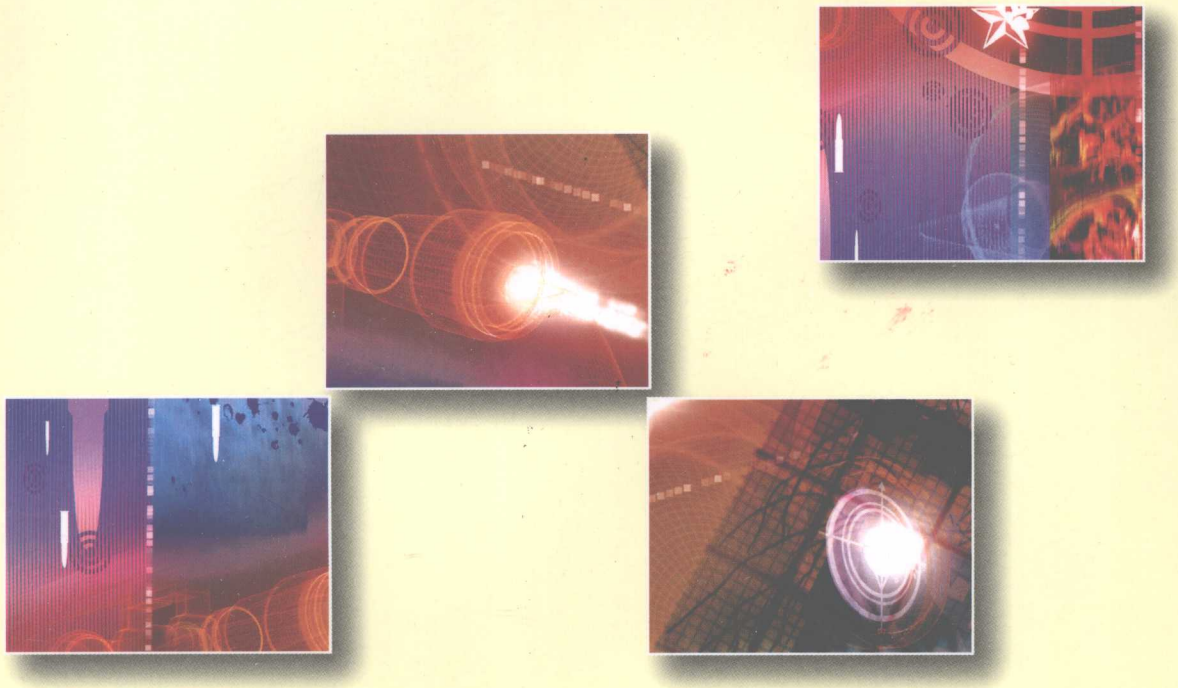
普通高等教育“十一五”国家级规划教材

高职高专计算机系列

数据结构与程序设计

文益民　周学毛　李　健　编著

陈焕文　主审



人民邮电出版社

POSTS & TELECOM PRESS

普通高等教育"十一五"国家级规划教材

高等学校计算机专业教材

数据结构与程序设计

[redacted]

[redacted]

普通高等教育“十一五”国家级规划教材

高职高专计算机系列

数据结构与程序设计

陈焕文 主审

文益民 周学毛 李 健 编著

人民邮电出版社
北京

图书在版编目（CIP）数据

数据结构与程序设计 / 文益民，周学毛，李健编著. —北京：人民邮电出版社，2008.9
普通高等教育“十一五”国家级规划教材. 高职高专计算机系列
ISBN 978-7-115-17793-3

Ⅰ. 数… Ⅱ. ①文…②周…③李… Ⅲ. ①数据结构—高等学校：技术学校—教材②程序设计—高等学校：技术学校—教材 Ⅳ. TP311

中国版本图书馆 CIP 数据核字（2008）第 032405 号

内 容 提 要

本书以提高学生的程序设计能力为宗旨，全面介绍了程序设计的基础知识、各种常用的数据结构以及排序、查找的各种算法及其应用。为了方便教学，书中各数据结构类型和基本运算首先用类 C 代码加以描述，并作了详细的注解。全书既注重原理，又强调实践，配有大量的图表和习题，概念讲解清楚，逻辑性强，可读性好。本书的主要特点在于：首次尝试采用“任务驱动”方式来设计教学内容，以帮助学生更好地理解所学知识；书中有大量以“课堂思考”形式出现的问题，能在恰当的时机激发思考，启发思维；使用脚注介绍计算科学发展史知识和其他相关知识，以拓展学生的知识范围。

本书可作为高职高专院校计算机专业教材，也可作为成人教育（面授或函授）教材，还可供广大从事计算机应用的科技人员参考。

普通高等教育“十一五”国家级规划教材
高职高专计算机系列

数据结构与程序设计

◆ 主　　审　陈焕文
编　　著　文益民　周学毛　李　健
责任编辑　张孟玮
执行编辑　左艾鑫

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京隆昌伟业印刷有限公司印刷

◆ 开本：787×1092　1/16
印张：13.75
字数：337 千字　　2008 年 9 月第 1 版
印数：1 – 3 000 册　　2008 年 9 月北京第 1 次印刷

ISBN 978-7-115-17793-3/TP

定价：24.00 元

读者服务热线：(010)67170985　印装质量热线：(010)67129223
反盗版热线：(010)67171154

前 言

计算机程序是计算技术的重要内容，而“数据结构”是计算机程序设计的重要理论和方法基础，它不仅是计算机专业的核心课程，而且已成为其他专业的重要教学内容。《数据结构》的教学目的是：学会分析需要计算机处理的数据对象的特性，并选择适当的数据结构、存储结构从而选择相应的算法；初步掌握算法性能分析的方法；初步掌握将实际问题求解转化为计算机程序的能力。学生通过本课程的学习，可得到更进一步的程序设计技能训练，提高编程能力，进而提高计算机软件工程能力。

很久以来，由于数据结构课程自身的抽象和严密，以及数据结构开设的时间通常在刚入大学的第二学期，老师感觉“数据结构”难教，学生普遍反映“数据结构”难学，学生很难独立完成算法的实现。基于上述问题，我们在编写本教材时充分考虑了学生的知识结构和教师的教学方法。本教材的编写宗旨是：既注重原理又注重实践；既注重抽象思维又注重形象思维；既方便自学又方便教学。本书主要具有以下特色。

1. 首次尝试采用“任务驱动”方式来设计教学内容。除第1章外，每一章当必要的基础知识介绍完毕后都安排一个问题作为本章学习需要解决的“任务”。该问题具有两个特征：（1）有一定的趣味性，能较好地激发学生的学习兴趣和解决问题的动力。（2）综合性较强，问题的解决需要使用到本章中的大多数知识点。

2. 通过“课堂思考”，提出问题拓展学生思维。“思考”不同于“习题”，“习题”的作用代替不了思考，因为习题在每个章节之后，一般要等到讲完一个章节后才能做，因此“习题”对于课堂教学是滞后的。采用提示“思考”的方式可以在教学中恰到好处地启发学生的思维。教材设置“课堂思考”标志通常有三种情况：第一种是提醒学生注意，第二种是启发学生基于当前知识基础进一步思考，第三种是提示本教材没有讲授的内容以引导学有余力的学生拓展自身知识面。

3. 在计算机专业的核心基础课程中增加计算机科学文化的知识，使学生在学习本教材的过程中，不但能学到专业知识，还能了解计算机科学发展历史的相关

知识和数据结构应用方面的知识。这对提高学生学习本课程的兴趣，拓宽学生的知识面大有好处。

4. 本教材还配有资源网站，网址为：http://jpkc.hunangy.com/c28/Course/Index.htm。网站的内容包括：教学课件、习题答案、数据结构课程的实践教学内容、国内外数据结构精品课程链接、本课程教学的 bbs 等。

本书包含 9 章内容：第 1 章绪论介绍数据结构和算法分析的基本概念及程序设计基础；第 2～5 章介绍线性结构及一部分与线性结构密切相关的非线性结构；第 6 章和第 7 章分别介绍树形结构和图结构；第 8 章和第 9 章分别介绍排序和查找。

本书可作为高职高专院校计算机专业的教材或技术应用型本科院校计算机专业教材。讲授 60 课时左右，除第 1 章外每章可安排 2 课时上机实习。本书还可作为成人教育（面授或函授）教材，另外，也可供广大从事计算机应用的科技人员参考。

本书是文益民、周学毛、李健三位教师在总结多年从事计算机专业“数据结构”课程教学经验的基础上，经过多次反复磋商和共同讨论而定稿的，是三位作者共同合作的产物。全书由文益民整体构思、统稿、修改，博士生导师陈焕文教授对本书进行了详细审阅，并提出了许多宝贵意见。倪芬芬、易新河、李树德参与了本书的排版和制图工作，作者在此一并致以诚挚的谢意。

由于编著者水平有限，书中缺点或错误在所难免，殷切期望广大读者批评指正。

编　者

2008 年 3 月于长沙

目　录

第 1 章 绪论

自 1946 年世界上第一台电子计算机[1]ENIAC 诞生开始，特别是近 20 年来计算机产业的飞速发展与应用已远远超出人们对它的预料。摩尔（Moore）定律[2]指出：微处理器的速度每过 18 个月就会翻一番，而半导体芯片的价格每隔 18 个月就会降低一半。迈特考夫（Metcalfe）定律[3]指出：网络的价值与上网人数的平方成正比。计算机在处理速度、存储容量、网络化，以及软件的精巧化方面经过数十年的发展，以难以想象的方式渗入科学、商业和文化领域中，而智能工程又将令其从量变转向质的飞跃。在科学领域，计算可以模拟气候变化，破解人类基因；在商业领域，低成本的计算、因特网和数字通信正在改变全球经济；在文化领域，计算类产品，如 iPod、YouTube 和计算机动画等无所不在。

计算机软件是计算机系统的灵魂，没有软件的计算机系统无法工作。由于计算机软件的核心是算法，而算法实质上是加工数据的过程，所以研究数据结构对提高编程能力和设计高性能的算法至关重要。本章介绍数据结构和算法分析的基本概念及程序设计基础。

1.1 数据结构的基本概念

1.1.1 数据结构实例

【例 1.1】 学生成绩登记表。

如图 1.1 所示的学生成绩表中的每一行可以当作一个结点（或称为记录），它由学号、姓名、性别、三门课程的成绩、总分等数据项组成。在这个表中，第一个记录没有直接前趋，

---

[1] 第二次世界大战期间，美国军方为了解决计算大量军用数据的难题，成立了由宾夕法尼亚大学的莫奇利和埃克特领导的研究小组，于 1946 年 2 月研制成世界上第一台计算机 ENIAC。我国的第一台计算机 103 机则于 1958 年 8 月由中科院计算所研制成功，该计算机的研制只花了两年左右的时间。

[2] 1965 年，作为英特尔公司的创始人之一，戈登·摩尔应邀撰写了一篇名为“让集成电路填满更多组件”（Cramming more components onto integrated circuits）的文章。文中，摩尔对未来半导体组件工业的发展趋势作出了预测——他指出，单块硅芯片上所集成的晶体管数目大约每年（注：1975 年，摩尔将周期修正为“每两年”）增加一倍。这一预言后来成为广为人知的“摩尔定律”。

[3] 迈特考夫博士（Dr. Metcalfe）于 1973 年发明了与传统的计算机网络实现方法完全不同的局域网络——以太网（Ethernet）。

称为开始结点；最后一个记录没有直接后继，称之为终端结点。除了第一个记录和最后一个记录以外，其余记录，都只有一个直接前趋记录和一个直接后继记录。这种结点之间的关系是“一对一”的关系，这种关系被称为“线性”关系。根据数据之间的这种逻辑关系，我们可以定义给出学生学号查询该同学的成绩或完成在某个同学的后面插入另一个同学的成绩记录等运算。

学号	姓名	性别	操作系统	计算机网络	高等数学	总分
01	丁一	男	90	73	94	257
02	马二	男	67	75	91	233
03	张三	女	87	91	67	245
04	李四	男	90	89	79	258
05	王五	女	97	85	85	267
06	赵六	男	78	93	91	262
……	……	……	……	……	……	……

图 1.1　学生成绩登记表

【例 1.2】　井字棋对弈问题

图 1.2（a）所示是井字棋对弈过程中的一个格局，任何一方只要使相同的 3 个子连成一条直线（可以是一行、一列、或一条对角线）即为胜方。如果下一步由“×”方下，可以派生出 5 个子格局，如图 1.2（b）所示，随后由“○”方接着下，对于每个子格局又可以派生出 4 个子格局……

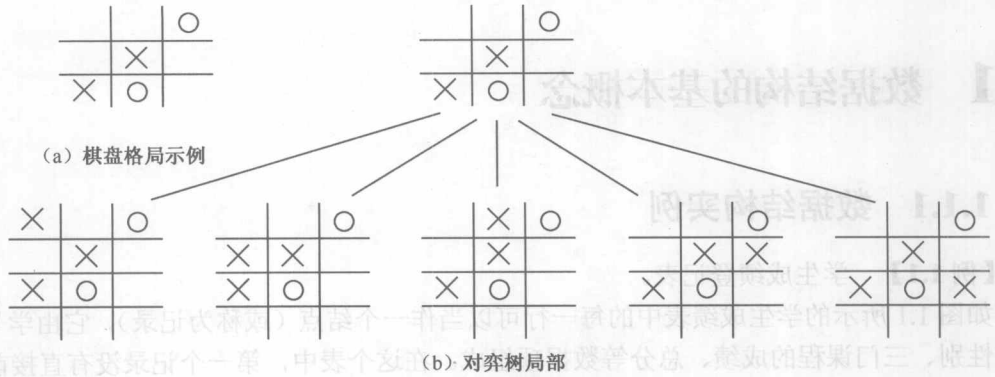


图 1.2　井字棋对弈“树”

若将从对弈开始到结束的过程中所有可能的棋盘格局画在一张图上，即形成一棵倒挂的对弈“树”。“树根”是对弈开始之前的棋盘格局，而所有“叶子”便是可能出现的结局，对弈过程就是从树根沿树枝到达某个叶子的过程。在本例中，对弈中出现的每一个棋盘格局都可以算作一个“结点”。对弈开始之前的棋盘格局没有直接前趋，称为开始结点（根），以后每走一步棋，都有多种应对的棋盘格局，结点之间存在着“一对多”的关系，这种关系被称为“树”关系。根据这种逻辑关系，我们可以求某个棋局有多少个后续棋局或者求某个棋局是从哪个棋局发展而来的等运算。

【例 1.3】　七桥问题

18 世纪，流经哥尼斯堡[1]的普雷格河的河湾处有两个小岛，七座桥连接了两岸和小岛。当地流传一个游戏：要求在一次散步中恰好通过每座桥一次，再回到原出发点。设这 4 块陆地分别为：*A*、*B*、*C*、*D*，可以用图 1.3 表示。

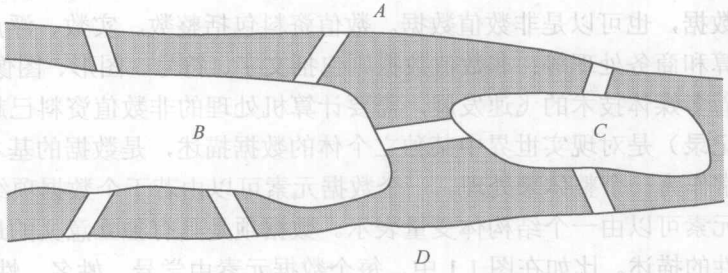


图 1.3　七桥问题

1736 年，瑞士数学家欧拉[2]（Euler）引入图的概念，证明七桥问题永远无解。欧拉以结点表示 4 块陆地，以边表示桥，把图 1.3 抽象为图 1.4，即欧拉回路，上述游戏变成图 1.4 中的图形能不能一笔画成的问题。欧拉对七桥问题的结论是：“所有结点的度（一个结点拥有的边数称为度）均为偶数时，原问题才有解”。我们不想讨论这一数学证明问题，我们感兴趣的是在图 1.4 中，每一个结点都有多个直接前驱和多个直接后继，也就是说，结点与结点之间存在“多对多”关系，这种关系被称为“图”关系。根据这种逻辑关系，我们可以定义与某一结点相邻的结点有哪些，两个结点间有几条通路等运算。

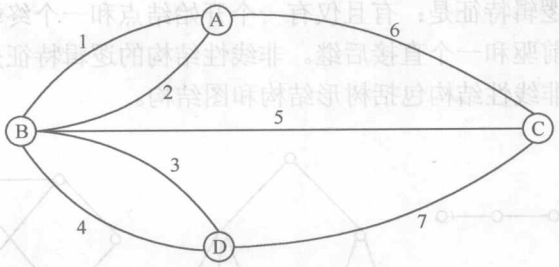


图 1.4　欧拉回路

综上所述，非数值计算问题的数学模型不再是数学方程，而是要研究诸如例 1.1、例 1.2 和例 1.3 所描述的数据之间的这些关系，以及通过数据之间的这些关系可以方便地进行什么运算。因此，简单地说，数据结构是一门研究数据之间的关系以及运算的学科。

---

[1] 今俄罗斯加里宁格勒。

[2] 欧拉（Euler，1707～1783），瑞士数学家及自然科学家。1707 年 4 月 15 日出生于瑞士的巴塞尔，1783 年 9 月 18 日于俄国的彼得堡去世。欧拉是 18 世纪数学界最杰出的人物之一，他不但为数学界做出贡献，更把数学推至几乎整个物理的领域。他是数学史上最多产的数学家，写了大量力学、分析学、几何学的课本，《无穷小分析引论》，《微分学原理》，以及《积分学原理》都成为数学中的经典著作。除了教科书外，欧拉平均以每年 800 页的速度写出创造性论文。他去世后，人们整理出他的研究成果多达 74 卷。

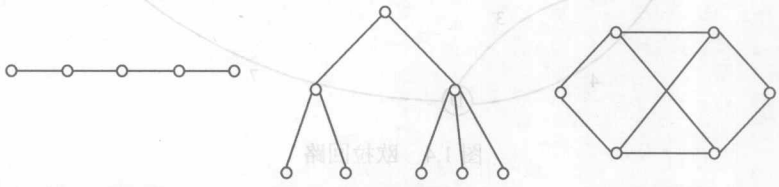
1.1.2 数据结构的概念

数据是信息的载体，是对客观事物的符号表示。通俗的说，凡是能被计算机识别、存取和加工处理的符号、字符、图形、图像、声音、视频信号、程序等一切信息都可以称为数据。数据可以是数值数据，也可以是非数值数据。数值资料包括整数、实数、浮点数或复数等，主要用于科学计算和商务处理等；非数值数据则包括文字、符号、图形、图像、动画、语音、视频信号等。随着多媒体技术的飞速发展，需要计算机处理的非数值资料已越来越多。

数据元素（记录）是对现实世界中某独立个体的数据描述，是数据的基本单位。在计算机中，数据元素常作为一个整体来处理。一个数据元素可以由若干个数据项组成，在 C 程序设计中一个数据元素可以由一个结构体变量表示。数据项是具有独立意义的最小数据单位，是对数据元素属性的描述。比如在图 1.1 中，每个数据元素由学号、姓名、性别、操作系统、计算机网络、高等数学、总分等数据项组成。

数据对象是具有相同性质的数据元素的集合，是数据的一个子集。例如字母字符数据对象是集合 C={'A', 'B', 'C', …, 'Z'}。

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。根据数据元素之间关系的特性。数据结构可由一个二元组(*D*,*S*)定义，其中 *D* 是数据元素的有限集，*S* 是 *D* 中元素的关系的有限集。通常有 3 类基本的数据结构：（1）线性结构。数据元素之间存在“一个对一个”的关系，如例 1.1。（2）树形结构。数据元素之间存在“一个对多个”的关系，如例 1.2。（3）图结构或网结构。数据元素之间存在“多个对多个”的关系，如例 1.3。图 1.5 所示为这三种数据结构。上面的三种数据结构可以划分为两大类：线性结构和非线性结构。线性结构的逻辑特征是：有且仅有一个开始结点和一个终端结点，并且所有的结点都最多只有一个直接前驱和一个直接后继。非线性结构的逻辑特征是一个结点可能有多个直接前驱或直接后继。非线性结构包括树形结构和图结构。



（a）线性结构　　（b）树形结构　　（c）图结构

图 1.5　三类基本数据结构的示意图

逻辑结构描述数据元素之间的逻辑关系。通常说的线性结构、树形结构和图结构就是指数据元素的逻辑结构。

物理结构是数据结构在计算机中的表示，又称为**存储结构**。具有某种逻辑结构的数据结构在计算机中有两种不同的表示方法：顺序映像和非顺序映像，由此得到两种不同的存储结构：顺序结构和链式结构（见图 1.7）。顺序结构的特点是借助数据元素在存储器中的相对位置来表示数据元素之间的逻辑关系。链式结构的特点是借助指示数据元素存储地址的“指针”来表示数据元素之间的逻辑关系。假如要存储字母表 *L*={*'A'*, *'B'*, *'C'*, …, *'Z'*}，它的存储结构

可以采取两种形式，这两种存储结构可以用图 1.6 来描述。数据的逻辑结构和物理结构是密切相连的两个方面，依据数据的逻辑结构可以设计合理的算法，而算法的实现要依据数据的存储结构。

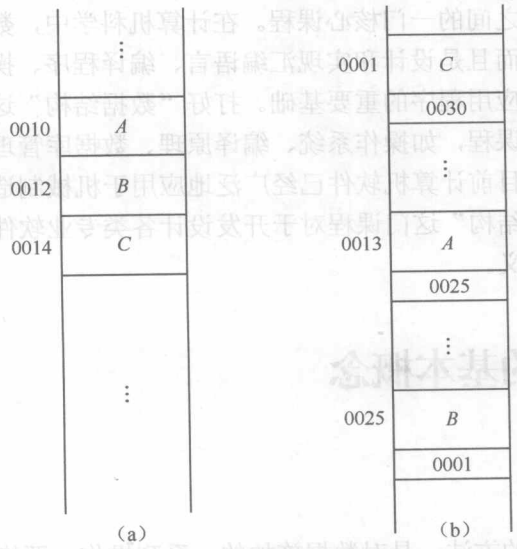


图 1.6　顺序存储和链式存储结构的示意图

1.1.3　学习数据结构的理由

数据结构作为一门独立的学科始于 1968 年，但在此之前有关内容已散见于编译原理及操作系统的教材之中。1968 年，美国的图灵（Turing）奖[1]获得者唐•欧•克努特（Donald Ervin Knuth）教授[2]开创了数据结构的最初体系，他所著作的《计算机程序设计艺术》第一卷《基本算法》是第一本比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的主要内容。人们越来越重视数据结构，著名的瑞士计算机科学家沃斯（N.Wirth）教授[3]指出：算法+数据结构=程序。从 20 世纪 70 年代中期到 80 年代初，各种版本的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

[1] 图灵奖由美国计算机学会（ACM）于 1966 年创立，以计算机概念的创始人图灵的名字命名，每年评选出 1～3 名获奖者，是世界计算机科学领域的最高奖项，有该领域的诺贝尔奖之称。

[2] 唐•欧•克努特是斯坦福大学教授，1974 年度图灵奖获得者，所著《计算机程序设计艺术》被誉为“计算机的圣经”。他还发明了计算机排版系统 TEX 和 METAFONT，他因这些成就和大量创造性的影响深远的论著而誉满全球。作为斯坦福大学计算机程序设计艺术的荣誉退休教授，他投入全部的时间来完成其关于计算机科学的史诗性的七卷集。克努特教授获得了许多奖项和荣誉，包括美国计算机协会图灵奖（ACM Turing Award），美国前总统卡特授予的科学金奖（Medal of Science），美国数学学会斯蒂尔奖（AMS Steele Prize），以及受尊重的京都奖（Kyoto Prize）。

[3] 沃斯：PASCAL 之父及结构化程序设计的首创者，1984 年图灵奖获得者。

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，越来越被人们所重视。

从我国计算机教学现状来看，“数据结构”不仅仅是计算机专业教学计划中的核心课程之一，而且已逐步成为非计算机专业的主要选修课程之一。数据结构又是一门介于数学、计算机硬件和计算机软件三者之间的一门核心课程。在计算机科学中，数据结构不仅是一般非数值计算程序设计的基础，而且是设计和实现汇编语言、编译程序、操作系统、数据库系统，以及其他系统程序和大型应用程序的重要基础。打好“数据结构”这门课程的扎实基础，对于学习计算机专业的其他课程，如操作系统、编译原理、数据库管理系统、软件工程、人工智能等都是十分有益的。目前计算机软件已经广泛地应用于机械制造、经济管理、人文艺术等各个行业，学好“数据结构”这门课程对于开发设计各类专业软件从而提高每个行业的工作效率具有非常重要的意义。

1.2 算法分析的基本概念

1.2.1 算法

算法是解决特定问题的方法，是对数据施加的一系列操作。严格地说：算法是由若干指令组成的有限序列，而程序是算法的实现。根据实际的需要，一个实际问题的解决可以有多种算法。算法包括数值算法和非数值算法两种。解决数值问题的算法叫做数值算法，如求解线性方程组，求解代数方程，求解微分方程等。解决非数值问题的算法叫做非数值算法，数据处理方面的算法都属于非数值算法，如各种排序算法、查找算法、插入算法、删除算法、遍历算法等。

一个算法必须具有以下 5 个特性。

（1）有穷性。一个算法包括的指令数必须有限，每一条指令被执行的次数必须有限。

（2）确定性。算法的每一条指令必须有确切的含义，无二义性。

（3）可行性。算法中的每一指令都可以通过有限次、可实现的基本运算且在有限的时间内来实现。

（4）输入。一个算法具有零个或多个输入。

（5）输出。一个算法具有一个或多个输出。

算法设计的要求如下。

（1）正确性。算法的执行结果应当满足预先设定的功能和要求。在实际应用中算法的“正确性”有多个层次的含义。

（2）可读性。一个算法应当思路清晰、层次分明、易读易懂，有利于人对算法的理解。

（3）健壮性。当输入非法数据时，应能作适当反应和处理，不至于引起莫名其妙的后果。

（4）高效性与低存储量。对同一个问题，执行时间越短，算法的效率就越高；完成相同的功能，执行算法时所占用的存储空间应尽可能少。

实际上，一个算法很难做到十全十美，原因是上述要求有时会相互抵触。比如，时间和空间就是一对矛盾。要节约算法的执行时间，往往以牺牲一定存储空间为代价；而为了节省

存储空间就可能耗费更多的计算时间。所以，实际操作中应以算法正确性为前提，根据具体情况而有所侧重。若一个程序使用次数较少，一般要求简明易懂即可；对于需要反复多次使用的程序，应尽可能选用快速的算法；若解决问题的数据量极大，而机器的存储空间又相对较小，则主要考虑如何节约存储空间。

1.2.2　算法效率的分析

算法执行时间需要根据该算法编制的程序在计算机上的执行时间来确定。一个算法所耗费的时间等于算法中每条语句执行的时间之和。每一条语句执行的时间是该语句的执行次数（**频度**，Frequency count）与该语句执行一次所需时间的乘积，而每条语句的执行时间取决于其对应机器指令的执行时间。一个使用高级程序语言编写的程序在计算机上运行所需的时间取决于多种因素：

（1）使用何种程序设计语言。实现编程的语言的级别越高，其执行效率就越低。

（2）选用何种策略的算法。

（3）算法涉及问题的规模（求解问题的输入量，通常用 n 表示）。例如，求 50 以内的素数与求 1 000 以内的素数其执行时间必然是不同的。

（4）编译程序所生成目标代码的质量。对于代码优化较好的编译程序其所生成的程序质量较高。

（5）机器执行指令的速度。

（6）计算机的体系结构。并行计算通常能缩短算法的计算时间。

显然，在各种因素不确定的情况下，使用执行算法的绝对时间来衡量算法的效率是不合适的。在上述各种与计算机相关的软、硬件因素确定以后，那么一个特定算法的运行时间的长短就只依赖于问题的规模（通常用正整数 n 表示）。

1.2.3　算法效率的评价

算法的效率通常用时间复杂度与空间复杂度来评价，它们反映了算法的执行所需的时间和空间与问题规模之间一种数量上的依赖关系。

1. 时间复杂度（Time complexity）

通常使用一个算法中所有语句的频度之和 $f(n)$来表示该算法所需的时间。在假定执行一条语句需要时间固定的情况下，语句的数量能大致表示算法运行所需的时间，因此 $f(n)$粗粒度地描述了一个算法所需的时间。假设当问题规模 n 趋向于无穷大时有下式成立：

$$T(n)=O(f(n)) \tag{1.1}$$ [1]

它表示随着问题规模的扩大，$T(n)$的增长率和 $f(n)$的增长率相同，把 $T(n)$称作算法的**渐近时间复杂度**（Asymptotic Time Complexity），简称**时间复杂度**。

---

[1] “O”的数学定义为：若 $f(n)$为正整数 n 的函数，则 $x_n=O(f(n))$表示存在一个正常数 M，使得 $n\geqslant n_0$ 时满足$|x_n|\leqslant M|f(x)|$。

下面，我们来看几个例子：

【例 1.4】　交换 A 和 B 的内容。

（1）Temp=A;

（2）A=B;

（3）B=Temp。

以上三条语句的频度都为 1，$f(n)=3$。所以该程序的执行时间与问题规模无关 $f(n)=3$。算法的时间复杂度为常数阶，记为 $T(n)=O(1)$。

【例 1.5】　累加。

```
（1）x=0;                          （1）执行 1 次;
（2）y=0;                          （2）执行 1 次;
     for (k=1; k<=n; k++)
（3）    x++;                      （3）要执行 n 次;
     for (i=1; i<=n; i++)
         for (j=1; j<=n;j++)
（4）        y++;                  （4）要执行 n² 次;
```

所有语句频度之和 $f(n)$ 为：n^2+n+2。

当 $n\to\infty$ 时，显然有：

$$\lim_{n\to\infty} f(n)n^2 = \lim_{n\to\infty}(n^2+n+2)/n^2 = 1$$

∴　$T(n)=O(n^2)$

将常见的时间复杂度按照数量级递增排列（见图 1.7），则依次为：常数阶 $O(1)$、对数阶 $O(\lg n)$、线性阶 $O(n)$、线性对数阶 $O(n\lg n)$、平方阶 $O(n^2)$、立方阶 $O(n^3)$、指数阶 $O(2^n)$。数量级越高说明算法所需的时间随问题规模的增大而以更高的速度增加。一个算法的时间复杂度最好是常数阶，最高不超过 k 次方阶。如果算法的时间复杂度为指数阶，则该算法无法使用，因为时间会随问题规模的增长而以指数级增长。

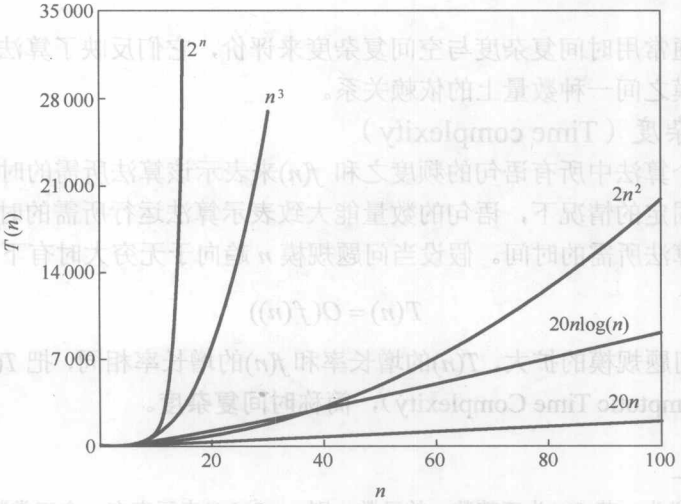


图 1.7　常见函数的增长率

2. 空间复杂度

一个程序的**空间复杂度**（Space complexity）是指运行完一个程序所需要内存的大小。利用程序的空间复杂度，可以对程序的运行所需要的内存多少有个预先估计。一个程序执行时除了需要存储空间来存储本身所使用的指令、常数、变量和输入数据外，还需要一些对数据进行操作的工作单元和存储一些为实现计算所需信息的辅助空间。程序执行时所需的存储空间包括以下两部分。

（1）固定部分。这部分空间的大小与输入/输出数据的个数多少、数值大小无关。主要包括指令空间（即代码空间）、数据空间（常量、简单变量）等所占的空间。这部分属于静态空间。

（2）可变部分。这部分空间主要包括动态分配的空间，以及递归栈所需的空间等。这部分空间大小与算法有关。

一个算法所需的存储空间用 $f(n)$ 表示。

$$S(n)=O(f(n)) \tag{1.2}$$

其中 n 为问题的规模，$S(n)$ 表示算法的空间复杂度。

【例 1.6】 试运行下面的 C 语言程序。将数组 a 的大小调整可以看到 f 执行的次数与 a 的大小有关。

```
#include "stdio.h"
void f()
{
    double a[10000];
    printf("The function f is called!\n");
    f();
}
main()
{
    f();
}
```

在进行时间复杂度和空间复杂度分析时，如果所需的时间和所占空间量依赖于特定的输入，一般都按最坏情况来分析。

算法设计有一条重要的原则：空间/时间权衡原则。即牺牲空间或者其他替代资源，通常可以减少时间代价。反过来有：付出时间代价可以减少对空间需求。

1.3 程序设计基础

1.3.1 软件工程的基本概念

软件产业已逐步从一个弱小的产业成长为新兴的、发展最快的和潜力巨大的产业。它代表着一个国家高新技术的水平。计算机系统由软件和硬件组成。通常所说的软件是相对于硬件而言的，它包括计算机运行时所需要的各种程序，一般分为系统软件和应用软件。随着计算机应用日益普及和深化，计算机软件的数量以惊人的速度急剧膨胀。而且现代软件的规模