽	6
李闽志	1
刘　红	3
宋大明	4
隋文涛	7
魏人民	10
于惠芳	2
于众群	9

b）姓名索引表

计算机科学与技术	1，5，9
英语	2，4，6，8，10
应用数学	3，7

c）专业索引表

600～609	4，5，8
610～619	3，6，7，9
620～629	
630以上	1，2，10

d）成绩索引表

图 1-4　考生信息系统中的数据结构

对于例 1.1 的送疫苗问题，不难看出它是一个图形结构的问题。

由以上三个例子可见，描述这类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。因此，可以说数据结构课程主要研究非数值计算的程序设计问题中所出现的计算机操作对象以及它们之间的关系和操作。

学习数据结构的目的之一是，将实际问题中所涉及的处理对象在计算机中表示出来并对它们进行处理。

1.2.2　有关概念和术语

在系统地学习数据结构知识之前，下面先对一些基本概念和术语赋予确切的含义。

数据（Data）是信息的载体，是描述客观事物的数、字符以及所有能输入到计算机中并被计算机程序识别和处理的符号的集合。现实世界信息的分析、复制、传播首先要符号化，以便于计算机的处理。数据分为两类：数值型数据（主要用于数学计算等）和非数值型数据（文字、图形、图像、音频、视频等）。

数据元素（Data Element）是数据的基本单位。在不同的条件下，数据元素又可称为元素、结点、顶点、记录等。例如，考生信息系统中考生信息表中的一个记录、八皇后问题中状态树的一个状态、送疫苗问题中的一个顶点等，都被称为一个数据元素。

一个数据元素可由不可分割的若干个**数据项**（Data Item）组成，例如，考生信息系统中考生信息表的每一个数据元素就是一个考生记录，它包括考生的考号、姓名、性别、成绩等数据项。数据项是数据的不可分割的最小单位。

数据对象（Data Object）是性质相同的数据元素的集合，是数据的一个子集。在具体问题中，数据元素都具有相同的性质（元素值不一定相等），例如，英文字符集是数据的一个子集，它由能够表示单词这样的共同属性的 26 个字母组成一个数据对象。数据元素是数据

对象的一个实例。

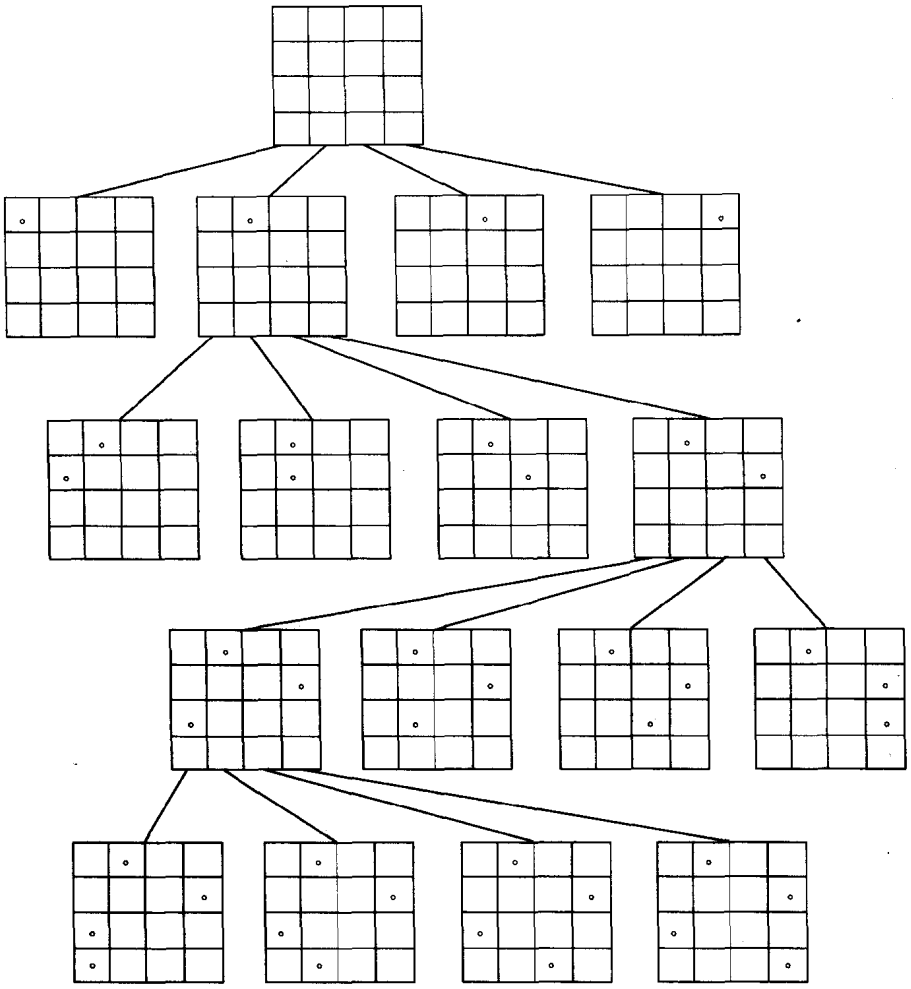


图 1-5 四皇后问题中隐含的状态树

数据结构（Data Structure）是指相互之间存在着一种或多种关系的数据元素的集合。在任何问题中，数据元素之间总是存在联系的。把某一数据对象及该数据对象中所有数据成员之间的关系组成的实体叫做数据结构。根据数据元素间关系的不同特性，通常有下列四类基本的结构：

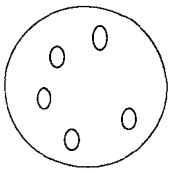
1）集合结构。在集合结构中，数据元素间的关系是“属于同一个集合”。集合是元素关系极为松散的一种结构。如蓝色、红色和黄色同属色彩集合。

2）线性结构。该结构的数据元素之间存在着一对一的关系。如考生信息表中的各元素。

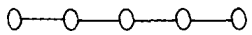
3）树形结构。该结构的数据元素之间存在着一对多的关系。如家谱，一个父亲对应着多个儿子。

4）图形结构。该结构的数据元素之间存在着多对多的关系。图形结构也称为网状结构，如城市交通图。

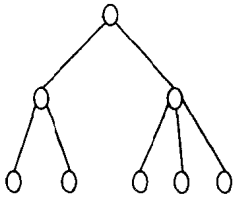
图 1-6 为表示上述四类基本结构的示意图。



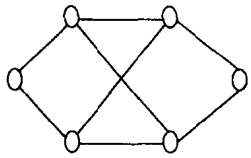
a）集合结构



b）线性结构



c）树形结构



d）图形结构

图 1-6　四类基本结构的示意图

由于集合是数据元素之间关系极为松散的一种结构，因此也可用其他结构来表示它。

1. 数据结构的定义

从上面所介绍的数据结构的概念中可以知道，一个数据结构有三个要素。一个是数据元素的集合，二是关系的集合，三是进行的运算。在形式上，数据结构通常可以采用一个二元组来表示。

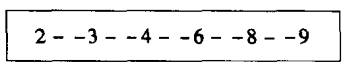
数据结构的形式定义为：

$$Data\_Structure = (D, R)$$

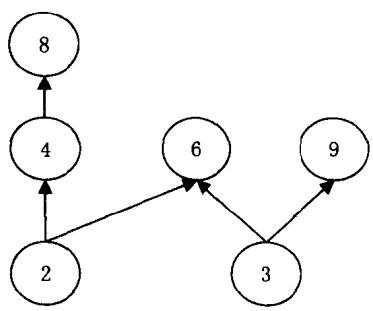
其中，D 是数据元素的有限集，R 是 D 上关系的有限集。

例 1.4　集合 $D = \{2, 3, 4, 6, 8, 9\}$ 上的数据结构。

设 $R1$ 为 D 上的“<”关系，$R2$ 为 D 上的整除关系，则 $R2 = \{(2,2),(2,4),(2,6),(2,8),(3,3),(3,6),(3,9),(4,4),(4,8),(6,6),(8,8),(9,9)\}$。由 D 上的两个关系 $R1$ 和 $R2$，可得到两种不同的数据结构，如图 1-7 所示。



a）数据结构（D, $R1$）



b）数据结构（D, $R2$）

图 1-7　数据结构

通过这个例子可以看出，所谓数据结构就是带有关系的集合，即使是由相同数据元素构成的集合，若其上的关系不同，也不是同一个数据结构。由此可见，数据结构不仅描述了在这个结构中有哪些数据元素，而且

还刻画了这些元素之间的关系。

计算机科学家把常用的数据结构按照它们的表示方法进行分类，并对各种数据结构的行为特性做了研究，总结出许多典型的数据结构。例如，顺序表、链表、堆栈、队列、树、二叉树、图、集合等。

2. 数据结构的逻辑结构与物理结构

“数据结构”定义中的“关系”是指数据之间的逻辑关系，故也称为数据结构的逻辑结构。数据结构的逻辑结构仅仅刻画了数据元素之间的逻辑关系，但是我们并不清楚这些结构在计算机内部是如何实现的，为此需要考虑数据结构在计算机的内存中是如何存放的。

数据结构在计算机中的表示称为物理结构，又称存储结构。它所研究的是数据结构在计算机中的实现方法，包括数据结构中元素的表示及元素间关系的表示。一种逻辑结构通常可以采用多种存储表示。通常，数据结构的存储结构采用顺序存储或链式存储的方法。

顺序存储方法是把逻辑上相邻的元素存储在物理位置相邻的存储单元中，由此得到的存储表示称为顺序存储结构。顺序存储结构是一种最基本的存储表示方法，通常借助于程序设计语言中的数组来实现。

链式存储方法对逻辑上相邻的元素不要求其物理位置相邻，元素间的逻辑关系通过附设的指针字段来表示，由此得到的存储表示称为链式存储结构。链式存储结构通常借助于程序设计语言中的指针类型实现。

除了通常采用的顺序存储方法和链式存储方法外，有时为了查找的方便还采用索引存储方法和散列存储方法。

数据结构的描述是与具体语言无关的，同一种数据结构可以用多种语言来描述。

例 1.5 例 1.2 中的考生信息表数据用 C 语言的结构体数组 *examinee*[100]来存储。

```
struct student
{int no;
 char name[8];
 char gender[2];
 char specialty[20];
 float grade;
}examinee[100]
```

3. 算法与数据结构课程的内容

算法与数据结构这门课程研究的是计算机科学中的许多最基本的问题，是研究问题求解技术、算法设计、程序设计的基础。在国外，从 1968 年才开始把数据结构设立为一门独立的课程。数据结构和算法方面最早的权威性著作是 Donald E. Knuth 所著的系列丛书 *The Art of Computer Programming*（计算机程序设计技巧），它形成了数据结构的最初体系。第 1 卷《基本算法》是第一本较系统地阐述数据的逻辑结构和存储结构及其操作的著作。从 20 世纪 60 年代末到 70 年代初，出现了大型程序，软件也相对独立，结构程序设计成为程序设计方法学的主要内容，人们越来越重视数据结构。从 20 世纪 70 年代中期到 80 年代，各种数据结构著作相继出现。目前，数据结构的发展并未终结，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，从抽象数据类型和面向对象的观点来讨论数据结构已成为一种新的趋势，越来越为人们所重视。

数据结构与数学、计算机硬件和软件有十分密切的关系。数据结构是计算机科学与技术

专业的核心课程，是编译原理、操作系统、数据库、人工智能等课程的基础。同时，在信息科学、系统工程、应用数学等工程技术领域中也广泛地使用数据结构技术。

数据结构课程主要讨论为程序设计构造好的数据结构，并实现数据结构，而且还考虑数据结构及其实现的评价与选择。数据结构的内容包括三个层次：抽象、实现和评价。

数据结构的核心技术是分解与抽象。通过分解可以划分出数据的三个层次；再通过抽象，舍弃数据元素的具体内容，就得到逻辑结构。类似地，通过分解将处理要求划分成各种功能，再通过抽象舍弃实现细节，就得到运算的定义。上述两个方面的结合可以将问题变换为数据结构。这是一个从具体（即具体问题）到抽象（即数据结构）的过程。然后，通过增加对实现细节的考虑进一步得到存储结构和实现运算，从而完成设计任务。这是一个从抽象（即数据结构）到具体（即具体实现）的过程。熟练地掌握这两个过程是数据结构课程在专业技能培养方面的基本目标。

对于每一种数据结构，都要考虑它的代价和效益。对于要解决的问题，都有时间和空间上的限制。在选择数据结构时，需要考虑存储空间和操作时间的限制，要选择适合问题的最好的数据结构。

例 1.6　为银行设计一个支持 ATM（自动取款机）的数据库系统。

［要求］该数据库系统可以添加和删除顾客的账户，顾客可以存款、取款和查询账户信息。银行还提供 ATM 让顾客自己进行存款、取款和查询。

［分析］在所要设计的数据库系统中，数据结构应该满足下列要求：因为经常存款、取款和查询，所以要求数据结构的查找效率高；对于顾客来说，销户是没有时间限制的，所以要求数据结构的删除效率低；因为顾客并不经常开新账户，开户的时间略长一点对于用户来说是可以接受的，所以要求数据结构的插入效率适度。这样，可以知道对于这个问题，所需要的是一个删除效率低、查找效率高、插入效率适度的数据结构。后面将介绍的散列表就是符合这些要求的。散列表不仅检索速度非常快，而且当不改变记录的长度时，修改操作非常容易。同时，散列表的插入效率也比较高。

［结论］选择散列表作为银行的数据库系统。

1.3　抽象数据类型

1. 抽象数据类型的概念

抽象数据类型（Abstract Data Type，ADT）可以使我们更容易描述现实世界。例如：用线性表描述学生成绩表，用树或图描述遗传关系。抽象数据类型是指一个数学模型以及定义在该模型上的一组操作。抽象数据类型使得使用它的人可以只关心它的逻辑特征，不需要了解它的存储方式。定义它的人同样不必要关心它如何存储。

抽象数据类型可分为原子类型和聚合类型。原子类型是其值不可分解的抽象数据类型，如 int 类型。聚合类型还可分成固定聚合类型和可变聚合类型。其中，固定聚合类型的值由确定数目的成分按某种结构组成，如复数；而可变聚合类型的值的成分数目不确定，如学生基本情况。

抽象数据类型的特征是使用与实现相分离，实行封装和信息隐蔽。也就是说，在抽象数据类型设计时，把类型的定义与其实现分离开来。