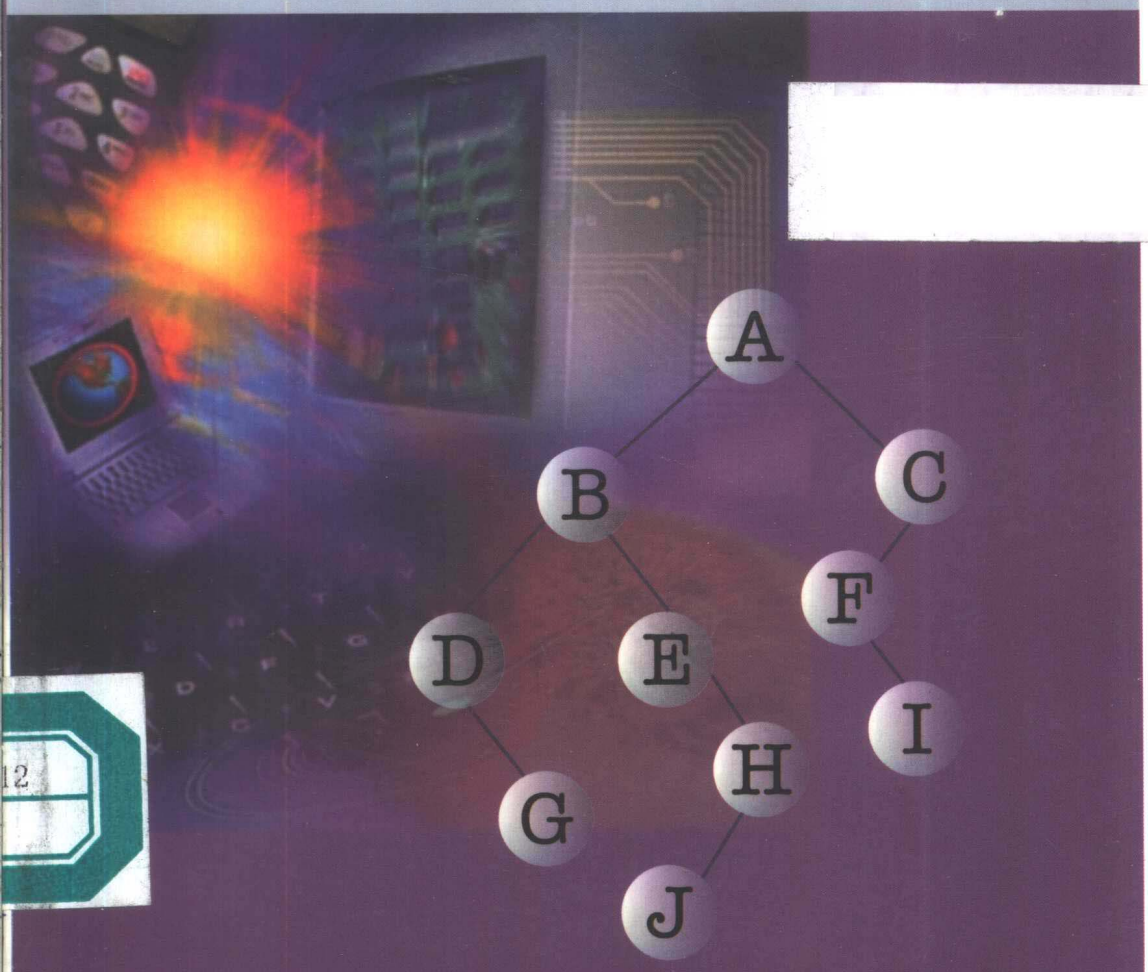
信息管理与信息系统专业系列教材

# 数据结构

杨开汉 主编　王少波 副主编




科学出版社

信息管理与信息系统专业系列教材

# 数据结构

杨开汉 主 编
王少波 副主编

科学出版社
2000

## 内 容 简 介

本书采用C语言描述算法，进行了适当的算法复杂性分析。内容以应用需要取舍编排。全书共10章，分别介绍了线性表、栈、队列、树、图、查找、内部分类、文件、外部分类等。每章最后有练习题。

本书按经济信息管理专业本科四年制教学计划编写，也适合其它管理专业或计算机应用专业的师生阅读。

**图书在版编目(CIP)数据**

数据结构/杨开汉主编 王少波副主编.-北京：科学出版社，2000
（信息管理与信息系统专业系列教材）
ISBN 7-03-006941-2

Ⅰ.数… Ⅱ.杨… Ⅲ.数据结构 Ⅳ.TP311.12

中国版本图书馆CIP数据核字（2000）第00614号

科学出版社 出版
北京东黄城根北街16号
邮政编码：100717

北京双青印刷厂 印刷

新华书店北京发行所发行 各地新华书店经售

*

2000年6月第 一 版 开本：787×1092 1/16
2000年6月第一次印刷 印张：18 3/4
印数：1—4 500 字数：442 000

**定价：25.00元**

(如有印装质量问题，我社负责调换〈环伟〉)

# 《信息管理与信息系统专业系列教材》

# 编 委 会 名 单

# 总　序

中南财经大学是财政部直属的一所以经济学科、管理学科为主，兼有法学、文学、哲学、理学等6个一级学科的，具有50年历史的高等学校。中南财经大学经济信息管理系始建于1978年，1980年开始招收本科生，是继中国人民大学之后在全国高校第二个建立信息管理专业的系，并于1990年，经国务院学位委员会批准建立信息经济硕士点，是全国首批设立的该专业4个硕士点之一。

改革开放20年，正是信息管理与信息系统专业不断建设成长的20年。中南财经大学信息系经过不断的探索和建设，在教学研究、师资队伍建设、教材建设、实验室建设及教学管理等方面均打下了良好的基础。

在专业发展和教材建设中，我们遵循教育必须为社会主义建设服务和必须面向现代化、面向世界、面向未来的要求，20年来，无论是专业目录调整前的管理信息系统专业，还是专业目录调整后的信息管理与信息系统专业，我们都努力在专业建设的深度以及市场经济建设的应用力度上下功夫，力求学生所学的专业知识在实际工作中能派上用场，在教学体系建设及教材建设中力求体现本专业的特色。经过20年艰苦奋斗与教学科研实践，中南财经大学信息管理与信息系统专业已经建立起规模适当，多层次多形式的办学体系；初步形成多学科有机结合，互相渗透的专业特色；建立了结构合理的教师队伍；具备了比较完善的办学条件；取得了一批先进水平的科研成果，溪国家培养了大批受社会欢迎的信息管理专门人才。

为了建设一套有信息管理与信息系统专业特色的教材，我们长期以来在加强基础、拓宽知识面、增强适应性、建立主动适应社会主义建设需要和适应现代科学技术、文化发展趋势的教学内容以及课程结构等方面搜集了大量的素材和案例，特别是在理论联系实际，面向经济建设主战场，强化学生的动手能力，结合最新的科技发展以及在教材中融进各位教师的研究成果上花了不少的精力。1998年我们按照教育部公布调整后的新专业目录，组织了两个小组到兄弟学校调查研究，进行了多次座谈和研讨，进一步明确了**信息管理与信息系统专业的性质是以系统的方法、现代信息处理技术来研究人类管理活动规律及其应用的学科**。它融合了管理学、经济学、计算机科学与技术等学科的知识，以系统观点为指导，运用定性与定量结合的方法及相关学科的研究手段，深入研究并有效地解决社会中各类信息管理问题。本专业的目标是：培养具备现代管理学理论基础、计算机科学技术知识及应用能力，掌握系统思想和信息系统分析与设计方法以及信息管理等方面的知识与能力，能在国家各级管理部门、工商企业、金融机构、科研单位等部门从事信息管理以及信息系统分析、设计、实施管理和评价等方面的高级专门人才。本专业的培养要求是：学生主要学习经济、管理、数量分析方法、信息资源管理、计算机及信息系统方面的基本理论和基本知识，接受系统和设计方法以及信息管理方法的基本训练，具备综合运用所学知识去分析和解决问题的基本能力。本专业的毕业生应具备以下的知识和能力：（1）掌握信息管理和信息系统的基本理论、基本知识；（2）掌

握管理信息系统的分析方法、设计方法和实现技术；（3）具有信息组织、分析研究、传播与开发利用的基本能力；（4）具有综合运用所学知识分析和解决问题的基本能力；（5）了解本专业相关领域的发展动态；（6）掌握文献检索、资料查询、收集的基本方法，具有一定的科研和实际工作能力。

基于上述思想，我们修订了信息管理与信息系统专业教学计划，相应地修订了相关课程的教学大纲，组织人员编写出有信息管理与信息系统专业特色的教材，供教学之需。经反复讨论，确定出版18种图书作为信息管理与信息系统专业系列教材，即：该套丛书包括以下18种教材：

《计算机实用技术基础》
《离散数学》
《数据结构》
《数据库原理与设计》
《计算机网络》
《计算机操作系统》
《管理信息系统分析与设计》
《计算机组成原理》
《多媒体与信息管理》
《管理决策分析》
《信息管理学》
《应用数理统计》
《运筹学（一）》
《运筹学（二）》
《经济预测方法》
《高等数学》
《线性代数》
《语言程序设计》

本套教材得以顺利出版，得到了科学出版社的大力支持，我代表本套教材的各位编写人员向科学出版社表示由衷的感谢！

由于水平所限，在陆续出版的系列教材中错误难免。望读者不吝赐教，以资改进，在此一并致谢！

邱家武

1999年元旦于中南财经大学

# 前　言

本书是按高等财经院校经济信息管理专业本科四年制教学计划数据结构课程教学大纲要求编写的教材，也适应于其它管理专业和计算机应用专业。本书还可以作为计算机科技工作者及其有关专业人员的参考书。

数据结构发展至今，已成为一门比较成熟的课程。它是计算机系统软件和应用软件研制者的必修课程。数据结构理论的应用范围已经深入到编译系统、操作系统、数据库、人工智能、信息科学、企业管理、系统工程、计算机辅助设计及其它信息管理的应用，它高效率并得心应手地解决常遇到的非数值计算应用问题。以此为目的，本书在阐明各种基本数据结构概念及其主要运算的算法时，还着重叙述其应用，尤其是在经济信息管理方面的应用。

本书简明扼要地叙述了各种基本数据结构的概念，包括数据结构的逻辑定义、物理实现及其运算，并举例说明怎样用这些抽象的概念来解决实际问题。叙述时，由浅入深，由简及繁，回避了复杂的数学定义与推导，力求通俗易懂，以使几乎没有计算机专业基础知识的读者也能顺利地自学全书的内容。通过本书的学习不仅能正确地掌握数据结构的基本理论，并能运用这些理论来解决实际问题。

本书系笔者集多年来从事计算机软件设计实践及讲授数据结构课程的体会，并参考分析了国内外数据结构书籍文献编写而成。内容的取舍以重应用为特征而区别于目前已出版的其它数据结构书籍。本书采用广泛使用的 C 语言描述算法，并进行了适当的算法复杂性分析。在使用 C 语言描述算法时为了使算法一目了然、清晰易读，有时省略了一些 C 语言的复杂语法规定，学生阅读时需加注意。

数据结构不但是一门理论性很强的课程，同时又是一门实践性很强的课程，在每一章的最后都安排了适当的习题，供读者练习。教师在教学中还应安排上机实习，以加强实践。

本书共分 9 章，介绍了线性表、栈、队列、树、图、查找、内部分类、文件、外部分类等基本数据结构及其算法与应用例题。全书内容丰富，逻辑性强，条理清楚，按教学大纲规定的 72 学时取材编写。读者学完本书后，对数据结构会有较全面且系统的认识，能够提高非数值计算应用软件设计的水平。

本书所有例题均在 IBM－PC 机中演练通过。本书由杨开汉教授主编、王少波副主编，具体分工是：第一、五、七、八、九、十章由杨开汉编写；第二、三、四章由王少波编写；第六章由屈振新编写，最后由杨开汉教授审阅了全书并作了修改。

本书在编写过程中，参考了清华大学严尉敏教授、国防科技大学王广芳教授分别编写的《数据结构》，同时，还翻阅了大量外文资料。在此一并表示感谢。

本书在编写和出版过程中得到中南财经大学各级领导的支持和鼓励，对此我们表示衷心的感谢。

由于水平有限，疏漏之处在所难免，望读者悉心指正。

作　者

1999 年 7 月

# 目　录


**第一章　绪论** …… ( 1 )
第一节　什么是数据结构 …… ( 1 )
第二节　基本术语介绍 …… ( 2 )
第三节　数据结构的发展和它在计算机科学中的地位 …… ( 3 )
第四节　学习数据结构的基本知识 …… ( 4 )
习题一 …… ( 10 )
**第二章　线性表** …… ( 11 )
第一节　线性表的基本概念 …… ( 11 )
第二节　线性表的顺序存储结构 …… ( 12 )
第三节　线性表的链式存储结构 …… ( 16 )
第四节　数组 …… ( 32 )
第五节　多重链表 …… ( 40 )
第六节　链表应用 …… ( 43 )
第七节　信息处理实例 …… ( 47 )
习题二 …… ( 63 )
**第三章　栈与队列** …… ( 65 )
第一节　栈 …… ( 65 )
第二节　队列 …… ( 74 )
第三节　队列的应用实例 …… ( 83 )
习题三 …… ( 90 )
**第四章　串** …… ( 92 )
第一节　串的逻辑特征 …… ( 93 )
第二节　串的存储结构及其运算实现 …… ( 95 )
习题四 …… ( 104 )
**第五章　树** …… ( 106 )
第一节　树、森林概述 …… ( 106 )
第二节　二叉树 …… ( 108 )
第三节　二叉树的遍历 …… ( 118 )
第四节　线索树 …… ( 128 )
第五节　一般树的表示与遍历 …… ( 134 )
第六节　树的应用举例 …… ( 138 )
习题五 …… ( 152 )
**第六章　图** …… ( 154 )
第一节　基本概念 …… ( 154 )
第二节　图的存储表示 …… ( 156 )

第三节 图的遍历 …… ( 160 )
第四节 图的连通性问题 …… ( 164 )
第五节 有向图及其应用 …… ( 170 )
第六节 最短路径 …… ( 177 )
第七节 图的应用举例 …… ( 182 )
习题六 …… ( 191 )
**第七章 查找** …… ( 193 )
第一节 基本查找技术 …… ( 193 )
第二节 树查找 …… ( 203 )
第三节 HASH 查找技术 …… ( 221 )
习题七 …… ( 229 )
**第八章 内部分类** …… ( 230 )
第一节 概述 …… ( 230 )
第二节 插入分类 …… ( 232 )
第三节 交换分类 …… ( 237 )
第四节 选择分类 …… ( 242 )
第五节 合并分类 …… ( 246 )
第六节 基数分类 …… ( 249 )
习题八 …… ( 254 )
**第九章 文件** …… ( 255 )
第一节 外存设备和信息存取 …… ( 255 )
第二节 数据文件的基本概念 …… ( 258 )
第三节 文件组织的基本方法 …… ( 261 )
第四节 顺序文件 …… ( 262 )
第五节 索引文件 …… ( 264 )
第六节 索引顺序文件 …… ( 266 )
第七节 直接存取文件（散列文件） …… ( 268 )
第八节 倒排文件 …… ( 269 )
习题九 …… ( 271 )
**第十章 外部分类** …… ( 272 )
第一节 外部分类的方法 …… ( 272 )
第二节 外部分类的效率分析 …… ( 273 )
第三节 “败者树”法多路平衡归并 …… ( 274 )
第四节 初始归并段的产生 …… ( 277 )
第五节 缓冲区的动态处理 …… ( 280 )
第六节 最佳归并树 …… ( 282 )
第七节 磁带外部分类 …… ( 283 )
习题十 …… ( 288 )

# 第一章　绪　论

自从世界上第一台电子计算机问世以来，在短短的50余年时间里计算机科学的发展之快，已远远超过人们原来对它的估计。计算机的应用范围不仅已深入到各个领域，而且在具体应用上也不再只是限于计算。在非数值计算应用中，计算机加工的数据对象往往是大量的数据。这些数据存取、查找、插入和删除等等操作效率的提高，仅仅依赖程序设计的技巧已经无法达到目的，必须对这些被加工数据的组织形式加以研究，找出最佳的数据组织形式，并与好的程序设计技巧相配合，才能达到提高效率的目的。其实，有时还不只是一个效率的问题，在一些情况下，若没有好的数据组织形式，就根本无法完成所需要做的工作。这也是我们为什么要学习并研究数据结构这门科学的原因。

## 第一节　什么是数据结构

为了说明这一问题，我们先回顾一下我们现实生活中的两个例子。当你拿起一本厚厚的汉语字典查找某一个汉字时，你首先必须知道你使用的字典的编码方法，然后才能按照偏傍部首、四角号码或者拼音等相应的编码方法较快地查到你所需要查找的汉字。你为什么能如此顺利地在几万个汉字中找到你所需要的汉字呢？这是由于字典中的每一个汉字都是按偏傍部首或四角号码或者拼音的规律严格地安排在它应处的页行（编码）上。查找时也必须遵循相同的规律。倘若不按某一规律，将几万个汉字任意安排，你为了查找某一个汉字就不得不从字典的第一页开始逐页地查找了。由这个例子我们可以看出高的查找效率是与字典中汉字的安排规律，也就是组织形式密切相关的。自然，对于不同组织形式必须采用相应的查找办法才能达到提高效率的目的。就如同四角号码字典，采用偏傍部首方法无从查找一样。

另一个例子是，当你要给你的朋友打电话时，你不知道他的电话号码，也不必着急，翻开桌上的电话号码本你很快找到他们单位的电话号码，你给这个单位打了电话，请该单位的人帮你找到你的朋友和你通话。在这个例子中首先你按系统或街道等规律，在厚厚的电话号码簿上很快地找到了你朋友所在单位的电话号码，然后又通过你朋友单位的人再找你的朋友通话。

这是两个并不陌生的例子。字典和电话号码簿，是按不同的规律将汉字和电话号码这些数据进行组织与安排的，以适应不同的目的和要求。这样可以在实现与组织形式相应的目的时，大大地提高操作效率。请注意在第二个例子中，为了与你的朋友通话，经历了两个查找过程，查电话簿和传呼人找你的朋友。

上面说的就是日常生活中的“数据结构”的例子。这类例子想必大家都可以遇到。在计算机中，与此类似，大量的数据存储在存储器中（主存和辅存），等待加工。为了查找的方便应将这些数据按某些规律存放在存储单元中。这样才能在数据的加工过程中采用相应的办法高效率地进行数据的访问、插入和删除待等运算，从而也提高了数据加

工的效率，这种数据存放规律就是数据的组织形式即数据结构形式。

综上所述，我们可以给数据结构这门学科下个定义：**数据结构就是研究计算机中大量数据存储的组织形式，并定义相应的运算以提高计算机的数据处理能力的一门科学。**

## 第二节 基本术语介绍

在本章中，我们将用到许多术语和名词，我们现在给出确切的定义，以便在今后的学习中能有统一的概念。

**数据**（data）：在计算机中，数据这个名词的含义异常广泛，可以认为它是描述客观事物的数字、字符，以及所有能输入到计算机中并能为计算机所接受的符号的集合。

**数据元素**（data element）：它是数据的基本单位，是数据这个集合中的个体。在数据存储组织中，它是基本的处理单位，正由于此，它的含义也十分广泛。根据不同的需要它可以是一个数、一个字符、一个字符串、一个描写客观事物的记录，甚至可以是一篇文章。

**数据对象**（data object）：它是具有相同特性的数据元素的集合，是数据的一个子集。例如，当我们研究整数时，其数据对象是 {0，±1，±2，±3，...}，而当我们研究字母字符时，其数据对象是 {A，B，C，...}。

**数据类型**（data type）：它是程序设计语言中所允许的变量的种类，也是变量可以取的值和可以进行运算的集合。每一种程序设计语言都有一组它所允许的基本数据类型。如在 FORTRAN 语言中允许五种基本数据类型：整数、实数、双精度数、复数和布尔量。而每一种数据类型都有它取值的范围和所允许的运算。在 PASCAL 语言中，除提供整数等四种标准类型外，还有其它非标准类型，甚至允许自己定义数据类型和它的取值范围。各种语言所能提供的数据类型的多少决定了该语言功能的强弱。我们可以把数据类型看成程序设计语言中已经实现了的数据结构。如复数、数组等。因此，数据类型也可以认为是数据结构（包括逻辑结构和物理结构）的另一种叫法。

**算法**（algorithm）：算法是非空的、有限的指令序列，遵循它就可以完成某一确定的任务。算法这个词并不是计算机出现后才有的。早在古代就有欧几里德算法（求两个数的最大公因子）和孙子算法（求若干个数最小公倍数）。只是随计算机科学的发展，对算法的研究也飞跃发展。算法并不是指通常所说的程序，它有五大特征：

（1）有穷性。一个算法在执行有限步骤之后必须终止。

（2）确定性。一个算法所给出的每一个计算步骤必须是精确定义的，无二义性。

（3）可行性。算法中要执行的每一个步骤都可以在有限时间内完成，可行性与有穷性和确定性是相容的。

（4）输入。算法一般有一个或多个输入信息（个别的算法可能不要求输入信息）。这些输入量是算法所需的初始数据。它取自某一特定的集合。

（5）输出。算法一般有一个或多个信息输出，它是算法对输入信息的执行结果。由以上五个特征可以看出，算法是不同于一般程序的。例如，程序可在无外来干涉情况下一直执行下去，程序可以既无输入又无输出信息。

还有一点需要说明的是，数据结构的学习是在学员已经掌握了一门高级语言之后进

行的。所以高级语言中已经学习了一些术语和概念还将用到数据结构的学习中，但有时同一个术语在高级语言和数据结构中又会稍有区别，这些问题主要是由于问题提出的角度不同而带来的，高级语言中谈到的数据类型（既数据结构）主要是从应用角度提出的，而本书讨论数据结构时提到的同一种术语往往是从逻辑结构和物理实现上来讨论。请读者在遇到这些问题时联系前后文，体会其中的细微区别，这有利于我们搞清楚问题。例如，我们在第二章讨论数组（概念与实现）之前已经用到了高级语言中数组的概念了，在讨论分类、查找之前我们也不得不用到一些简单的分类、查找技术来说明问题，这是难以避免的。数据结构的学习是把问题研究的更深入、更广泛而已。

## 第三节 数据结构的发展和它在计算机科学中的地位

“数据结构”最早是1968年在美国被确定为一门独立的课程的。在此之前，数据结构课程的一些内容曾在其它课程中，如表处理语言中讲述。1968年，美国一些大学虽将数据结构规定为一门课程的内容，对但该课程的内容并未作明确的规定。其后发表的一些文件和出版的书籍中，对数据结构这个术语有各种不同的解释和理解。最初，数据结构几乎是图论，特别是表和树的理论的同义语，随后这个概念又扩充到包括网络、代数、集合论、关系等现在称之为“离散数据结构”的那些内容，它与现在的“数据结构”的内容结合在一起，统称为“数据结构”。

由于数据需要在计算机中处理，因此不能局限于数据本身的数学概念的研究，还必须考虑到数据的物理结构，即数据在存储器中如何存储的问题，这就进一步扩大了数据结构的研究范围。1968年著名的美国计算机科学教授唐·欧·克努特所著《计算机程序设计技巧》的第一卷《基本算法》，是第一本系统地阐述数据的逻辑结构以及运算的著作。从60年代末到70年代初出现了大型程序，程序与数据相对独立，结构程序设计成为程序设计方法学的主要内容，人们越来越感到数据结构的重要。认为程序设计的实质是为问题选择一种较好的数据结构加之一种好的算法。至今，对数据结构的研究仍在发展中。

由于数据结构是高级程序设计语言、操作系统、数据库、人工智能等课程的基础，同时，数据结构技术也广泛地应用于信息科学、系统工程、应用数学以及各种工程技术领域，所以数据结构不仅仅是计算机专业和信息管理专业的核心课程之一，而且也是其它与计算机（应用）有关的专业的必修课程。

数据结构的研究涉及的知识面十分之广，可以认为它是介于数学、计算机硬件和软件之间的一门核心课程。图1-1表示出数据结构与其它课程的关系。

数据结构的侧重点在于实践技术。我们研究各种数据结构的性质，定义相应的算法并分析算法的效率，同时研究各种数据结构的应用范例，其目的全在于引导我们设计好的数据结构与程序，提高计算机的数据处理能力。换句话说，数据结构的研究目标在于在解决具体数据处理问题时减少时间和空间（存储单元）的开销。

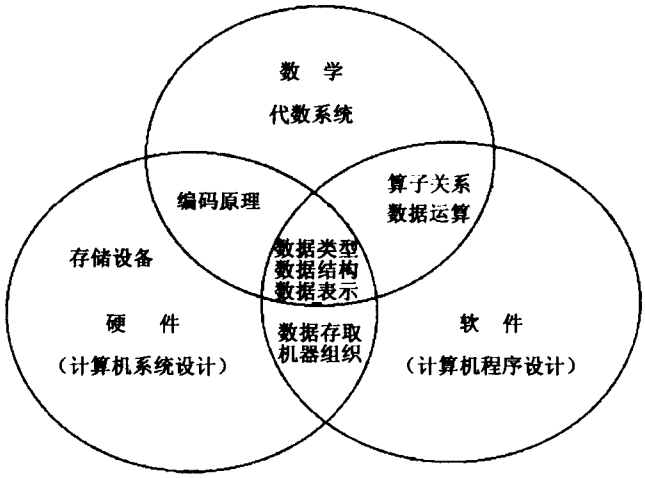



图 1-1 数据结构与其它课程的关系

## 第四节 学习数据结构的基本知识

### 一、算法分析方法说明

前面已经说到，在编制程序之前，首先应该选择一个恰当的数据结构和一个好的算法。那么，如何衡量一个算法的好坏呢？

显然，首先应确保算法是正确的，此外，通常还有三个方面的考虑：

(1) 依据算法所编制的程序在计算机中运算时所消耗的时间，即时间特性。

(2) 依据算法所编制的程序在计算机中运行时占有内存容量的大小（也要考虑占辅存容量的多少），即空间特性。

(3) 算法是否易读，易于转换成任何其它可运行的语言程序以及是否易于调试。

从主观上说，我们希望选择一个既不占很多存储单元，运行时间又短且简明易读的算法。然而，实际上不可能做得十全十美。往往是，一个看起来很简单的程序其运行时间要比复杂的程序慢得多，而一个运行时间较短的程序占用内存单元也多。因此，在不同的情况下应有不同的偏重选择。如果算法使用次数较少则应力求简明易读，易于转换成上机程序；若算法转换成的程序反复运行多次，则应选择运行时间尽可能少的算法。若待解决的问题数据量大，而所使用的计算机存储容量又较小，则相应算法应着重考虑如何节省内存单元。在本书中，主要讨论算法的时间特性，有时候也考虑一下空间特性。

一个程序在计算机上运行所消耗的时间主要取决下述因素：

(1) 程序运行时所需要输入的数据总量。

(2) 对源程序进行编译所需要的时间。

(3) 计算机执行每条指令所需要的时间。

(4) 计算机指令重复执行的次数。

上面四条，前三条取决于执行计算机的硬、软件系统的客观条件。因此，一般把第

四条即语句的重复执行次数作为算法的时间变量。

为此，我们引入语句频度的概念（frequency count）。所谓语句频度即为语句重复执行的次数。例如，两个 n×n 的矩阵相乘，其算法可描述如下：

```
void matrix-product(int a[],b[],c[],n);          重复次数
{
 for (i=1;i<=n;i++)                               n+1
   { for (j=1;j<=n;j++                            n(n+1)
     { c[i,j]:=0;                                 n²
       for (k=1;k<=n;k++)                         n²(n+1)
           c[i,j]=c[i,j]+a[i,k]*b[k,j];           n³
     }
   }
```

其中，每一语句的频度如上述算法右列所示。整个算法中所有语句的频度之和可约定作为该算法执行时间的度量，记作：

$$T(n) = 2n^3 + 3n^2 + 2n + 1$$

显然，它是矩阵阶 n 的函数，并且当 $n \to \infty$ 时，有 $T(n)/n^3 \to 2$，若引入“O”记号（读作“大 O”），则有 $T(n) = O(n^3)$，其意为在 n 较大时，算法的执行时间与 $n^3$ 成正比。或者说，$T(n)$ 数量级与 $n^3$ 数量级相同，我们称 $T(n)$ 为算法时间复杂性（time complexity）。

一般情况下，n 为问题规模的度量，如矩阵的阶，多项式的项数，图中顶点的个数等等。一个算法的时间复杂性为 $T(n) = O[f(n)]$。因此，通常可以通过判定程序段中重复执行次数最多的语句的频度来估算算法的时间复杂性。

例如，下面有三个简单的程序段。

(a) x=x+1;

(b) for (i=1;i<=n;i++) x:=x+1;

(c) for (i=1;i<=n;i++)
　　for (j=1;j<=n;j++) x:=x+1;

假定（a）中语句 x+1 不在任何的循环中，则语句频度为 1，其执行时间是一个常数，而在（b）中，同一语句执行 n 次，其频度为 n。显然，在（c）中，语句频度为 $n^2$。因此，这三个语句的时间复杂度为 $O(1)$，$O(n)$，$O(n^2)$。它们分别称为常数阶，线性阶和平方阶。算法还可能呈现的复杂度分别为对数阶 $O(\log_2 n)$、指数阶 $O(2^n)$等等。图 1-2 表示出不同数量级的特性曲线。由图可知，我们应当尽量选用对数阶 $O(\log_2 n)$ 的算法或线性阶 $O(n)$ 的算法，而应尽量避免指数阶 $O(2^n)$ 的算法。

对于较复杂的算法，我们可以将它分隔成容易估计的几个部分，然后利用 O 的求和原则得到整个算法的时间复杂性。例如，若算法的两部分的时间复杂性分别为 $T_1(n) = O[f(n)]$ 和 $T_2(n) = O[g(n)]$，则总的时间复杂性应为 $T_1(n) + T_2(n) = O(\max(f(n), g(n)))$。又若 $T_1(m) = O(f(m))$，而 $T_2(n) = O(g(n))$，则总的时间复杂性应为 $O(f(m) + g(n))$。

然而，很多程序的执行时间不仅依赖于问题的规模，而且随它处理的数据集不同而

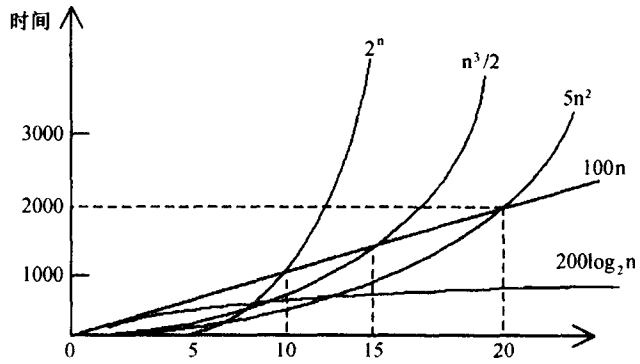



图 1－2 各种数量级的 T（n）

不同。例如，有的排序法对某些原始数据（如从小到大有序），其时间复杂度为 O（n），而对另一些数据就可能达到 O（$n^2$）。对于这类算法，除特别申明外，均应依据各种可能出现的数据集中最坏的情况来估计算法的时间复杂度。有时我们也在某个约定（如概率）下讨论算法的平均时间复杂性。

一个算法的时间复杂性是反应算法性能的重要标准。对于某一问题而言，如果算法时间复杂性的数量级越低，则说明算法的效率越高。也许有人说，现代计算机发展速度快得惊人，算法的效率无足轻重，然而，事实并非如此。在今天，人们需要处理的数据量越来越大，算法效率的高低对所能处理的数据量的多少有决定性的作用。当输入量急剧增加时，如果没有高效率的算法，单纯依靠提高计算机的速度，有时是无法达到要求的。可以通过下述实例来说明这个问题。

设有五个算法 $A_1$，$A_2$，$A_3$，$A_4$，$A_5$，它们的时间复杂性如表 1－1 所示。表中一个算法的时间复杂性是它处理完一个大小为 n 的输入所需要的单位时间数。当取单位时间为一毫秒时，算法 $A_1$ 可以处理完 1000 个数据输入，而算法 $A_5$ 则只能处理完 9 个数据的输入。表 1－2 给出了 1 秒钟、1 分钟和 1 小时内，这五个算法所能解决的输入量上界。从这里我们不难得到一个直观的概念：由于算法时间复杂性的数量级不同，它们在相同的时间里所能解决的问题相差是极为悬殊的。

**表 1－1**

| 算　法 | 时 间 复 杂 性 |
|---|---|
| $A_1$ | $T_1(n)=n$ |
| $A_2$ | $T_2(n)=n\log_2 n$ |
| $A_3$ | $T_3(n)=n^2$ |
| $A_4$ | $T_4(n)=n^3$ |
| $A_5$ | $T_5(n)=2^n$ |

当计算机的速度成倍提高时，比如假定下一代计算机的速度比当代计算机的速度快 10 倍或 10000 倍，我们来分析一下上面这五个算法所能处理的输入量的大小有何变化。表 1－3 说明了由于计算机速度的提高给每个算法所带来的处理能力的改变情况。算法

$A_1$ 和 $A_2$ 在计算机的速度提高 10 倍或 10000 倍后，同一时间里所能处理的输入量几乎也增加 10 倍或 10000 倍；而算法 $A_3$ 和 $A_4$ 就差一些，最令人沮丧的算法是 $A_5$，即使计算机的速度提高 10000 倍，算法 $A_5$ 在某一时间内所能处理的输入量不过比原来增加 13 个左右，真是廖廖无几。对于表 1-2 不难推出这样的结果：对于算法 A5 而言，一台高速计算机在 1 分钟内所能处理完的输入量只不过是比一台速度为它万分之一的低速计算机 1 分钟内所能处理完的输入量的两倍（这里假定取一毫秒作时间单位）。

**表 1-2 在某一时间内不同算法所能处理的输入量上限**

| 算法 | 时间复杂性 | 1 秒钟能处理的最大输入量 | 1 分钟能处理的最大输入量 | 1 小时能处理的最大输入量 |
|---|---|---|---|---|
| $A_1$ | $n$ | 1000 | $6\times10^4$ | $3.6\times10^6$ |
| $A_2$ | $n\log_2 n$ | 140 | 4893 | $2.0\times10^5$ |
| $A_3$ | $n^2$ | 31 | 244 | 1897 |
| $A_4$ | $n^3$ | 10 | 39 | 153 |
| $A_5$ | $2^n$ | 9 | 15 | 21 |

**表 1-3 计算机速度提高 10 倍和 1 万倍的效果**

| 算法 | 时间复杂性 | 速度提高前单位时间内能处理的最大输入量 | 速度提高 10 倍后单位时间内能处理的最大输入量 | 速度提高 1 万倍后单位时间内能处理的最大输入量 |
|---|---|---|---|---|
| $A_1$ | $n$ | $S_1$ | 10 $S_1$ | 10000 $S_1$ |
| $A_2$ | $n\log_2^n$ | $S_2$ | 对大的 $S_2$ 接近于 $S_2$ | 当 $10\log_2 S_2>\log_2 9000$ 时超过 9000 $S_2$ |
| $A_3$ | $n^2$ | $S_3$ | 3.16 $S_3$ | 100 $S_3$ |
| $A_4$ | $n^3$ | $S_4$ | 2.15 $S_4$ | 21.54 $S_4$ |
| $A_5$ | $2^n$ | $S_5$ | $S_5+3.32$ | $S_5+13.32$ |

由以上例子可以看出，随着计算机应用的发展和要求处理的信息量越来越大，分析算法效率，设计出高效的算法是多么重要。从这里也可以看出，我们为什么要学习数据结构。

程序运行所占的存储量，同时也是问题规模的函数，在书中，读者也会看到它和时间复杂性类似，我们将以空间复杂性（space complexity）来作为它的度量。

## 二、递归技术介绍

在数据结构的学习中，我们将经常使用递归技术。递归是一个十分重要的概念，递归技术在程序设计中是十分有用的。在 PASCAL 语言程序设计中大家已经遇到过递归问题，递归概念对于初学程序设计者来说往往觉得十分神秘，不容易接受。为了保证后面数据结构的学习，在这里，先对递归技术作简单介绍。

1．*递归函数*（recursive function）

一个使用函数自身给出定义的函数叫做递归函数。这似乎是一个循环的定义，不好

理解。其实不然，它并非是一个循环的定义。我们不妨先看下面的两个例子。

（1）阶乘函数。这是一个在数学中经常用到的函数。它的含义是给定一个整数 n，n 的阶乘被定义为 1 到 n 之间的所有整数的乘积。例如，5 的阶乘为 5 * 4 * 3 * 2 * 1 = 120，而 6 的阶乘为 6 * 5 * 4 * 3 * 2 * 1 = 720。0 的阶乘被定义为 1。为此，如果用“!”表示阶乘符号，我们可以这样来表示 n 阶阶乘函数的定义。

$$n! = 1 \qquad (\text{当 } n=0)$$

$$n! = n * (n-1) * (n-2) * \ldots * 1 \quad (\text{当 } n>0)$$

根据上面的定义，我们可以写出求整数 1 到 5 的阶乘函数的过程。

```
0! =1
1! =1
2! =2*1
3! =3*2*1
4! =4*3*2*1
5! =5*4*3*2*1
```

这就是计算机中阶乘函数的递归算法。它比较直观地体现了递归函数的特征。易写易读。上述算法上的语句 fact2：=x * fact2（x－1）是一个直接调用自身的过程。我们将算法中这种直接或间接调用自身的过程称为递归过程。它有着两种形式：直接递归过程和间接递归过程。

直接递归过程的一般形式为：

```
void   abc(参数表 1);
    {
      .
      .
      .
      p= abc(参数);
      .
      .
      .
      };
```

过程 abc 的直接调用了 abc 本身叫直接递归，但它的参数表的内容是有区别的。

间接递归过程的一般形式为：

```
void  abc(参数表 1);
   {
      .
      .
      .
      q= bcd(参数表);
      .
      .
      .
   }
```