

## <<数据结构与算法>>

### 图书基本信息

书名：<<数据结构与算法>>

13位ISBN编号：9787302119982

10位ISBN编号：7302119988

出版时间：2006-1

出版时间：清华大学出版社

作者：[美] 乔兹德克 (Drozdek, A.)

页数：594

译者：郑岩,战晓苏

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

## <<数据结构与算法>>

### 内容概要

《国外计算机科学经典教材·数据结构与算法：C++版（第3版）》全面系统地介绍了计算机科学教育中的一个重要组成部分——数据结构，并以C++语言实现相关的算法。

书中主要强调了数据结构和算法之间的联系，使用面向对象的方法介绍数据结构，其内容包括算法的复杂度分析、链表、栈队列、递归技术、二叉树、图、排序以及散列。

《国外计算机科学经典教材·数据结构与算法：C++版（第3版）》还清晰地阐述了同类教材中较少提到的内存管理、数据压缩和字符串匹配主题。

书中包含大量的示例分析和图形，便于读者进一步理解和巩固所学的知识。

《国外计算机科学经典教材·数据结构与算法：C++版（第3版）》适用于计算机科学及其他相关专业的师生。

对于需要参加计算机考试，或者希望自学计算机软件开发的人员也大有裨益。

## <<数据结构与算法>>

### 作者简介

Adam Drozdek, 毕业于美国莱特州立大学, 现任迪尤肯大学计算机科学系副教授, 曾出版畅销教材, 包括Data Structures and Algorithms; n Java和Elements of Data Compression等。

## &lt;&lt;数据结构与算法&gt;&gt;

## 书籍目录

第1章 C++面向对象程序设计1.1 抽象数据类型1.2 封装1.3 继承1.4 指针1.4.1 指针和数组1.4.2 指针和复制构造函数1.4.3 指针和析构函数1.4.4 指针和引用变量1.4.5 函数指针1.5 多态性1.6 C++和面向对象程序设计1.7 标准模板库1.7.1 容器1.7.2 迭代器1.7.3 算法1.7.4 函数对象1.8 标准模板库中的向量1.9 数据结构与面向对象编程1.10 案例分析：随机访问文件1.11 习题1.12 程序设计作业第2章 复杂度分析2.1 计算复杂度和渐近复杂度2.2 大O符号2.3 大O符号的性质2.4 Q符号与@符号2.5 可能的问题2.6 复杂度举例2.7 确定渐近复杂度举例2.8 最好、平均和最坏情况2.9 阻尼复杂度2.10 NP完整性2.11 习题第3章 链表3.1 单链表3.1.1 插入3.1.2 删除3.1.3 查找3.2 双链表3.3 循环链表3.4 跳跃链表3.5 自组织链表3.6 稀疏表3.7 标准模板库中的链表3.8 标准模板库中的双端队列3.9 小结3.10 案例分析：图书馆3.11 习题3.12 程序设计作业第4章 栈与队列4.1 栈4.2 队列4.3 优先队列4.4 标准模板库中的栈4.5 标准模板库中的队列4.6 标准模板库中的优先队列4.7 案例分析：迷宫问题4.8 习题4.9 程序设计作业第5章 递归5.1 递归定义5.2 函数调用与递归实现5.3 递归调用的剖析5.4 尾部递归5.5 非尾部递归5.6 间接递归5.7 嵌套递归5.8 不合理递归5.9 回溯5.10 小结5.11 案例分析：递归下降解释器5.12 习题5.13 程序设计作业第6章 二叉树6.1 树、二叉树和二叉搜索树6.2 二叉树的实现6.3 二叉搜索树的查找6.4 树的遍历6.4.1 广度优先遍历6.4.2 深度优先遍历6.4.3 不用栈实现的深度优先遍历6.5 插入6.6 删除6.6.1 合并删除6.6.2 通过复制进行删除6.7 树的平衡6.7.1 DSW算法6.7.2 AVL树6.8 自调整树6.8.1 自重新构造树6.8.2 “张开”策略6.9 堆6.9.1 将堆作为优先队列6.9.2 将数组组织为堆6.10 波兰记号和表达式树6.11 案例分析：计算单词出现的频率6.12 习题6.13 程序设计作业第7章 多叉树7.1 B树家族7.1.1 B树7.1.2 B\*树7.1.3 B+树7.1.4 前缀B+树7.1.5 位树7.1.6 R树7.1.7 2-4树7.1.8 标准模板库中的集和多集7.1.9 标准模板库中的映射和多映射7.2 trie7.3 小结7.4 案例分析：拼写检查器7.5 习题7.6 程序设计作业第8章 图8.1 图的表示法8.2 图的遍历8.3 最短路径8.4 环的检测8.5 生成树8.6 连通性8.6.1 无向图中的连通性8.6.2 有向图中的连通性8.7 拓扑排序8.8 网络8.8.1 最大流8.8.2 成本最低的最大流8.9 匹配8.9.1 稳定匹配问题8.9.2 分配问题8.9.3 非二分图中的匹配集合8.10 欧拉（Eulerian）图与汉密尔顿（Hamiltonian）图8.10.1 欧拉图8.10.2 汉密尔顿图8.11 给图加上颜色8.12 图理论中的NP完整性问题8.12.1 派系问题8.12.2 三色问题8.12.3 顶点覆盖问题8.12.4 汉密尔顿环问题8.13 案例分析：唯一代表8.14 习题8.15 程序设计作业第9章 排序第10章 散列第11章 数据压缩第12章 内存管理第13章 字符串匹配附录A 计算大O附录B 标准模板库中的算法附录C NP完整性

## &lt;&lt;数据结构与算法&gt;&gt;

## 章节摘录

5.10 小结 在了解了所有的例子（后面还有一个例子）之后，对作为编程工具的递归有什么认识呢？

与数据结构中的其他问题一样，应该在作出正确判断的基础上使用递归。

什么时候使用递归、什么时候不使用递归，没有通用的规则，应当视情况而定。

递归的效率常常比与它等价的迭代形式低。

但是如果递归程序花费的时间为100毫秒（ms），而迭代程序花费的时间为10ms，尽管后者的速度比前者快十倍，但其中的差别很难察觉到。

另外递归程序的代码具有清晰性、可读性和简单性的特点，所以运行时间上的差距可以不考虑。

递归方法比迭代方法一般要简单一些，和原始算法在逻辑上的一致性较好。

阶乘函数和幂函数就是这样的例子，本章的后面还会看到一些更有趣的例子。

尽管每个递归过程都可以转化为迭代形式，但转化并不总是一件轻而易举的事情。

特别是转化过程可能包括对栈的显式操作。

这时“时空交换”（time-space trade-off）就发挥作用了：使用迭代方法常常需要用一种新的数据结构来实现栈的功能，而使用递归方法则减轻了程序员的工作量，即将工作转交给系统完成。

无论使用哪种方法，如果其中包括了非尾部循环，栈的维护常常需要由程序员或者系统完成。

但是程序员可以决定让谁来完成这项工作。

有两种场合，使用非递归的实现方式更为可取，尽管递归方法更自然一些。

第一种情况是在所谓的实时系统中，快速响应对程序正常发挥作用至关重要。

例如在军事环境中，在航天器中，或者在某些类型的科学实验中，响应时间是10ms还是100ms有很大的差别。

第二种情况是鼓励程序员在执行好几百次的程序中避免使用递归，这种程序的最佳例子是编译器。

不过不要太死板地看待这些规则，因为有时递归实现比非递归实现的速度还快。

硬件可能会带有内置的栈操作，显著提高了对运行时栈进行操作的函数的速度，比如递归函数。

运行一个简单的程序，分别采用递归方式和迭代方式，并对两者的运行时间进行比较，这样会有助于决定是否应该采用递归方式——实际上，递归可以比迭代方式执行得更快。

如果使用了尾部递归，这样的测试就特别重要。

不过，如果在迭代方式中还要使用栈，则推荐使用递归方式，因为这两种方式的运行时间的差距不明显——一般不会相差10倍。

.....

## <<数据结构与算法>>

### 编辑推荐

《国外计算机科学经典教材·数据结构与算法：C++版（第3版）》的示例分析贯穿全文，便于学生在真实的环境下了解数据结构的概念。

《国外计算机科学经典教材·数据结构与算法：C++版（第3版）》每章最后都提供了程序设计作业，给学生提供大量的实践机会，巩固所学内容。

《国外计算机科学经典教材·数据结构与算法：C++版（第3版）》配以丰富的图示，便于学生对数据结构有直观的理解。

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>