

高 等 学 校 教 材

数据结构与算法

—— C++语言描述

陈慧南



高等教育出版社

高等学校教材

数据结构与算法

——C++ 语言描述

陈慧南

高等教育出版社

内容提要

本书根据作者多年在南京邮电学院讲授“数据结构”和“算法设计与分析”课程的教学经验,在编写用 Pascal、C 和 C++ 语言描述的几本数据结构教材基础上,参考近几年国内外多种优秀教材编写而成。

本书涵盖了“数据结构与算法”的核心知识单元,使用 C++ 语言描述。书中不仅系统介绍了各种传统的数据结构和搜索、排序算法,还引入了比较高级的数据结构,如伸展树和跳表。本书讨论算法分析和算法设计策略,讨论搜索和排序算法的时间下界,还介绍了随机算法以及 NP 难度和 NP 完全问题。

全书条理清晰,内容翔实。书中算法都有完整的 C++ 程序,程序结构清晰,构思精巧,既是读者学习数据结构与算法的很好示例,也是很好的 C++ 程序设计示例。本书深入浅出,配有大量的实例和图示,并有丰富的习题,适于自学。

本书是一本数据结构与算法知识合二为一的教材,且易于取舍和重组,因此可作为高等院校计算机专业或其他相关专业的“数据结构”或“数据结构与算法”课程的教材,也可供学习该领域知识的人员参考。

图书在版编目(CIP)数据

数据结构与算法: C++ 语言描述/陈慧南. —北京: 高等教育出版社, 2005.1
ISBN 7-04-015876-0

I. 数... II. 陈... III. ①数据结构—高等学校—教材②算法分析—高等学校—教材③C 语言—程序设计—高等学校—教材 IV. TP311.12

中国版本图书馆 CIP 数据核字 (2004) 第 143349 号

策划编辑 倪文慧 责任编辑 武林晓 市场策划 陈 振
封面设计 刘晓翔 责任印制 孔 源

出版发行 高等教育出版社
社 址 北京市西城区德外大街 4 号
邮政编码 100011
总 机 010-58581000
经 销 北京蓝色畅想图书发行有限公司
印 刷 北京星月印刷厂

购书热线 010-58581118
免费咨询 800-810-0598
网 址 <http://www.hep.edu.cn>
<http://www.hep.com.cn>
网上订购 <http://www.landaco.com>
<http://www.landaco.com.cn>

开 本 787×1092 1/16
印 张 29.5
字 数 620 000

版 次 2005 年 1 月第 1 版
印 次 2005 年 1 月第 1 次印刷
定 价 33.00 元

本书如有缺页、倒页、脱页等质量问题,请到所购图书销售部门联系调换。

版权所有 侵权必究

物料号: 15876-00

前 言

美国计算机协会(ACM)和美国电气和电子工程师学会计算机分会(IEEE-CS)的**计算学科教学计划 1991**(Computing Curricula 1991, CC1991)将**算法与数据结构**定义为计算学科研究的 9 个主领域之一。**CC2001** 报告调整了计算学科研究领域的划分,最终将计算学科划分为 14 个主领域,其中数据结构和算法的知识主要包含在**程序设计基础(PF)**、**程序设计语言(PL)**以及**算法与复杂性(AL)**等领域中。这些领域中的多数知识单元被规定为计算机及相关学科本科学生必须掌握的核心单元。如**基本数据结构(PF3)**,包括数组、字符串、堆栈、队列、树、图和散列表等,**递归技术(PF4)**,**数据类型和数据抽象(PL4 和 PL9)**,**面向对象的程序设计(PL6)**,**算法分析的基本方法(AL1)**,**基本计算算法(AL3)**,包括查找、排序、哈希表算法、搜索树和图算法,以及常用的**算法设计策略(AL2)**都是核心单元。

计算机学科的高速发展引起学科教学知识体系的迅速变化,计算机学科的课程教学必须随之调整和更新,舍去已失去活力的内容,引入新理论、新方法和新技术,才能顺应学科发展的需求。不少高等院校参考和借鉴 CC2001,对计算机专业本科生教学计划做了相应调整。

“数据结构与算法”不仅是计算机科学与技术专业的一门核心课程,也是电子信息等其他理工科专业学生必备的基础知识。掌握该领域知识有助于理工科学生进一步学习和掌握计算机应用技术,适应更广泛的职业挑战。

目前,我国多数高等院校将“数据结构”作为计算机专业的必修课,而将“算法设计与分析”列为本科本专业的选修课。但也有许多高等院校以“数据结构与算法”的名称设课。

按照 CC2001 报告,应将算法与复杂性领域中更多的核心单元包括在本科教学计划的基础课程中。CC2001 建议的关于数据结构与算法的基础课程,涵盖了基本数据结构、基本计算算法、算法设计策略和算法分析的基本内容。

本书包括三部分内容。第一部分为基础知识,包括概论、算法分析的基本方法、数据抽象和面向对象程序设计、程序证明和递归技术;第二部分包括传统数据结构的内容;第三部分介绍常见的算法设计策略、随机算法、NP 难度和 NP 完全问题。

虽然传统数据结构和算法设计策略知识在本教材中分属第二、三两部分,但在书中并没有截然分离。分治策略的基本思想在讨论树、二叉树、对半搜索、快速排序和合并排序算法时被引入;在讨论最小代价生成树和单源最短路径算法的同时介绍了贪心策略。这样做,对于算法设计策略和数据结构算法两者的学习都是有益的。但有关算法设计与分析的更深入的内容,安排在第三部分讲授。

所以,这是一本将数据结构和算法知识有机结合的教材,且易于取舍和重组。因此,本书可作为高等院校计算机专业和其他相关专业的本科或研究生“数据结构”或“数据结构与算法”课程的教材,也可供学习该领域知识的人员参考。

作为“数据结构”教材,本书在内容上注意兼顾广度和深度,不仅系统介绍了各种传统的数据结构和各种搜索、内外排序方法,还引入了一些比较高级的数据结构,如伸展树和跳表,并强化对数据结构算法的时间和空间分析,包括搜索和排序时间的下界分析。作为“数据结构与算法”或“算法与数据结构”教材,它涵盖了该课程必需的核心单元,加强了算法分析,并包含算法设计策略、随机算法以及 NP 难度和 NP 完全问题等可选知识。希望上述考虑能够兼顾课程教学的现状和发展,满足多层次多学科专业对计算机软件知识的需求。

本书采用面向对象方法讨论和设计数据结构,使用 C++ 语言描述算法。学生通过学习,可以掌握数据抽象原理和面向对象方法。

本书既注重数据结构和算法原理,又十分强调软件工程和程序设计训练。书中算法都有完整的 C++ 程序,程序结构清晰,构思精巧。所有程序都在 VC++ 环境下编译通过并能正确运行。它们既是学习数据结构和算法的很好示例,也是很好的 C++ 程序设计示例。

全书条理清晰,内容翔实,深入浅出,书中对算法都做了较详细的解释,尽可能做到可读易懂,并配有大量的实例和图示,有利于读者理解算法的实质和编程思想。每章结尾处的小结帮助读者了解一章的要点,并有丰富的习题,适于自学。读者可以通过南京邮电学院主页 <http://www.njupt.edu.cn>, 点击“教育网站”,链接至作者主持的数据结构与算法的 Web 站点,获取有关教学材料。

本书被列为高等教育出版社“高等教育百门精品课程教材建设计划”的选题计划之一,得到了高等教育出版社的大力支持,并得到了南京邮电学院和计算机科学与技术系领导的推荐和关心,在此表示衷心感谢。

书中若有不当之处,敬请读者批评指正。

作 者

2005 年 1 月于南京

郑 重 声 明

高等教育出版社依法对本书享有专有出版权。任何未经许可的复制、销售行为均违反《中华人民共和国著作权法》，其行为人将承担相应的民事责任和行政责任，构成犯罪的，将被依法追究刑事责任。为了维护市场秩序，保护读者的合法权益，避免读者误用盗版书造成不良后果，我社将配合行政执法部门和司法机关对违法犯罪的单位和个人给予严厉打击。社会各界人士如发现上述侵权行为，希望及时举报，本社将奖励举报有功人员。

反盗版举报电话：(010) 58581897/58581896/58581879

传 真：(010) 82086060

E - mail：dd@hep.com.cn

通信地址：北京市西城区德外大街 4 号

高等教育出版社打击盗版办公室

邮 编：100011

购书请拨打电话：(010)58581118

目 录

第一部分 基础知识

第1章 概论	(3)	习题	(18)
1.1 算法与数据结构	(3)	第2章 算法基础	(20)
1.1.1 算法	(3)	2.1 算法复杂度	(20)
1.1.2 数据结构	(5)	2.1.1 什么是好的算法	(20)
1.1.3 数据的逻辑结构	(6)	2.1.2 影响程序运行时间的因素	(21)
1.1.4 数据的存储表示	(8)	2.1.3 算法的时间复杂度	(21)
1.1.5 数据结构的运算	(9)	2.1.4 使用程序步分析算法	(23)
1.2 数据抽象和抽象数据类型	(10)	2.1.5 算法的空间复杂度	(24)
1.2.1 抽象、数据抽象和过程抽象	(10)	2.2 渐近表示法	(25)
1.2.2 封装与信息隐蔽	(10)	2.2.1 大 O 记号	(25)
1.2.3 数据类型和抽象数据类型	(11)	2.2.2 Ω 记号	(27)
1.2.4 数据结构与抽象数据类型	(12)	2.2.3 Θ 记号	(28)
1.3 面向对象方法	(12)	2.2.4 小 o 记号	(28)
1.3.1 面向对象方法的由来	(12)	2.2.5 算法按时间复杂度分类	(28)
1.3.2 面向对象方法的基本思想	(12)	2.3 递归、归纳和递推	(30)
1.3.3 面向对象方法的要素	(13)	2.3.1 递归	(30)
1.3.4 面向对象方法和抽象数据类型	(14)	2.3.2 递归算法示例	(32)
1.4 描述数据结构和算法	(15)	2.3.3 证明方法	(34)
1.4.1 数据结构的规范	(15)	2.3.4 递推关系	(35)
1.4.2 实现数据结构	(16)	本章小结	(38)
本章小结	(17)	习题	(38)

第二部分 数据结构

第3章 数组和链表	(43)	3.2.2 二维数组	(47)
3.1 结构和类	(43)	3.2.3 多维数组	(48)
3.1.1 结构	(43)	3.3 链表	(49)
3.1.2 结构表示元素	(44)	3.3.1 指针	(49)
3.2 数组	(46)	3.3.2 单链表	(53)
3.2.1 一维数组	(47)	3.3.3 带表头结点的单链表	(56)

3.3.4 单循环链表	(57)	5.2.3 项结点类	(105)
3.3.5 双向链表	(57)	5.2.4 多项式类	(106)
3.4 采用模拟指针的链表	(59)	5.2.5 多项式的输入和输出	(107)
3.4.1 结点结构	(59)	5.2.6 多项式相加	(108)
3.4.2 可用空间表	(59)	5.2.7 重载运算符函数	(109)
3.5 异常处理	(62)	5.3 数组作为抽象数据类型	(110)
本章小结	(63)	5.3.1 数组 ADT	(111)
习题	(63)	5.3.2 一维数组的 C++ 类	(112)
第 4 章 堆栈和队列	(65)	5.4 特殊矩阵	(113)
4.1 堆栈	(65)	5.4.1 对称矩阵	(113)
4.1.1 堆栈 ADT	(65)	5.4.2 带状矩阵	(114)
4.1.2 堆栈的顺序表示	(66)	5.5 稀疏矩阵	(115)
4.1.3 堆栈的链接表示	(69)	5.5.1 稀疏矩阵 ADT	(115)
4.2 队列	(72)	5.5.2 稀疏矩阵的顺序表示	(116)
4.2.1 队列 ADT	(72)	5.5.3 稀疏矩阵转置	(118)
4.2.2 队列的顺序表示	(73)	5.5.4 稀疏矩阵的正交链表表示	(120)
4.2.3 队列的链接表示	(76)	5.5.5 建立正交链表	(124)
4.3 表达式计算	(77)	5.5.6 打印正交链表	(125)
4.3.1 表达式	(77)	本章小结	(126)
4.3.2 中缀表达式转换为后缀表 达式	(78)	习题	(126)
4.3.3 计算后缀表达式的值	(81)	第 6 章 字符串和广义表	(128)
4.4 实现递归	(85)	6.1 字符串	(128)
4.4.1 子程序调用和系统栈	(85)	6.1.1 字符串 ADT	(128)
4.4.2 递归函数的性能	(87)	6.1.2 字符串的存储表示	(129)
4.4.3 尾递归	(87)	6.1.3 串运算的实现	(130)
4.5 演示与测试	(88)	6.1.4 简单模式匹配算法	(131)
本章小结	(92)	6.1.5 模式匹配的 KMP 算法	(134)
习题	(92)	6.2 广义表	(139)
第 5 章 线性表和数组 ADT	(94)	6.2.1 广义表的概念	(139)
5.1 线性表	(94)	6.2.2 广义表 ADT	(140)
5.1.1 线性表 ADT	(94)	6.2.3 广义表的存储表示	(140)
5.1.2 线性表的顺序表示	(96)	6.2.4 广义表算法	(142)
5.1.3 线性表的链接表示	(99)	本章小结	(142)
5.1.4 两种存储表示的比较	(103)	习题	(143)
5.2 多项式的算术运算	(103)	第 7 章 树	(144)
5.2.1 多项式 ADT	(103)	7.1 树的基本概念	(144)
5.2.2 多项式的链接表示	(104)	7.1.1 树的定义	(144)
		7.1.2 基本术语	(145)

7.2 二叉树····· (146)	7.9.5 改进的函数 Union 和 Find ··· (187)
7.2.1 二叉树的定义····· (147)	7.9.6 按等价关系分组····· (188)
7.2.2 二叉树的性质····· (148)	本章小结····· (189)
7.2.3 二叉树 ADT ····· (149)	习题····· (189)
7.2.4 二叉树的存储表示····· (150)	第 8 章 集合和搜索 ····· (192)
7.2.5 二叉树类····· (151)	8.1 集合和搜索····· (192)
7.2.6 二叉树基本运算····· (153)	8.1.1 集合和搜索的概念····· (192)
7.3 二叉树的遍历····· (155)	8.1.2 动态集 ADT ····· (194)
7.3.1 二叉树遍历算法····· (155)	8.1.3 集合的表示····· (195)
7.3.2 二叉树遍历的递归算法····· (157)	8.2 顺序搜索····· (196)
7.3.3 二叉树遍历的应用实例····· (158)	8.2.1 无序表的顺序搜索····· (196)
7.4 二叉树遍历的非递归算法····· (159)	8.2.2 有序表的顺序搜索····· (196)
7.4.1 遍历器类····· (159)	8.2.3 平均搜索长度····· (197)
7.4.2 中序遍历器类····· (161)	8.2.4 自组织表····· (198)
7.5 二叉线索树····· (163)	8.3 二分搜索····· (198)
7.5.1 二叉线索树的定义····· (163)	8.3.1 分治法和二分搜索····· (198)
7.5.2 构造中序线索树····· (164)	8.3.2 对半搜索····· (200)
7.5.3 遍历二叉线索树····· (165)	8.3.3 二叉判定树····· (202)
7.6 树和森林····· (167)	8.3.4 斐波那契搜索····· (204)
7.6.1 森林与二叉树的转换····· (167)	8.4 搜索算法的时间下界····· (207)
7.6.2 树和森林的存储表示····· (168)	本章小结····· (208)
7.6.3 树和森林的遍历····· (171)	习题····· (208)
7.7 堆和优先权队列····· (172)	第 9 章 动态集和搜索树 ····· (209)
7.7.1 堆····· (172)	9.1 二叉搜索树····· (209)
7.7.2 优先权队列 ADT ····· (175)	9.1.1 二叉搜索树的定义····· (209)
7.7.3 优先权队列类····· (176)	9.1.2 二叉搜索树的搜索····· (210)
7.7.4 实现优先权队列····· (176)	9.1.3 二叉搜索树的插入····· (211)
7.8 哈夫曼树和哈夫曼编码····· (179)	9.1.4 二叉搜索树的删除····· (213)
7.8.1 树的路径长度····· (179)	9.1.5 平均情况时间分析····· (215)
7.8.2 哈夫曼树和哈夫曼算法····· (180)	9.2 二叉平衡树····· (217)
7.8.3 哈夫曼树类····· (181)	9.2.1 二叉平衡树的定义····· (217)
7.8.4 构造哈夫曼树····· (182)	9.2.2 二叉平衡树类····· (217)
7.8.5 哈夫曼编码····· (183)	9.2.3 二叉平衡树的平衡旋转····· (218)
7.9 并查集和等价关系····· (184)	9.2.4 二叉平衡树的插入····· (225)
7.9.1 并查集 ADT ····· (184)	9.2.5 二叉平衡树的删除····· (227)
7.9.2 并查集的存储表示····· (185)	9.2.6 二叉平衡树的高度····· (230)
7.9.3 并查集类····· (186)	9.3 B-树····· (231)
7.9.4 函数 Union 和 Find ····· (186)	9.3.1 m 叉搜索树····· (232)

9.3.2 B-树的定义	(234)	11.2.2 图的邻接矩阵实现	(276)
9.3.3 B-树的高度	(234)	11.2.3 图的邻接表表示法	(278)
9.3.4 B-树的搜索	(235)	11.2.4 图的邻接表实现	(279)
9.3.5 B-树的插入	(235)	11.3 图的遍历	(282)
9.3.6 B-树的删除	(238)	11.3.1 扩充的图类	(282)
9.4 键树	(240)	11.3.2 深度优先遍历	(283)
9.4.1 键树的定义	(240)	11.3.3 宽度优先遍历	(285)
9.4.2 双链树	(241)	11.3.4 基本遍历方法	(286)
9.4.3 Trie 树	(242)	11.4 拓扑排序	(287)
9.5 伸展树	(243)	11.4.1 用顶点代表活动的 AOV 网	(288)
9.5.1 伸展树的伸展操作	(243)	11.4.2 拓扑排序算法	(289)
9.5.2 性能分析	(245)	11.4.3 拓扑排序 C++ 程序	(290)
本章小结	(247)	11.5 关键路径	(292)
习题	(248)	11.5.1 用边代表活动的 AOE 网	(292)
第 10 章 跳表和散列表	(250)	11.5.2 关键路径算法	(293)
10.1 字典	(250)	11.5.3 关键路径 C++ 程序	(295)
10.2 跳表	(250)	11.6 最小代价生成树	(296)
10.2.1 什么是跳表	(251)	11.6.1 基本概念	(296)
10.2.2 跳表类	(253)	11.6.2 普里姆算法	(297)
10.2.3 跳表的搜索	(255)	11.6.3 克鲁斯卡尔算法	(299)
10.2.4 跳表的插入	(255)	11.6.4 最优化问题和贪心法	(302)
10.2.5 跳表的删除	(257)	11.6.5 贪心法求解最小代价 生成树	(302)
10.3 散列表	(258)	11.7 最短路径	(304)
10.3.1 散列技术	(258)	11.7.1 贪心法和单源最短路径	(304)
10.3.2 散列函数	(259)	11.7.2 迪杰斯特拉算法	(305)
10.3.3 拉链法	(261)	11.7.3 所有顶点之间的最短路径	(309)
10.3.4 开地址法	(262)	本章小结	(311)
10.3.5 线性探查法	(262)	习题	(311)
10.3.6 其他开地址法	(266)	第 12 章 内排序	(314)
10.3.7 性能分析	(268)	12.1 基本概念	(314)
本章小结	(269)	12.2 简单排序算法	(315)
习题	(269)	12.2.1 直接插入排序	(315)
第 11 章 图	(270)	12.2.2 简单选择排序	(318)
11.1 图的基本概念	(270)	12.2.3 冒泡排序	(319)
11.1.1 图的定义与术语	(270)	12.3 快速排序	(321)
11.1.2 图 ADT	(273)	12.3.1 分治法与快速排序	(321)
11.2 图的存储结构	(274)	12.3.2 快速排序算法	(322)
11.2.1 图的矩阵表示法	(274)		

12.3.3 性能分析	(324)	第13章 文件和外排序	(340)
12.4 两路合并排序	(326)	13.1 辅助存储器简介	(340)
12.4.1 合并两个有序序列	(326)	13.1.1 主存储器和辅助存储器	(340)
12.4.2 分治法和两路合并排序	(327)	13.1.2 磁盘存储器	(340)
12.4.3 合并排序算法	(328)	13.2 文件	(342)
12.4.4 性能分析	(329)	13.2.1 文件的基本概念	(342)
12.5 堆排序	(329)	13.2.2 文件的组织方式	(342)
12.5.1 堆排序算法	(329)	13.3 文件的索引结构	(345)
12.5.2 实现堆排序	(330)	13.3.1 静态索引结构	(345)
12.5.3 性能分析	(331)	13.3.2 动态索引结构	(346)
12.6 排序算法的时间下界	(331)	13.4 外排序	(347)
12.7 基数排序	(333)	13.4.1 外排序的基本过程	(347)
12.7.1 分配排序	(333)	13.4.2 初始游程的生成	(348)
12.7.2 基数排序算法	(334)	13.4.3 多路合并	(352)
12.7.3 实现基数排序	(335)	13.4.4 最佳合并树	(354)
本章小结	(337)	本章小结	(355)
习题	(338)	习题	(356)

第三部分 算法设计与分析

第14章 问题求解和算法设计	(359)	14.4.2 最优二叉搜索树	(382)
14.1 问题求解方法	(359)	14.5 回溯法	(387)
14.1.1 问题和问题求解	(359)	14.5.1 一般方法	(387)
14.1.2 问题求解过程	(360)	14.5.2 n -皇后问题	(391)
14.1.3 系统生命周期	(360)	14.5.3 子集和数问题	(395)
14.1.4 问题求解策略	(361)	14.6 分支限界法	(399)
14.2 分治法	(361)	14.6.1 一般方法	(399)
14.2.1 一般方法	(361)	14.6.2 基于上下界的分支限界法	(401)
14.2.2 递推关系	(362)	14.6.3 0/1 背包问题	(403)
14.2.3 求最大、最小元	(363)	14.6.4 旅行商问题	(408)
14.2.4 选择问题	(366)	14.7 随机算法	(413)
14.2.5 斯特拉森矩阵相乘	(369)	14.7.1 基本概念	(413)
14.3 贪心法	(371)	14.7.2 拉斯维加斯算法	(414)
14.3.1 一般方法	(371)	14.7.3 蒙特卡罗算法	(416)
14.3.2 最佳合并模式	(372)	14.7.4 舍伍德算法	(418)
14.3.3 背包问题	(375)	本章小结	(419)
14.3.4 贪心法的基本要素	(378)	习题	(419)
14.4 动态规划法	(379)	第15章 NP 难度和 NP 完全问题	(421)
14.4.1 动态规划法的基本要素	(379)	15.1 基本概念	(421)

15.1.1 不确定算法和可满足性 问题	(422)	A.3 实习要求	(435)
15.1.2 P类和NP类问题	(425)	A.4 实习步骤	(435)
15.1.3 NP难度和NP完全问题	(426)	A.5 实习报告	(436)
15.1.4 Cook定理	(426)	A.6 实习题	(436)
15.2 一些典型的NP完全问题	(426)	附录B C++程序设计概要	(438)
15.2.1 最大集团判定问题	(427)	B.1 函数与参数传递	(439)
15.2.2 顶点覆盖判定问题	(428)	B.2 动态存储分配	(442)
15.2.3 3元CNF-可满足性	(429)	B.3 类与对象	(443)
15.2.4 图着色数判定问题	(430)	B.4 函数和操作符重载	(444)
本章小结	(431)	B.5 继承性和派生类	(445)
习题	(432)	B.6 多态性、虚函数和动态联编	(445)
附录	(433)	B.7 纯虚函数和抽象类	(447)
附录A 实习要求和实习题	(433)	B.8 模板函数和模板类	(448)
A.1 面向对象方法概述	(433)	B.9 友元函数和友元类	(450)
A.2 实习目的	(435)	附录C 专有名词中英文对照表	(452)
		参考文献	(460)

第一部分

基础知识

第 1 章 概 论

在计算机科学中,算法和数据结构知识是至关重要的。在本章中,首先介绍什么是算法,什么是数据结构,算法和数据结构在计算学科中的地位;继而介绍数据抽象和抽象数据类型的概念;随后简要回顾面向对象方法;最后给出数据结构和算法的描述方法。

1.1 算法与数据结构

计算机由硬件和软件组成,硬件通过软件发挥效用,服务于人类。硬件是躯体,软件是灵魂,软件的核心是程序。学习程序设计需要掌握一门程序设计语言,它是学习计算机后继课程所必需的技能,但程序设计不等于编码,为了充分利用计算机资源,开发高效的程序,计算机人员还必须掌握计算机学科多方面知识:数据的组织、算法的设计和分析以及软件工程技术等。

对计算机学科来说,算法和数据结构的概念至关重要,它们是计算学科的基础之一,更是软件技术的基础。算法和数据结构之间有着本质的联系。当谈论一种算法时,自然要涉及算法所处理的数据问题,而讨论数据的组织或结构,离开了对处理此类数据的运算及其算法的研究也没有意义。有人概括过一个公式:

$$\text{算法} + \text{数据结构} = \text{程序}$$

在**计算学科教学计划 1991**(Computing Curricula, CC1991)中,算法和数据结构是计算机学科的 9 个研究领域之一。**计算学科教学计划 2001**(CC2001)将其调整成 14 个领域,数据结构和算法的基本内容主要涵盖在**算法与复杂性**(algorithms and complexity, AL)和**程序设计基础**(Programming Fundamentals, PF)两个领域中。CC2001 强调了算法和程序设计,它建议使用 90 个核心学时讲授这方面知识,占整个核心学时数的 32.2%。

1.1.1 算法

1. 什么是算法

读者在学习了一门计算机程序设计语言,如 C/C++ 或 Pascal 之后,对算法一词已不陌生。编写一个程序,实际上是在实现使用计算机求解某个问题的方法。在计算机科学中,算法一词用于描述一个可用计算机实现的问题求解(problem-solving)方法。

什么是算法?笼统地说,算法(algorithm)是求解一类问题的任意一种特殊的方法。较

严格的说法是:一个算法是对特定问题求解步骤的一种描述,它是指令的有限序列;此外,算法具有下列五个特征。

- (1) **输入(input)**:算法可以有零个或多个输入;
- (2) **输出(output)**:算法至少产生一个输出;
- (3) **确定性(definiteness)**:算法的每一条指令都有确切的定义,没有二义性;
- (4) **能行性(effectiveness)**:算法的每一条指令必须足够基本,它们可以通过已经实现的基本运算执行有限次来实现;
- (5) **有穷性(finiteness)**:算法必须总能在执行有限步之后终止。

所有算法都必须具有以上五个特征。如果取消有穷性限制,则只能称为**计算过程**(computational procedure)。描述一个算法有多种方法,可以用自然语言、流程图或程序设计语言来描述。当一个算法使用计算机程序设计语言描述时,就是**程序**(program)。但算法必须可终止,计算机程序并没有这一限制,比如一个操作系统是一个程序,却不是一个算法,它一旦运行,只要计算机不关闭,操作系统程序不会终止运行。所以,一个程序是使用程序设计语言描述的一个计算过程。

2. 算法设计与分析

对算法的研究通常包括如何设计算法、如何表示算法、如何确认算法的正确性、如何分析一个算法的效率以及如何测量程序的性能等方面。主要有以下几个方面。

(1) 如何设计算法。

一般说来,算法的设计是一项创造性活动,不可能完全自动化,但学习一些基本的**算法设计策略**(algorithm design strategy)是非常有用的。对于所求解的问题,只要符合某种算法设计策略的前提,便可以利用它设计出精致而有效的算法。在本书中,我们将介绍如下几种算法设计策略:遍历技术、分治法、贪心法、动态规划法、回溯法和分支限界法以及随机算法等。

(2) 如何确认算法。

如果一个算法对于所有允许的输入都能算出正确的结果,那么此算法是正确的。确认一个算法是否正确的活动称为**算法确认**(algorithm validation)。算法确认的目的在于确认一个算法能否正确无误地工作。使用数学方法证明程序的正确性,称为**程序证明**(program proof),此项工作目前尚在研究中。一个完整的程序证明常常比原始程序要长得多。目前,人们一般通过程序测试和调试来排错。**程序测试**(program testing)是指对程序模块或程序总体输入事先准备好的样本数据(称为**测试用例**,test case),检查该程序的输出,来发现程序存在的错误以及判定程序是否满足其设计要求的一项积极活动。测试的目的是为了“发现错误”,而不是“证明程序正确”。程序经过测试暴露了错误,需进一步诊断错误的准确位置,分析错误的原因,纠正错误。**调试**(debugging)是诊断和纠正错误的过程。

(3) 如何分析算法。

算法的分析(algorithm analysis)活动是指对算法的执行时间和所需空间的估算。求解

同一问题可以编写不同的算法,通过算法分析,可以比较两个算法的效率高低。对于算法所需的时间和空间的估算,一般不需要将算法写成程序在实际的计算机上运行。当然在算法写成程序后,可以使用样本数据,实际测量一个程序所消耗的时间和空间,这就是程序的**性能测量**(performance measurement)。

本书第二部分的重点是数据结构。在学习数据结构的同时,将在相应的章节简要引入适当的算法设计策略,并初步运用其设计思想,解析数据结构上定义的运算的算法。对数据结构算法的分析应作为数据结构学习的重要部分。关于算法设计与分析的更深入的内容将在本书第三部分讨论。第三部分内容对“数据结构与算法”或“数据结构”课程教学是可选的,可以根据需要选读其中部分或全部内容。

1.1.2 数据结构

1. 数据结构的由来

随着计算机科学与技术的发展,计算机应用已远远超出了单纯地进行科学计算的范围。从传统的应用领域,如工业控制、情报检索、企业管理、商务处理、图形图像、人工智能等诸多的数据处理领域,发展到电子政务、电子商务、办公自动化、企业资源管理、电子图书馆、远程教育、远程医疗等更广泛的领域。计算机技术已渗透到国民经济的各行各业和人们日常生活的方方面面。今天,信息技术作为现代技术的标志,已成为世界各国经济增长的主要动力。

现实世界各领域中的大量信息都必须转换成数据才能在计算机中存储、处理。数据是信息的载体。应用程序处理各种各样的数据。笼统地说,所谓**数据**(data),就是计算机加工处理的对象。数据一般分两类:**数值数据**(numerical data)和**非数值数据**(non-numerical data)。数值数据是一些整数、实数或复数,主要用于工程计算、科学计算和商务处理等。非数值数据包括字符、文字、图形、图像、语音、表格等。这类数据的特点是量大,而且往往有着复杂的内在联系。如果单纯依靠改进程序设计技巧已无法编制出高效可靠的程序,而必须对数据本身的结构进行研究。数据的组织和表示方法直接影响使用计算机求解问题的效率。算法设计通常建立在所处理数据的一定组织形式之上。在许多应用中,对于相同数据的同样处理要求,如果选择不同的数据结构,会有不同的处理效率:运算时间和存储空间。数据结构和算法两者是紧密结合的。数据结构主要是为研究和解决如何使用计算机组织和处理这些非数值问题而产生的理论、技术和方法,它已成为计算机学科研究的基本课题之一。

2. 什么是数据结构

前面提到,数据是计算机加工处理的对象。一个数据可以是由成分数据构成的具有某种结构的数据。在这里,称组成数据的成分数据为**数据元素**(data element)。一般地,数据元素可以是简单类型,如整数、实数、字符等,也可以是结构类型,如**记录**(record)。如果把每个学生的记录看做一个数据元素,它包括学号、姓名、性别等**数据项**(data item),数据项是不可