

<<程序员实用算法>>

图书基本信息

书名：<<程序员实用算法>>

13位ISBN编号：9787111272960

10位ISBN编号：711127296X

出版时间：2009-9

出版时间：机械工业出版社

作者：Andrew Binstock, John Rex

页数：437

译者：陈宗斌

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>

<<程序员实用算法>>

前言

数据结构与算法是计算机专业的核心课程，是计算机软件开发和应用人员必备的专业基础。今天的大多数关于算法的图书都是大学教科书，或者是令人厌倦的相同算法集合改头换面后的作品。本书是给出所有算法的完整代码实现的第一本书，这些算法对于开发人员在其日常工作中是有用的。

本书介绍了关于算法的基础知识、基本数据结构、散列、查找、排序、树、日期和时间、任意精度的算术运算、数据压缩以及数据完整性和验证等内容。

本书的目的是为在应用程序中使用的算法提供一个实用的纲要。

与关于算法的大多数著作不同的是，本书不是一本教材：书中没有提供实现细节，把它作为练习留给读者完成；也没有利用较小的代码段对算法进行高度理论化的讨论，以说明如何进行实现。

相反，本书完全用c语言实现了各种算法，并且讨论了如何在各种应用程序中最佳地使用它们。

本书只要求读者具有c语言的初级知识以及不超出基本代数之外的数学知识。

源代码是符合ANSI标准的，并且对它们进行了测试，它们都可以运行在UNIX下以及Borland、Microsoft和Watcom的编译器上。

本书非常适合于高等院校计算机专业的学生阅读，对于从事计算机软件开发的人员，也将从本书中受益匪浅。

参加本书翻译的人员有：陈宗斌、张景友、易小丽、陈婷、管学岗、王新彦、金惠敏、张海峰、徐晔、戴锋。

由于时间紧迫，加之译者水平有限，错误在所难免，恳请广大读者批评指正。

<<程序员实用算法>>

内容概要

《程序员实用算法》重点关注的是实用、立即可用的代码，并且广泛讨论了可移植性和特定于实现的细节。

《程序员实用算法》作者介绍了一些有用但很少被讨论的算法，它们可用于语音查找、日期和时间例程（直到公元1年）、B树和索引文件、数据压缩、任意精度的算术、校验和与数据验证，并且还最全面地介绍了查找例程、排序算法和数据结构。

《程序员实用算法》结构清晰，示例丰富，可作为广大程序员的参考用书。

<<程序员实用算法>>

作者简介

Andrew Binstock, 是《UNIX Review》的主编和《C Gazette》的创刊编辑。
他是《HP LaserJet Programming》(Addison-Wesley, 1991) 的第一作者。

John Rex, 是一位计算机顾问, 专攻C和C++。
他是《C Gazette》的前任技术编辑, 并且为许多杂志撰写文章。

<<程序员实用算法>>

书籍目录

译者序前言致谢第1章 绪论1.1 评估算法1.2 修改算法1.2.1 主要的优化：I/O1.2.2 主要的优化：函数调用1.3 资源和参考资料第2章 基本数据结构2.1 链表2.1.1 双向链表2.1.2 链表的其他特征2.2 栈和队列2.2.1 栈的特征2.2.2 队列的特征第3章 散列3.1 散列的概念3.2 散列函数3.3 冲突解决方法3.3.1 线性再散列法3.3.2 非线性再散列法3.3.3 外部拉链法3.4 性能问题3.5 资源和参考资料第4章 查找4.1 查找的特征4.1.1 准备时间4.1.2 运行时间4.1.3 回溯的需要4.2 蛮力查找4.3 Boyer-Moore查找4.3.1 启发式方法#1：跳过字符4.3.2 启发式方法#2：重复模式4.4 多字符串查找4.5 用于正则表达式的字符串查找：grep4.6 近似字符串匹配技术4.7 语音比较：Soundex算法4.8 Metaphone：现代的Soundex4.9 选择技术4.10 资源和参考资料4.10.1 通用参考资料4.10.2 Boyer-Moore4.10.3 多字符串查找4.10.4 正则表达式查找4.10.5 近似字符串匹配4.10.6 Soundex算法和Metaphone算法第5章 排序5.1 排序的基本特征5.1.1 稳定性5.1.2 对哨兵的需求5.1.3 对链表进行排序的能力5.1.4 输入的阶的相关性5.1.5 对额外存储空间的需求5.1.6 内部排序技术与外部排序技术5.2 排序模型5.2.1 冒泡排序5.2.2 插入排序5.2.3 希尔排序5.2.4 快速排序5.2.5 堆排序5.3 对链表进行插入排序5.4 对链表进行快速排序5.5 对多个键进行排序——不稳定排序的修正方法5.6 网络排序5.7 小结：选择一种排序算法5.8 资源和参考资料第6章 树6.1 二叉树6.1.1 树查找6.1.2 节点插入6.1.3 节点删除6.1.4 二叉查找树的性能6.1.5 AVL树6.2 红黑树6.3 伸展树6.4 B树6.4.1 保持B树平衡6.4.2 实现B树算法6.4.3 B树实现的代码6.5 可以看见森林吗6.6 资源和参考资料第7章 日期和时间7.1 日期例程的库7.2 时间例程7.3 用于日期和时间数据的格式7.4 最后的提醒7.5 资源和参考资料第8章 任意精度的算术8.1 构建计算器8.2 表示数字8.3 计算8.4 加法8.5 减法8.6 乘法8.7 除法8.8 关于计算器要注意的最后几点8.9 用于计算平方根的牛顿算法8.10 分期付款表8.11 资源和参考资料第9章 数据压缩9.1 行程编码9.2 霍夫曼压缩9.2.1 代码9.2.2 其他问题9.3 滑动窗口压缩9.4 基于字典的压缩（LZW）9.4.1 LZW算法的伪代码9.4.2 LZW压缩的实现9.4.3 填满字典9.5 使用哪种压缩方法9.6 资源和参考资料第10章 数据完整性和验证10.1 简单的校验和10.2 加权校验和10.3 循环冗余校验10.3.1 CRC-CCITT10.3.2 CRC-1610.3.3 CRC-3210.4 资源和参考资料

<<程序员实用算法>>

章节摘录

第1章 绪论 1.1 评估算法 除了最直观的应用之外，算法是所有程序的核心和灵魂。算法一般被设计用于以最小的代价高效地解决特定的问题。

算法的价值一般取决于两方面因素：如何恰当地解决问题以及如何高效地实现解决方案。这些是算法分析的定性和定量方面。

对于许多算法，质量不是一个问题。

例如，对于排序算法，必须保证每次都对所有元素正确地进行了排序。

一旦出错，就必须丢弃它并且严格说来不能将其视为一种算法。

在其他领域，不能基于这种简单的通过／失败测试来度量质量。

例如，在第4章中介绍的Soundex算法允许检索听起来相同的单词或名字。

与排序算法不同，可以调整Soundex算法，以寻找接近的匹配或者相当宽泛的匹配；这取决于实现算法的方式和开发人员的需求。

在这种情况下，质量是可度量的并且是算法的重要方面，并且指导我们认真选择不同的解决方案。

算法设计的定量方面尝试确定算法所需的资源。

一般来说，最重要的度量标准是时间：即算法运行得有多快？偶尔，计算机资源（比如可用的内存）也是一个重要因素。

度量性能 与基准的性能不同，算法的性能很少依据时间来加以说明。

在论及排序例程时，你几乎从未听到它完成排序要花费8.62秒这样的说明。

这有一个很好的理由：这种计时难以复制，并且通常依赖于正在处理的数据的具体特征。

算法不依赖于计时，而是依赖于一个直观的方程，以显示输入的大小与性能之间的关系。

用于显示这种关系的传统方法是使用符号D，称为大O表示法（big.oh notation）。

其工作方式是：假定你有一个算法，它简单地通读一个文本文件，从中查找单词flea。

一种合理的方法是寻找字母f的每个实例（参见第4章）。

当找到一个f，该算法就测试4个字母的序列，看看它是不是单词flea。

在这个示例中，显然执行时间直接与文本文件的大小成正比。

如果给定的文件包含IV个字符，那么我们就说该算法的执行时间的界限是O（N）。

你会注意到这种表述没有考虑到可能影响性能的其他因素——例如，字母f在文本中出现的频繁程度。

在查找字符串时（比如fleas rarely wear collars），字符串的长度以及其中相似字符串（比如fleasrarely weal"colors）出现的频率也会影响性能。

不过，严格来讲，这些因素是要处理的数据的函数，而不是算法的函数。

因此，在大O表示法中，它们不会出现在公式中。

该表示法只是简单地说明数据规模（一般用IV表示，偶尔也用n表示）与算法的典型性能之间的关系。

。……

<<程序员实用算法>>

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>