普通高校本科计算机专业特色教材精选·算法与程序设计

数据结构（C语言版）（第2版）

秦玉平　马靖善　主编

清华大学出版社

普通高校本科计算机专业特色教材精选·算法与程序设计

数据结构（C语言版）（第2版）

秦玉平　马靖善　主编

清华大学出版社
北京

内 容 简 介

数据结构是高等学校计算机及其相关专业的核心课程，是计算机程序设计的基础，也是程序员考试和硕士研究生入学考试的必考科目。

本书共分 11 章，第 1 章是数据结构的概述；后 10 章分别讨论了顺序表、链表、栈、队列、串、数组、广义表、树、二叉树、图、查找、内部排序、外部排序、动态存储管理和文件等基本类型的数据结构。本书中的算法都已经过调试，不用修改就能在 Turbo C2.0 系统下正常运行。

本书可作为高等学校计算机及其相关专业的教材，也可作为自学者或各种计算机培训班的教材。

图书在版编目（CIP）数据

数据结构(C 语言版) / 秦玉平，马靖善主编. —2 版. —北京：清华大学出版社，2012.3
（普通高校本科计算机专业特色教材精选·算法与程序设计）
ISBN 978-7-302-27494-0

Ⅰ. ①数… Ⅱ. ①秦… ②马… Ⅲ. ①数据结构－高等学校－教材 ②C 语言－程序设计－高等学校－教材 Ⅳ. ①TP311.12 ②TP312

中国版本图书馆 CIP 数据核字(2011)第 260589 号

责任编辑：焦　虹
封面设计：傅瑞学
责任校对：梁　毅
责任印制：王秀菊

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　**邮　　编**：100084
社 总 机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质量反馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：保定市中画美凯印刷有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：18.5　　**字　　数**：439 千字
版　　次：2012 年 3 月第 2 版　　**印　　次**：2012 年 3 月第1 次印刷
印　　数：1～3000
定　　价：30.00 元

产品编号：038143-01

出版说明

INTRODUCTION

在我国高等教育逐步实现大众化后，越来越多的高等学校将会面向国民经济发展的第一线，为行业、企业培养各级各类高级应用型专门人才。为此，教育部已经启动了“高等学校教学质量和教学改革工程”，强调要以信息技术为手段，深化教学改革和人才培养模式改革。如何根据社会的实际需要，根据各行各业的具体人才需求，培养具有显著特色的人才，是我们共同面临的重大问题。具体地说，培养具有一定专业特色和特定能力的计算机专业应用型人才是计算机教育要解决的重要问题。

为了适应21世纪人才培养的需要，培养具有特色的计算机人才，急需一批适合各种人才培养特点的计算机专业教材。目前，一些高校在计算机专业教学和教材改革方面已经做了大量工作，许多教师在计算机专业教学和科研方面已经积累了许多宝贵经验。将他们的教学和科研成果转化为教材的形式，向全国其他学校推广，对于深化我国高等学校的教学改革具有十分重要的意义。

清华大学出版社在经过大量调查研究的基础上，决定组织编写《普通高校本科计算机专业特色教材精选》丛书。本套教材是针对当前高等教育改革的新形势，以社会对人才的需求为导向，主要以培养应用型计算机人才为目标，立足课程改革和教材创新，广泛吸纳全国各地高等院校优秀教师参与编写，从中精选出版确实反映计算机专业教学改革成果的特色教材，供普通高等院校计算机专业学生使用。

本套教材具有以下特点。

1. 编写目的明确

本套教材是在深入研究各学校办学特色的基础上，面向普通高校的计算机专业学生编写的。学生通过本套教材，主要学习计算机专业的基本理论和基本知识，接受利用计算机解决实际问题的基本训练，培养研究和开发计算机系统，特别是应用系统的基本能力。

2. 理论知识与实践训练相结合

根据计算学科的三个学科形态及其关系，本套教材力求突出学科的理论与实践紧密结合的特征，结合实例讲解理论，使理论来源于实践，又进一步指导实践。学生通过实践深化对理论的理解，更重要的是使学生学会理论方法的实际运用。在编写教材时突出实用性，并做到通俗易懂、易教易学，使学生不仅知其然，还要知其所以然，更要会其如何然。

3. 注意培养学生的动手能力

力求做到：既注重培养学生分析问题的能力，也注重培养学生解决问题的能力，以适应新经济时代对人才的需要，满足就业要求。对每种教材都增加了能力训练部分的内容，以使学生通过学习和练习，能比较熟练地应用计算机知识解决实际问题。

4. 注重教材的立体化配套

对大多数教材都将陆续配套教师用课件、习题及其解答提示、学生上机实验指导等辅助教学资源。对有些教材还提供能在我社网站上下载的文件，以方便教学。

由于各学校的培养目标、教学要求和办学特色所不同，所以对特色教学的理解也不尽一致。我们恳切希望大家在使用教材的过程中，及时提出批评和改进意见，以便我们做好教材的修订改版工作，使其日趋完善。

我们相信经过大家的共同努力，这套教材一定能成为特色鲜明、质量上乘的优秀教材。同时，我们也希望通过本套教材的编写与出版，为“高等学校教学质量和教学改革工程”作出贡献。

清华大学出版社

前言

PREFACE

数据结构是计算机专业的一门重要的专业必修课，是绝大多数高等学校招收计算机专业及相关专业硕士研究生的必考科目之一。

本课程主要研究数据在计算机中的存储和操作，它涉及一系列较为实用的算法，这些算法在实际的程序设计中是非常有用的。但这门课程内容丰富、学习量大，其算法又十分抽象。经过多年的教学实践，我们总结出一些该课程的特点和教学方法，为此，编写了这部教材，以满足广大同学的要求和计算机教学的需要。

本书是在第1版的基础上修改而成的，主要是将原书第2章分解为两章，并对全书中的算法重新进行了编写和补充，使之更加适合于读者学习和掌握各种算法的基本思想和用法。

本书共分11章。第1章为概述，主要介绍了数据结构的简单发展史、基本概念和算法的描述与分析方法；第2章为线性表，主要介绍了顺序表和各种链表的存储表示与实现；第3章为特殊线性表，主要介绍了栈、队列和串的存储表示与实现；第4章为数组和广义表，主要介绍了数组和广义表的存储表示与实现；第5章为树和二叉树，主要介绍了二叉树的基本知识、性质、存储、遍历及其应用；第6章为图，主要讨论图的基本概念、存储、遍历及其应用；第7章为查找，主要介绍了静态查找、动态查找和散列表；第8章为内部排序，分别介绍了几种常用的排序算法及性能；第9章为外部排序，主要研究在内存和外存之间如何调动和组织数据进行排序；第10章为动态存储管理，主要研究系统如何响应用户的请求分配内存和回收内存；第11章为文件，主要讨论文件在外存储器中的表示方法和各种运算的实现方法。

本书的算法都用C语言函数实现，不作任何修改就可被其他函数调用。本书结构合理，内容紧凑，知识连贯，逻辑性强。为了使读者更好地掌握各章节内容，各章末均配有精选的大量习题，可使读者快速熟悉和掌握所学的知识。本书既可作为计算机专业的本、专科教材，也可作为与计算机学科相关的其他专业的教材。

本书第1、5章由马靖善编写，第2~4章由秦玉平编写，第6章由周连秋编写，第7章由王丽君编写，第8章由冯佳昕编写，第9章由沈泽刚编写，第10章由彭霞编写，第11章由李春杰编写。全部书稿由范立南和秦玉平审校，所有算法由马靖善和王丽君调试。

在本书编写过程中，编者参考了大量有关数据结构的书籍和资料，在此对这些参考文献的作者表示感谢。

由于编者水平有限，书中难免存在错误和不当之处，恳请广大读者批评指正，以便再版时改进。

注：目录中带**的章节为可选内容；带*的章节为专科选学。

编者

目录

CONTENTS

第1章 概 述

CHAPTER

1.1 数据结构的发展

计算机从开始发展到现在已经深入到了社会的各个领域，应用越来越广泛。人们已经离不开计算机了，计算机的应用与普及对人类的生产和生活产生了巨大的影响。在计算机上开发的各种软硬件产品层出不穷。就编程而言，一是要提高程序的执行效率，二是要想办法降低存储空间需求。对同一个问题，每个人的理解不完全一样，编程的方法和手段也不尽相同，但是，追求完美是人们的共识。那么，如何才能做到这一点呢？为了编写出“好”的程序，必须分析待处理对象的特性以及各个对象之间的关系。这就是“数据结构”这门学科形成和发展的背景。

数据结构是一门综合性的学科，它是在20世纪60年代末期开始形成和发展起来的。1968年，在美国一些大学的计算机系开始开设了这门课程。当时，数据结构几乎和图论，特别是表、树等理论为同义语。随后，数据结构这个概念被扩充到包括网络、集合代数论、格、关系等方面，从而变成了现在称为离散数学的内容。1968年，美国学者唐·欧·克努特(Knuth D. E)教授开创了数据结构的最初体系。他所著的《计算机程序设计技巧》第一卷《基本算法》是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中的主要内容。人们越来越重视数据结构，认为程序设计的实质是对确定的问题选择一种好的结构及合适的算法。从20世纪70、80年代开始，各种版本的《数据结构》著作和教材相继问世。

目前，数据结构已不仅仅是各个高校计算机专业的核心课程，也是其他相关专业的主要选修课程之一。它是集数学、计算机硬件和软件于一身的综合学科。它以数学方法为基础，研究如何更加合理有效地组织数据，编制出高质量的程序。数据结构作为一门较新的学科还在不断地发展，也越来越受到专业人士的关注。

很多计算机工作者认为，程序设计的实质就是通过分析问题，确定数学模型和算法，然后再选择一个好的数据结构。即

程序＝算法＋数据结构

因此，建议读者在学习数据结构这门课程时应做到以下4点：

（1）牢记典型算法的基本思想和求解步骤。

（2）注意各种结构之间的相互联系和区别。

（3）按照算法编制程序，上机调试。

（4）分析遇到的问题，并研究解决问题的方法。

学习这门课程，要有前驱课程的准备。本教材中的算法都是用C语言编写的，不用改动就能上机运行，所用的编程环境为Turbo C。建议读者在学习过程中一定要牢牢掌握数据结构中的一些基本概念和典型算法的基本思想，并注重上机实验的过程，它可以加深对所学算法的理解和掌握，同时也有助于提高编程能力。

1.2 基本概念

要学好数据结构这门课程，必须要明确各种概念及相互之间的联系。本节只介绍一些主要的基本概念，其他相关术语将在后续章节中陆续讲述。

1. 数据

数据(data)是对客观事物的符号表示。在计算机学科中，数据是指所有能输入到计算机中，并能被计算机程序所处理的符号的总称。因此，除了日常所习惯的符号，比如数字构成的整数和实数、字母构成的串之外，还有标点符号、键盘符号，甚至于图形、图像、声音等也都是数据的表示形式。

2. 数据元素和数据项

数据元素(data element)是描述数据的基本单位。**数据项**(data item)是描述数据的最小单位。在计算机中表示数据时，都是以一个数据元素为单位。比如，一个整数表示的一个数据元素，一条记录表示的一个数据元素等。而用一条记录表示一个数据元素时，这条记录中一般还会有多个描述记录属性的分项，称为数据项。比如，描述一台自行车的记录中可以包括车型、颜色、出厂日期、材质等分项，描述一个班级的记录中可以包括班级名、人数、男女生比例、教室、班委会成员和团支部成员等分项。数据项是具有独立含义的最小标识单位。

3. 数据对象

数据对象(data object)是性质相同的数据元素的集合。数据是非常广泛的一个概念，是用来描述千变万化的客观世界的。从数据中取出一部分，而这部分元素都有共同的性质，这些数据元素就可以组成一个数据对象。实质上，数据对象是数据的一个子集。比如，整数集、字符集、由记录组成的文件等。

4. 数据结构

数据结构(data structure)是由数据和结构两部分组成的。其中，数据部分是指数据元素的集合；结构就是关系，结构部分就是指数据元素之间关系的集合。所以说，数

据结构是指数据元素的集合及数据元素之间关系的集合。概括而言，数据结构是指相互之间有一种或多种特定关系的数据元素的集合。在计算机上要处理数据，就要保存数据及它们之间的关系。在这里，关系就是结构。通常，数据之间有如下 4 种关系。

(1) **集合结构**：结构中的数据元素除了“同属于一个集合”的关系外，再无其他关系。

(2) **线性结构**：结构中的数据元素之间存在着“一对一”的邻接关系。比如，生产过程中的流水作业、体育比赛中的接力赛跑等。

(3) **树状结构**：结构中的数据元素之间存在着“一对多”的关系。比如，多米诺骨牌、政府组织机构等。

(4) **图状结构**或**网状结构**：结构中的数据元素之间存在着“多对多”的关系。比如，城市交通图、电话网等。

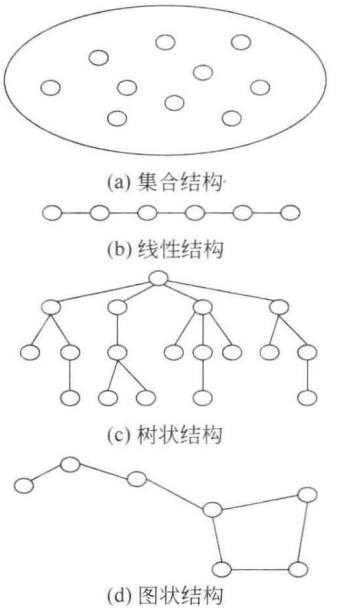

图 1.1　四种基本结构关系图

图 1.1 是上述 4 种结构的关系图。其中，树状结构和图状结构又称为非线性结构。由于集合中数据元素之间的关系是非常松散的，因此，常用其他几种结构来描述集合。数据结构依据抽象描述方式和机内存储形式可分为逻辑结构和物理结构两大类。

(1) **逻辑结构**(logical structure)是以抽象的数学模型来描述数据结构中数据元素之间的逻辑关系。通常用二元组来描述这种关系：

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

其中，D 是数据元素的有限集，R 是 D 上的关系的有限集。例如，复数的逻辑结构可以用如下的二元组来描述：

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

其中，$D=\{x|x\text{ 为实数}\}$，$R=\{<x,y>|x,y\in D,x\text{ 为实部},y\text{ 为虚部}\}$。其中，$<x,y>$表示的是一个**有序偶**，即 x 和 y 有顺序关系，$<x,y>$不等价于$<y,x>$。常用(x,y)表示**无序偶**，即 x 和 y 无顺序关系，(x,y)等价于(y,x)。若在 D 中任取两个实数，比如 5 和 3，则通过关系$<5,3>$可以表示一个复数 5+3i，而通过关系$<3,5>$可以表示一个复数 3+5i。

(2) **物理结构**(physical structure)又称存储结构，是数据结构在计算机内的存储表示，也称内存映像。一个存储在内存中的数据元素又称为**结点**，数据元素中的每个数据项又称为**域**。因此，数据元素或结点可以看成是数据元素在计算机中的映像。数据元素可以存放到内存某个单元中。那么如何存储数据元素之间的关系呢？在计算机内部主要是用顺序存储和链式存储这两种结构来表示数据元素之间的逻辑关系。顺序存储结构的特点是用物理地址相邻接来表示数据元素在逻辑上的相邻关系。链式存储结构的特点是逻辑上相邻接的数据元素在存储地址上不一定相邻接，数据元素逻辑上的相邻关系是通过指针来描述的，常用它来描述树状结构和图状结构在计算机内的存储。除这两种存储结构外，在计算机内还有索引存储结构和散列存储结构等，其实质都是通过顺序存储结构和链式存储结构复合而成。

逻辑结构和物理结构是描述数据结构的密不可分的两个方面。任何一个算法的设计取决于选定的逻辑结构，而算法的实现则取决于依托的存储结构。

5. 数据类型

在客观世界中，任何数据元素都应该有它自身的取值范围和它所允许进行的运算操作。比如，车是一个数据对象，车族中有各种各样的汽车、火车、三轮车、自行车等。如果给车安装一对翅膀那就变成了飞机，超出了车的范围。车能进行的操作是安装、行驶、载人、运货、比赛等。如果让车到天上去飞，那就太难为它了。因此，数据类型(data type)就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，C语言中的基本整型(signed int)，它的值集是−32 768～32 767，在这个值集上能进行的操作有加、减、乘、除和取余数等；而在实型(float)上就不能进行取余数的操作。按值的不同特性，数据类型又可分为不可分解的**原子类型**及可分解的**结构类型**。比如C语言中的整型、实型、字符型就属于原子类型，而数组、结构体和共用体类型就属于结构类型，可由其他类型构造得到。

6. 抽象的数据类型

抽象的数据类型(abstract data type，ADT)是指一个数学模型以及定义在该模型上的一组操作。抽象数据类型的定义仅取决于它的一组逻辑特性，而与其在计算机内部如何表示和实现无关。即不论其内部结构如何变化，只要它的数学特性不变，都不影响其外部的使用。抽象的数据类型可以分为如下3种类型：

(1) **原子类型**(atomic data type)，其值是不可分的。

(2) **固定聚合类型**(fixed-aggregate data type)，其值由确定数目的成分按某种结构组成。

(3) **可变聚合类型**(variable-aggregate data type)，其值由不确定数目的成分构成。

一个抽象的数据类型的软件模块通常包含定义、表示和实现三个部分。

7. 多型数据类型

多型数据类型(polymorphic data type)是指其值的成分不确定的数据类型。

1.3 算法描述与分析

要解决实际问题，就是要找出解决问题的方法。要用计算机解决实际问题，就要先给出解决问题的算法，再依据算法编制程序完成要求。所谓**算法**(algorithm)就是对求解问题步骤的一种描述，也称为**算法设计**。描述算法的工具有很多，比如自然语言、框图、计算机语言程序、伪代码、类计算机语言等。本书主要采用C语言函数来描述算法，但在书写算法时可能会忽略一些细节，如变量定义，交换两个变量的值等。读者只要补充上这些细节，再添加上主函数(main)，就可以上机调试算法程序了。

1. 算法的五个重要特性

(1) **有穷性**：一个算法必须(对任何合法的输入值)在执行有穷步之后结束，且每一步都可在有穷的时间内完成。这也是算法与程序最主要的区别，程序可以无限地循环下去。如操作系统的监控程序，在计算机启动后就一直在监测着操作者鼠标的动作和输入的命令。

(2) **确定性**：算法中的每一条指令都必须有明确的含义，不应使读者产生二义性。并且，在任何条件下，算法只有唯一的一条执行路径，即对于相同的输入只能得到相同的输出。

(3) **可行性**：一个算法是可以被执行的，即算法中的每个操作都可以通过已经实现的基本运算执行有限次来完成。

(4) **有输入**：根据实际问题需要，一个算法在执行时可能要接收外部数据，也可能无需外部输入。因此，一个算法应有零个或多个输入，这取决于算法本身要实现的功能。

(5) **有输出**：一个算法在执行完成后，一定要有一个或多个结果或结论。这就要求算法一定要有输出，这些输出是与输入有着某些联系的量。

2. 算法的评价

通常，解决同一个问题，不同的人有不同的想法，即使是同一个人，在不同的时间里可能对同一个问题的理解也不完全相同。而算法是依据个人的理解和想法人为设计出来的求解问题的步骤，不同的人，或同一个人在不同的时间里设计出来的算法也不尽相同。那么哪种算法设计得好？如何评价一个算法的好与不好？通常，在算法设计时应该考虑如下 5 个方面。

(1) **正确性**：这是算法设计最基本的要求。算法应该严格按照特定的规格说明去设计，要能够解决给定的问题。但是，“正确”一词的含义在通常的用法中有很大的区别，大体上可分为以下 4 个层次：

① 依据算法所编制的程序中不含语法错误。

② 程序对于几组输入数据能够得到满足规格说明要求的结果。

③ 程序对于经过精心挑选较为苛刻的几组输入数据也能够得到令人满意的结果。

④ 程序对于所有符合要求的输入数据都能得到正确的输出。

对于大型软件需要进行专业测试。一般情况下，通常以第三方要求作为衡量算法正确性的标准。

(2) **可读性**：设计算法的主要目的是解决实际问题。在设计实现一个项目时，往往不是一个人独立完成。如果别人看不懂你设计的算法，怎么去交流？又如何依据算法去编制程序？为了达到可读性的要求，在设计算法时，一般要使用有一定意义的标识符给变量、函数等起名，达到“见名知意”。再有，可以在算法的开头或指令的后面加注释，解释算法和指令的功能。

(3) **健壮性**：当输入不合法数据时，算法能作出相应的反应或进行适当的处理，避免带着非法数据执行，从而导致莫名其妙的结果。

(4) **高效率**：依据算法编制的程序运行速度较快。

(5) **低存储**：依据算法编制的程序运行时所需内存空间较少。

对于一个系统设计人员来说，前三项很容易实现。在使用计算机软件时，人们更加注重于软件的运行速度，而后两项恰恰是影响速度的主要因素。