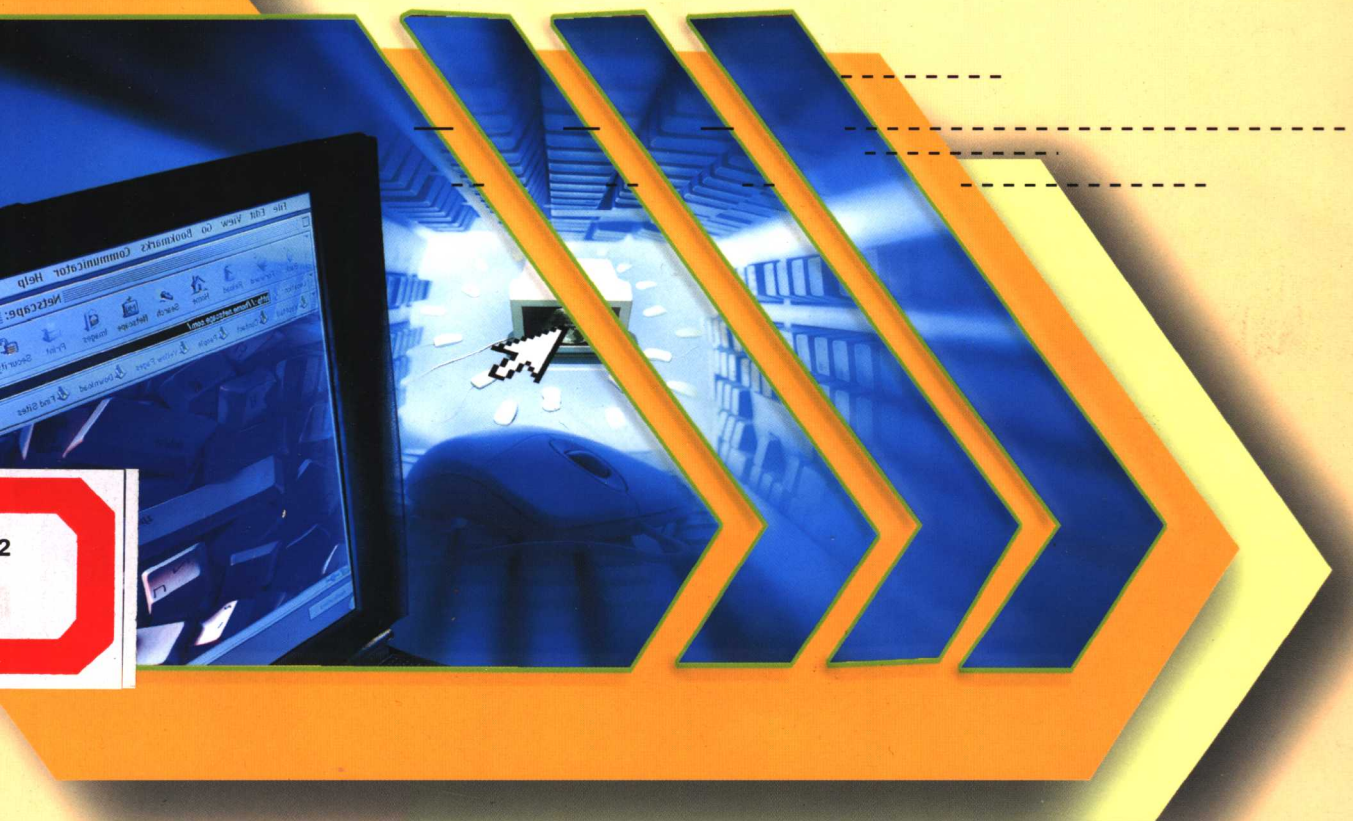
高职高专计算机系列教材

主编 谭浩强

数据结构实用教程

陈 明 编著



清华大学出版社

高职高专计算机系列教材

主编 谭浩强

数据结构实用教程

陈 明 编著

清华大学出版社

北京

内容简介

本书系统地介绍了各种典型的数据结构，主要包括线性表、栈和队列、串、数组和广义表、树、图、查找、排序等，为了加强对算法的理解，还介绍了算法分析方面的内容。本书选材精炼，概念清楚，注重实用，逻辑性强。书中所涉及的数据结构与算法都给出了C语言描述。本书附有大量的习题，便于学生理解与掌握。

本书可作为高职高专院校计算机专业及相关专业的教材，也可作为计算机应用技术人员的参考书。

图书在版编目(CIP)数据

数据结构实用教程/陈明编著. —北京：清华大学出版社，2004.11

(高职高专计算机系列教材)

ISBN 7-302-09599-X

Ⅰ.数…　Ⅱ.陈…　Ⅲ.数据结构—高等学校：技术学校—教材　Ⅳ.TP311.12

中国版本图书馆CIP数据核字(2004)第096[illegible]8号

出 版 者：清华大学出版社　　地　　址：北京清华大学学研大厦

http://www.tup.com.cn　　邮　　编：100084

社 总 机：010-62770175　　客户服务　010 62776969

责任编辑：谢　琛

印 装 者：北京国马印刷厂

发 行 者：新华书店总店北京发行所

开　　本：185×260　印张：13.25　字数：302千字

版　　次：2004年11月第1版　2004年11月第1次印刷

书　　号：ISBN 7-302-09599-X/TP·6659

印　　数：1～5000

定　　价：18.00元

编辑委员会

《高职高专计算机系列教材》

目录

第1章

绪　论

在深入学习数据结构(data structure)之前，首先了解学习数据结构的意义、数据结构的基本术语及数据结构的一些相关概念等。这对于深刻理解书中后面章节的内容将有很大的帮助。

1.1　数据结构的重要性

在计算机发展的初期，人们使用计算机主要是处理数值的计算问题，程序设计人员也主要把精力集中在程序设计的技巧上。但随着计算机应用领域的扩大和软硬件的发展，计算机对信息的加工处理已从单一的数值计算，发展到大量地解决非数值问题。其加工处理的信息也由简单的数值发展到字符、图像、声音等具有复杂结构的数据。数据结构这门学科随着计算机数据的复杂化而产生并发展起来了。

非数值问题的数据之间的相互关系一般无法完全用数学方程式加以描述，并且数据的表示方法和组织形式直接关系到程序对数据的处理效率，而系统程序和许多应用程序的规模很大，结构复杂，这时人们考虑问题的关键已不再是分析数学和计算方法，而是放在是否能设计出合适的数据结构，有效地解决问题上。

总之，计算机科学是一门研究用计算机进行信息表示和处理的科学。这里面涉及到两个问题：信息的表示和信息的处理，其中，信息的表示和组成又直接关系到处理信息程序的效率。随着计算机的普及，信息量的增加，信息范围的拓宽，使许多系统程序和应用程序的规模很大，结构又相当复杂，这就需要人们对计算机程序加工的对象进行系统地研究，即研究数据的特性及数据之间存在的关系，而数据结构正是描述数据的特性及数据之间存在的关系的一门课程。

数据结构是计算机专业的核心课程之一，在众多的计算机系统软件和应用软件中都要用到各种数据结构。可以这样说，数据结构不仅是一般程序设计的基础，而且是实现编译程序、操作系统、数据库系统及其他系统程序和大型应用程序的基础。因此，仅掌握几种计算机语言是难以应付众多复杂的研究课题的，要想有效地使用计算机，还必须学习数据结构的知识。

瑞士计算机科学家 N. Wirth 教授曾提出这样一个等式：算法＋数据结构＝程序，这

个等式形象地描述了算法、数据结构和程序之间的关系，这里的数据结构指的是数据的逻辑结构和存储结构，而算法就是对数据运算的描述。由此可见，程序设计的实质就是对实际问题选取一种优秀的数据结构，加之设计一个优秀的算法，而且优秀的算法很大程度上取决于描述实际问题的数据结构。

1.2 基本术语

为了更好地理解数据结构这个概念，首先解释数据结构中的一些常用名词和术语。

数据(data) 是信息的载体，它是描述客观事物的数、字符及所有能输入到计算机中被计算机程序识别、加工处理的信息的集合。数据不只是通常意义下的整数和实数等，随着计算机的广泛应用，数据的范畴也随之拓广，计算机可以处理的字符串、图像、声音等都可以被称为数据。所以不能只是泛泛地理解数据这个概念。下面进一步解释数据的定义。如表 1-1 所示，例如张风的英语成绩为 92 分，92 就是该学生的成绩数据。

表 1-1 学生成绩表

学 号	姓 名	语 文	数 学	英 语
S01012	张风	85	69	92
S01022	李强	87	73	74
S02013	王海	92	64	84

数据元素(data element) 是数据的基本单位，是对一个客观实体的数据描述。一个数据元素可以由一个或若干个数据项组成。数据元素也被称为结点或记录。

数据项(data item) 是数据的具有独立意义的不可分的最小单位，它是对数据的数据元素属性的描述。数据项也被称为字段或域。

利用上面表 1-1 的例子，来说明一下数据项和数据元素，整个表记录的是学生的成绩数据，而每个学生的一条记录(包括学号、姓名、语文(分)、数学(分)、英语(分))就是其中的一个数据元素，见图 1-1。

图 1-1 数据元素和数据项

数据对象(data object) 是具有相同性质的数据元素的集合，它是数据的一个子集。如上例所示，一个班级的成绩表可以看做一个数据对象。例如，集合{1,2,3,4,5,…}是自然数的数据对象，而集合{'a','b','c','d',…,'z'}是英文字母表的数据对象。可以看出，数据对象可以是无限的，也可以是有限的。

数据类型(data type) 是具有相同性质的计算机数据的集合及定义在这个数据集

合上的一组操作的总称。例如：在C语言中的整数类型是集合C＝{0，±1，±2，±3，±4，…}及定义在该集合上的加、减、乘、整除和取余等一组操作。数据类型封装了数据存储与操作的具体细节。

每个数据项属于某个确定的基本数据类型，数据类型可分为两类：原子类型和结构类型。

原子类型：如果一个数据元素由一个数据项组成，这个数据元素的类型就是这个数据项的数据类型，其值在逻辑上不可分解(如int i，float j)。

结构类型：如果一个数据元素由多个不同的类型的数据项组成，则这个数据元素的类型就是由各数据项类型构造而成的结构类型。在上面的学生成绩表中，数据项姓名的数据类型为字符型，而成绩的数据类型是数值型的，所以这个数据元素是一个结构类型。上述成绩表数据用C语言的结构体数组Classlstu[50]来存储：

```
Struct Stu
{/*数据项*/
  int stuID;
  char name[20];
  int maths_score;
  int chinese_score;
  int english_score;
} Classlstu[50];
```

不同的高级语言提供的基本数据类型有所不同，在C语言中，提供了实型、整型、字符型和指针型等基本数据类型。

抽象数据类型(abstract data type) 简写为ADT，是用户在数据类型基础上新定义的数据类型。抽象数据类型定义包括数据组成和对数据的处理操作。可见抽象数据类型是数据和数据使用者的一个接口。

抽象数据类型的定义仅取决于它的一组逻辑特性，而与它在计算机中如何表示和实现无关，只要其数学特性不变，都不影响其外部的使用。从抽象数据类型的角度入手设计软件系统，可大大提高软件构件的复用率。抽象数据类型的定义包括数据对象定义、数据关系定义和基本操作定义3部分。其书写格式如下：

ADT：抽象数据类型名

数据对象：数据对象的定义。

数据关系：数据关系的定义。

基本操作：基本操作的定义。

数据结构(data structure) 就是数据之间的相互关系(即数据的组织形式)及在这些数据上定义的数据运算方法的集合。

1.3 数据结构的概念

简单地说，数据结构指的是数据之间的相互关系，即数据的组织形式，一般包括以下3个方面的内容：

(1) 数据之间的逻辑关系，也称为数据的逻辑结构。

(2) 数据元素及其关系在计算机存储器内的表示，称为数据的存储结构，也就是物理结构。

(3) 数据的运算，即对数据进行的操作。

为了进一步理解数据结构，举一个简单的例子来说明。

例如现在有一个学生的基本情况表，排列顺序没有任何规律，如表 1-2 所示，表中记录了某校全体学生的姓名和相应的基本信息。现在要求设计一个算法，当给定任何一个学生的姓名时，计算机能够查出该学生的基本信息，如果根本就不存在这个学生，计算机就输出“无此学生记录!”。

表 1-2　学生基本情况表

编号	姓名	年级	年龄	性别
01	张风	1 年级	12	男
02	李强	1 年级	14	男
03	林海	2 年级	14	男
04	李南	2 年级	10	男
05	韩凤	3 年级	11	女
06	赵加	1 年级	12	女
…	…	…	…	…

这个例子实现的是查找的功能。可以看出，这个算法的设计完全依赖于基本情况表中学生姓名和相应信息在计算机内的存储方式。

如果学生基本情况表中的学生的姓名是随意排列的，那么，在给定一个学生姓名时，则只能对学生的基本情况表从头到尾逐个与给定的姓名比较，顺序查找，直至找到所给的姓名为止，很有可能查找完全部基本情况表还没有找到这个人。虽然这种方法很简单，是线性查询，但是这样会浪费很多时间，效率低。

现在如果将基本情况表进行适当地组织，按字母顺序排列学生的姓名和相应的情况，如表 1-3 所示，并且再构造一个字母索引表如表 1-4 所示，这个表用来登记以某个字母开头的第一个学生姓名在基本情况表中的起始位置。当查找某学生的情况时，先从索引表中查到以该字母开头的第一个学生姓名在基本情况表中的起始位置，然后，从此起始处开始查找，而不必去查看以其他字母开头的学生记录，通过建立这样一种数据组织形式，查找的效率就会有很大的提高。另外，还可以按年级进行排序，然后建立一个年级的索引表，当查询某个年级的学生时，可以先找到这个年级所在的开始位置，然后再查询，这样就大大提高了查找速度。

对于不同的存储结构，就要构造出完全不同的算法。算法和数据结构是密切相关的，算法依赖于具体的数据结构，数据结构直接关系到算法的选择和效率。

此外，当新生入校时，学生基本情况表就需要添加新生的姓名和相关的信息，在学生毕业或转学时，应从基本信息表中删除他的记录。这就要求在已安排好的结构上进行插入和删除操作。除此之外，还可能对学生基本情况表进行修改等运算。对于这些运算，由计算机来完成，就需要设计相应的算法，也就是说，数据结构还需要给出每一种结构中的

表 1-3　按开头字母排序的基本情况表

编号	姓名	年级	年龄	性别
⋮	⋮	⋮	⋮	⋮
11	韩风	3 年级	11	女
⋮	⋮	⋮	⋮	⋮
24	李强	1 年级	14	男
25	李南	2 年级	10	男
26	林海	2 年级	14	男
⋮	⋮	⋮	⋮	⋮
87	张风	1 年级	12	男
88	赵加	1 年级	12	女
⋮	⋮	⋮	⋮	⋮

表 1-4　字母索引表

开头字母	编号
⋮	⋮
H	11
⋮	⋮
L	24
⋮	⋮
Z	87
⋮	⋮

各种运算和算法。

通过上面的讨论，可以直观认为：数据结构是研究数据元素之间的相互关系和这种关系在计算机中的存储表示，并对这种结构定义相应的运算，设计出相应的算法，而且确保经过这些运算后所得到的结果仍然是原来的结构类型。

数据结构讨论的问题主要有以下几个方面：如何以最节省存储空间的方式来表示数据和各种不同的数据结构表示方法及其相关算法；如何有效地改进算法效率使程序的执行速度更快；数据处理的各种技巧，如排序、查找等算法的介绍等。

数据结构包括逻辑结构和存储结构(物理结构)。逻辑结构是在逻辑关系上描述数据的，它与数据在计算机内的存储方式没有关系，可看做是从具体问题中抽象出来的数据模型。其关系表示如表 1-5 所示。

表 1-5　逻辑结构和存储结构

逻辑结构		数据结构定义中的“关系”指数据间的逻辑关系，故也称数据结构为逻辑结构
存储结构	顺序存储结构	数据结构在计算机内的表示称为物理结构；又称存储结构
	链式存储结构	

1.4　数据的逻辑结构

数据结构在形式上可定义为一个二元组：

Data_Structure=(D,R)

其中，D 是数据元素的有限集合，R 是 D 上关系的有限集合。

由上可以看出，数据结构是由两部分构成：

(1) 数据元素的集合 D。

(2) 数据元素之间关系的集合。

现在需要设计一个数据结构，要求每个课题组由一位教授、1～4 名研究生和 1～8 名

本科生组成；在小组中，一位教授指导 1～4 名研究生，每位研究生指导 1～2 名本科生，得到一个数据结构：

Group=(P,R)

其中，P 表示数据元素，包括教授、研究生、本科生，即 $P=\{T,G_1,\cdots,G_n,S_{11},S_{12},\cdots,S_{nm}\}$，$1\leqslant n\leqslant 4$，$1\leqslant m\leqslant 2$。

R 表示小组成员的关系，他们的关系有两种，教授和研究生：$R1=\{\langle T,G_i\rangle \mid 1\leqslant i\leqslant n, 1\leqslant n\leqslant 4\}$；研究生和本科生：$R2=\{\langle G_i,S_{ij}\rangle \mid 1\leqslant i\leqslant n, 1\leqslant j\leqslant m, 1\leqslant n\leqslant 4, 1\leqslant m\leqslant 2\}$。

再举一个例子，一星期的数据结构可表示为：

Group=(D,S)

D={星期一，星期二，星期三，星期四，星期五，星期六，星期日}

S={〈星期日，星期一〉，〈星期一，星期二〉，〈星期二，星期三〉，〈星期三，星期四〉，〈星期四，星期五〉，〈星期五，星期六〉，〈星期六，星期日〉}

以上数据结构用图表示为如图 1-2 所示的关系。

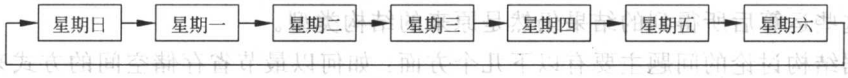


图 1-2　一星期数据结构图示

总而言之，数据结构是相互之间存在一种或多种特定关系的数据元素的集合，这个关系描述的是数据元素之间的逻辑关系。数据的逻辑关系也称为数据的逻辑结构，它与数据的存储无关，独立于计算机，因此，数据的逻辑结构可以看成是从具体的问题中抽象出来的数学模型。

数据的逻辑结构分为 3 种典型结构，第 1 种是集合，第 2 种是线性结构，第 3 种是非线性结构。

(1) 集合是元素间为松散的关系，只是同属于一个集合而已，如各种颜色属于色彩集合。

(2) 线性结构的逻辑特征是有且仅有一个起始结点和一个终端结点，并且所有结点只有一个直接前趋和一个直接后继，如线性表、队列等，结点之间是一对一的关系，如前面的学生成绩表中每个数据元素之间的关系。

(3) 非线性结构的特征是一个结点可能有多个直接前趋或多个直接后继，如树、图等，树的结点之间是一对多的关系，图的结点之间的关系是多对多的关系(这些将在后面的章节中讲述)。

1.5　数据的存储结构

数据的存储结构是数据的逻辑结构在计算机内部的表示或实现，又称为数据的物理结构，它包括数据元素的表示和关系的表示，它与计算机语言无关。

通常，在计算机内数据元素用一组连续的位串来表示，称这个位串为结点。数据元素

之间的关系，即结点之间的关系，在计算机内有 4 种基本的存储表示方法。

1. 顺序存储方法

该方法是将逻辑上相邻的结点存储在物理位置上也相邻的存储单元里，结点之间的逻辑关系由存储单元的邻接关系来表示(也就是说，只存储结点的值，不存储结点之间的关系)，这种存储表示称为顺序存储结构。它主要应用于线性的数据结构，非线性的数据结构也可以通过某种线性化的过程后，进行顺序存储。

顺序存储结构的主要特点如下：

(1) 结点中只有自身信息域，没有连接信息域。因此存储密度大，存储空间利用率高。

(2) 可以通过计算直接确定数据结构中的第 I 个结点的存储地址 L_i，计算公式为

$$L_i = L_1 + (I-1) * m$$

其中，L_1 为第一个结点的存储位置，m 为每个结点所占用的存储单元个数。

(3) 插入和删除都会引起大量的结点移动。

【例 1-1】 有一个数据结构如下

$A=(D,S)$

$D=\{a,b,c,d,e\}$

$S=\{\langle a,b\rangle,\langle b,c\rangle,\langle c,d\rangle,\langle d,e\rangle\}$

设第一个结点的存储单元位置为 1000，每一个结点所占的存储单元的个数为 1。

存储结构如图 1-3 所示。

存储单元	a	b	c	d	e
地址	1000	1001	1002	1003	1004

图 1-3 顺序存储结构

如果要计算第 3 个结点(即“C”)的存储地址 L_3，则：

$$L_3 = L_1 + (I-1) * m = 1000 + (3-1) * 1 = 1002$$

2. 链式存储方法

链式存储方法不要求逻辑上相邻的结点在物理位置上也相邻，结点间的关系由附加的指针来表示，指针指向结点的邻接结点，这样将所有结点串联在一起，称为链式存储结构。也就是说，链式存储方法不仅存储结点的值，而且还存储结点之间的关系。

所以，链式存储方法中的结点由两部分组成，一个是存储结点本身的值，称为数据域；另一个是存储该结点的各后继结点的存储单元地址，称为指针域(可包含一个或多个指针)。

链式存储结构的主要特点如下：

(1) 结点中除自身信息外，还有表示连接信息的指针域，因此比顺序存储结构的存储密度小，存储空间利用率低。

(2) 逻辑上相邻的结点，物理上不必邻接，可用于线性表、树、图等多种逻辑结构的存

储表示。

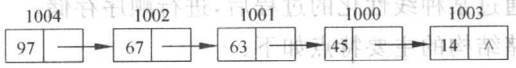
(3) 删除和插入操作灵活方便，不必移动结点，只要改变结点中的指针值即可。

这种存储结构将在后面的章节中详细讲述，在这里只举一个简单例子说明一下。

【例 1-2】 假设存在这样一个线性结构的结点集合 D={45,63,67,14,97}，以结点值的降序为关系 S={⟨97,67⟩,⟨67,63⟩,⟨63,45⟩,⟨45,14⟩}，链接存储结构如图 1-4 所示。

地址	数据	指针
1000	45	1003
1001	63	1000
1002	67	1001
1003	14	^
1004	97	1002

(a) 存储结构



(b) 逻辑结构

图 1-4 线性结构的链接存储

3. 索引存储方法

索引存储是在存储结点信息的同时，再建立一个附加的索引表，然后利用索引表中索引项的值来确定结点的实际存储单元地址。索引表中的每一项称为索引项，索引项的一般形式为(关键字，地址)，关键字能惟一标识一个结点。

4. 散列存储方法

散列存储方法的基本思想是根据结点的关键字直接计算出结点的存储地址。

把结点的关键字作为自变量，通过一个称为散列函数 H(a.key)的计算规则，确定出该结点的确定存储单元地址。

上面这 4 种方法既可以单独使用，也可以组合起来对数据结构进行存储。同一种逻辑结构采用不同的存储方法，可以得到不同的存储结构。选取哪种存储结构来表示相应的逻辑结构，视具体的情况而定，主要考虑的是数据的运算是否方便及相应算法的时间复杂度和空间复杂度的要求。

1.6 数据的运算

为了有效地处理数据，提高数据的运算效率，可以按一定的逻辑结构把数据组织起来，并选择适当的存储方法，把数据存储到计算机内，然后对其进行运算。

数据的运算是定义在数据的逻辑结构之上的，每一种逻辑结构都有一个运算的集合，例如插入、删除、修改等。这些运算实际上是在数据元素上施加的一系列的抽象的操作。所谓抽象的操作，是指我们只知道这些操作要求我们“做什么”，而无须去考虑“如何做”，只有在确定了存储结构之后，再考虑如何具体实现这些运算。下面简单地介绍几种数据的运算。