

学习指导与习题解答

# Schaum's Outline of Data Structures with C++



## 数据结构 (C++语言版) 学习指导与习题解答

John R. Hubbard 著

陈宗斌 等 译

清华大学出版社



学习指导与习题解答

# Schaum's Outline of Data Structures with C++



## 数 据 结 构

学习指导与习题解答

(C++语言版)

清华大学出版社  
北京

John R. Hubbard

**Schaum's Outline of Data Structures with C++**

EISBN: 978-0-07-118358-1

Copyright © 2010 The McGraw-Hill Companies, Inc.

All Rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including without limitation photocopying, recording, taping, or any data-base, information or retrieval system, without the prior written permission of the publisher.

This authorized Chinese translation edition is jointly published by McGraw-Hill Education (Asia) and Tsinghua University Press. This edition is authorized for sale in the People's Republic of China only, excluding Hong Kong, Macao SAR and Taiwan.

Copyright © 2010 by McGraw-Hill Education (Asia), a division of the Singapore Branch of The McGraw-Hill Companies, Inc. and Tsinghua University Press.

版权所有。未经出版人事先书面许可,对本出版物的任何部分不得以任何方式或途径复制或传播,包括但不限于复印、录制、录音,或通过任何数据库、信息或可检索的系统。

本授权中文简体字翻译版由麦格劳-希尔(亚洲)教育出版公司和清华大学出版社合作出版。此版本经授权仅限在中华人民共和国境内(不包括香港特别行政区、澳门特别行政区和台湾)销售。

版权© 2010 由麦格劳-希尔(亚洲)教育出版公司与清华大学出版社所有。

北京市版权局著作权合同登记号 图字:01-2009-5148 号

本书封面贴有 McGraw-Hill 公司防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

### 图书在版编目(CIP)数据

数据结构学习指导与习题解答: C++ 语言版/(美)哈伯德(Hubbard, J. R.)著;陈宗斌等译.  
—北京:清华大学出版社,2010.7

书名原文: Schaum's Outline of Data Structures with C++

ISBN 978-7-302-22679-6

I. ①数… II. ①哈… ②陈… III. ①数据结构—高等学校—教学参考资料 ②C 语言—程序设计—高等学校—教学参考资料 IV. ①TP311.12 ②TP312

中国版本图书馆 CIP 数据核字(2010)第 084966 号

责任编辑:龙啟铭

责任校对:焦丽丽

责任印制:王凤芝

出版发行:清华大学出版社

<http://www.tup.com.cn>

社 总 机:010-62770175

投稿与读者服务:010-62795954, [jsjic@tup.tsinghua.edu.cn](mailto:jsjic@tup.tsinghua.edu.cn)

质 量 反 馈:010-62772015, [zhiliang@tup.tsinghua.edu.cn](mailto:zhiliang@tup.tsinghua.edu.cn)

地 址:北京清华大学学研大厦 A 座

邮 编:100084

邮 购:010-62786544

印 刷 者:北京市清华园胶印厂

装 订 者:北京国马印刷厂

经 销:全国新华书店

开 本:185×230 印 张:32.5

版 次:2010 年 7 月第 1 版

印 数:1~3000

定 价:59.00 元

字 数:701 千字

印 次:2010 年 7 月第 1 次印刷

产品编号:034569-01

# 出版说明

计算机专业知识学习的最佳方式，就是阅读良好设计且具有完整解释的示例，并自己动手进行实践练习。本系列图书就是遵循这种写作风格，每本书首先简明扼要地说明其重要知识点，然后通过大量丰富的示例和习题，并给出详尽的答案，让读者全面透彻地理解概念，掌握每个知识点，使读者能举一反三，灵活运用理论知识解决实际问题，并检验知识的掌握程度，因此，本系列图书不仅很适合自学，也非常适合用作高等院校计算机专业核心课程的参考教材或辅助教材。

本系列图书包括：

- 计算机科学导论学习指导与习题解答
- 离散数学学习指导与习题解答
- 数据结构学习指导与习题解答（C++语言版）
- 数据结构学习指导与习题解答（Java 语言版）
- C++程序设计学习指导与习题解答
- Java 程序设计学习指导与习题解答
- 操作系统学习指导与习题解答
- 计算机体系结构学习指导与习题解答
- 软件工程学习指导与习题解答
- 计算机网络学习指导与习题解答
- 关系数据库基础学习指导与习题解答
- 计算机图形学学习指导与习题解答

本系列图书由“计算机专业课程研究组”策划、翻译和审校，读者如对本书有任何建议和意见，请来信：[tuppub.cs@gmail.com](mailto:tuppub.cs@gmail.com)。

## 作者介绍

John R. Hubbard 是里士满大学的数学和计算机科学的教授。他从密歇根大学获得了哲学博士学位（1973 年），从 1983 年起就成为了里士满大学中的教师。他主要感兴趣的是数字算法和数据库系统。Hubbard 博士还是另外几本图书的作者，包括 *Schaum's Outline of Programming with C++*、*Schaum's Outline of Programming with Java* 和 *Schaum's Outline of Fundamentals of Computing with C++*。

# 译者序

数据结构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。今天的大多数关于数据结构的图书都是大学教科书，而本书可用于自学，更适合与使用 C++ 程序设计语言的数据结构方面的常规课程配合使用。

本书首先复习了 C++ 的基础知识，然后介绍了数据结构的相关内容，包括递归、栈、队列、链表、表、树、二叉树、查找树、堆和优先级队列、排序。本书附录中列出了相关的参考资源，提供了必要的数学知识，并给出了一些算法和类的源代码实现。

本书包含大量的示例，有助于读者从中举一反三，深入学习数据结构方面的相关知识。在每一章末还提供了相关的复习题和问题，有利于学生巩固所学的知识，并且给出了所有复习题和问题的答案，为学生学习提供了方便。

本书内容简明扼要、结构组织合理、示例丰富，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、陈红霞、张景友、易小丽、陈婷、管学岗、王新彦、金惠敏、张海峰、徐晔、戴锋、张德福、张士华、张锁玲、杜明宗、高玉琢、王涛、申川、孙玲、李振国、高德杰、宫飞、侯经国、刘淑妮、张春林、李大成、程明、张路红、张淑芝、孙先国、刘冀得、梁永翔、张广东、郁琪琳、邵长凯、蒲书箴、潘曙光、刘瑞东、李军、焦敬俭等。

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。

# 前 言

像 *Schaum's Outline* 系列的所有图书一样，本书可用于自学，但更适合与使用 C++ 程序设计语言的数据结构方面的常规课程配合使用。

本书包括 200 多个示例和问题。作者坚定地相信可以从良好构造的示例及完整解释的集合中学习数据结构的原理。本书旨在提供这种支持。

可以从作者的 Web 站点 (<http://www.richmond.edu/~hubbard>、<http://www.jhubbard.net> 或 <http://www.projectEuclid.net>) 下载本书中所有示例和问题的源代码。这些站点还包含针对本书的任何校正和补充材料。

我感谢我的所有朋友、同事、学生以及 McGraw-Hill 的工作人员，他们对本书原稿给出了至关重要的评论，从而给我提供了帮助。特别感谢我的妻子 Anita Hubbard，感谢她的建议、鼓励，以及为本书提供的一些创造性问题。本书中使用的许多原创性的思想都来自于她。

John R. Hubbard  
弗吉尼亚州里士满市

# 目 录

|                            |    |
|----------------------------|----|
| 第 1 章 C++复习 .....          | 1  |
| 1.1 标准 C++程序设计语言 .....     | 1  |
| 1.2 条件语句.....              | 3  |
| 1.3 运算符.....               | 5  |
| 1.4 迭代.....                | 8  |
| 1.5 函数.....                | 10 |
| 1.6 字符串.....               | 12 |
| 1.7 文件.....                | 14 |
| 复习题.....                   | 15 |
| 练习题.....                   | 16 |
| 复习题答案.....                 | 18 |
| 练习题答案.....                 | 20 |
| 第 2 章 指针与数组 .....          | 32 |
| 2.1 指针.....                | 32 |
| 2.2 派生类型.....              | 34 |
| 2.3 引用.....                | 35 |
| 2.4 按引用传递.....             | 35 |
| 2.5 空指针异常.....             | 37 |
| 2.6 new 和 delete 运算符 ..... | 38 |
| 2.7 数组.....                | 39 |
| 2.8 动态数组.....              | 40 |
| 2.9 把数组传递给函数 .....         | 41 |
| 2.10 多维数组.....             | 42 |
| 复习题.....                   | 43 |
| 练习题.....                   | 44 |
| 复习题答案.....                 | 46 |
| 练习题答案.....                 | 47 |



|                      |     |
|----------------------|-----|
| 第3章 类                | 60  |
| 3.1 Point 类          | 60  |
| 3.2 实例、隐式形参和 this 指针 | 63  |
| 3.3 编译类及其客户程序        | 64  |
| 3.4 友元函数             | 68  |
| 3.5 Line 类           | 68  |
| 3.6 用于随机数的类          | 71  |
| 3.7 静态成员             | 73  |
| 3.8 复合               | 75  |
| 3.9 继承               | 78  |
| 复习题                  | 81  |
| 练习题                  | 82  |
| 复习题答案                | 85  |
| 练习题答案                | 87  |
| 第4章 递归               | 98  |
| 4.1 阶乘函数             | 98  |
| 4.2 跟踪递归调用           | 99  |
| 4.3 斐波纳契数列           | 99  |
| 4.4 二项式系数            | 101 |
| 4.5 欧几里得算法           | 102 |
| 4.6 正确性的归纳证明         | 103 |
| 4.7 递归算法的复杂度分析       | 104 |
| 4.8 动态程序设计           | 105 |
| 4.9 汉诺塔              | 105 |
| 4.10 相互递归            | 107 |
| 复习题                  | 108 |
| 练习题                  | 108 |
| 复习题答案                | 110 |
| 练习题答案                | 111 |
| 第5章 栈                | 119 |
| 5.1 stack 接口         | 119 |
| 5.2 使用 stack 对象      | 119 |

---

|                          |            |
|--------------------------|------------|
| 5.3 栈的应用.....            | 121        |
| 5.4 消除递归.....            | 124        |
| 5.5 连续实现.....            | 126        |
| 5.6 链表实现.....            | 129        |
| 复习题.....                 | 133        |
| 练习题.....                 | 134        |
| 复习题答案.....               | 135        |
| 练习题答案.....               | 136        |
| <b>第6章 队列.....</b>       | <b>142</b> |
| 6.1 queue 接口.....        | 142        |
| 6.2 使用 queue 对象.....     | 142        |
| 6.3 队列的应用.....           | 144        |
| 6.4 连续实现.....            | 149        |
| 6.5 链表实现.....            | 153        |
| 复习题.....                 | 155        |
| 练习题.....                 | 156        |
| 复习题答案.....               | 157        |
| 练习题答案.....               | 157        |
| <b>第7章 链表.....</b>       | <b>164</b> |
| 7.1 list 接口.....         | 164        |
| 7.2 使用 list 对象.....      | 165        |
| 7.3 迭代器.....             | 166        |
| 7.4 应用.....              | 168        |
| 7.5 循环链表.....            | 173        |
| 7.6 有序链表.....            | 176        |
| 7.7 不受限制的 Integer 类..... | 177        |
| 7.8 List 类的实现.....       | 183        |
| 复习题.....                 | 188        |
| 练习题.....                 | 189        |
| 复习题答案.....               | 191        |
| 练习题答案.....               | 192        |

|                       |     |
|-----------------------|-----|
| 第 8 章 表               | 200 |
| 8.1 标准的 pair 类型       | 200 |
| 8.2 使用 map 类模板的应用     | 202 |
| 8.3 散列表               | 207 |
| 8.4 散列函数              | 212 |
| 8.5 单独串接              | 215 |
| 复习题                   | 219 |
| 练习题                   | 219 |
| 复习题答案                 | 219 |
| 练习题答案                 | 220 |
| 第 9 章 树               | 224 |
| 9.1 树的术语              | 224 |
| 9.2 决策树和转换图           | 226 |
| 9.3 树遍历算法             | 230 |
| 9.4 Tree 类的接口         | 231 |
| 9.5 Tree 类的实现         | 234 |
| 复习题                   | 238 |
| 练习题                   | 240 |
| 复习题答案                 | 244 |
| 练习题答案                 | 245 |
| 第 10 章 二叉树            | 259 |
| 10.1 定义               | 259 |
| 10.2 统计二叉树            | 261 |
| 10.3 完整二叉树            | 262 |
| 10.4 恒等、相等和同构         | 263 |
| 10.5 完全二叉树            | 264 |
| 10.6 树遍历              | 266 |
| 10.7 表达式树             | 268 |
| 10.8 森林               | 270 |
| 10.9 BinaryTree 类的接口  | 271 |
| 10.10 BinaryTree 类的实现 | 274 |
| 复习题                   | 278 |

|                                                |            |
|------------------------------------------------|------------|
| 练习题.....                                       | 279        |
| 复习题答案.....                                     | 282        |
| 练习题答案.....                                     | 283        |
| <b>第 11 章 查找树.....</b>                         | <b>291</b> |
| 11.1 二叉查找树.....                                | 291        |
| 11.2 二叉查找树的实现.....                             | 293        |
| 11.3 二叉查找树的性能特征.....                           | 296        |
| 11.4 AVL 树.....                                | 297        |
| 复习题.....                                       | 300        |
| 练习题.....                                       | 300        |
| 复习题答案.....                                     | 301        |
| 练习题答案.....                                     | 301        |
| <b>第 12 章 堆和优先级队列.....</b>                     | <b>305</b> |
| 12.1 堆.....                                    | 305        |
| 12.2 自然映射.....                                 | 305        |
| 12.3 插入到堆中.....                                | 307        |
| 12.4 从堆中删除.....                                | 308        |
| 12.5 优先级队列.....                                | 309        |
| 12.6 使用 <code>priority_queue</code> 对象.....    | 309        |
| 12.7 使用堆实现 <code>PriorityQueue</code> 类模板..... | 311        |
| 12.8 优先级队列的实现.....                             | 313        |
| 复习题.....                                       | 318        |
| 练习题.....                                       | 318        |
| 复习题答案.....                                     | 319        |
| 练习题答案.....                                     | 319        |
| <b>第 13 章 排序.....</b>                          | <b>322</b> |
| 13.1 基础知识.....                                 | 322        |
| 13.2 冒泡排序.....                                 | 323        |
| 13.3 选择排序.....                                 | 324        |
| 13.4 插入排序.....                                 | 325        |
| 13.5 归并排序.....                                 | 327        |
| 13.6 快速排序.....                                 | 330        |

|                              |     |
|------------------------------|-----|
| 13.7 堆.....                  | 331 |
| 13.8 堆排序.....                | 332 |
| 13.9 希尔排序.....               | 337 |
| 13.10 比较排序的速度限制 .....        | 338 |
| 复习题.....                     | 338 |
| 练习题.....                     | 338 |
| 复习题答案.....                   | 340 |
| 练习题答案.....                   | 340 |
| 附录 A 参考资料 .....              | 344 |
| 附录 B 必要的数学知识 .....           | 353 |
| B.1 下取整函数和上取整函数.....         | 353 |
| B.2 对数 .....                 | 353 |
| B.3 数学归纳法的第一定律.....          | 354 |
| B.4 数学归纳法的第二定律.....          | 355 |
| B.5 几何级数 .....               | 356 |
| B.6 求和公式 .....               | 357 |
| B.7 渐近复杂度等级 .....            | 357 |
| B.8 调和数 .....                | 358 |
| B.9 斯特灵公式 .....              | 360 |
| B.10 斐波纳契数字 .....            | 361 |
| B.11 黄金分割率 .....             | 361 |
| B.12 欧几里得算法 .....            | 363 |
| 附录 C 标准容器类 .....             | 365 |
| C.1 vector 类模板 .....         | 365 |
| C.2 deque 类模板 .....          | 372 |
| C.3 stack 类模板 .....          | 372 |
| C.4 queue 类模板 .....          | 372 |
| C.5 priority_queue 类模板 ..... | 373 |
| C.6 list 类模板 .....           | 374 |
| C.7 map 类模板 .....            | 377 |
| C.8 set 类模板 .....            | 379 |

---

|                                 |     |
|---------------------------------|-----|
| 附录 D 泛型算法 .....                 | 382 |
| 附录 E 示例类 .....                  | 417 |
| E.1 BinaryTree 类 .....          | 417 |
| E.2 BinarySearchTree 类 .....    | 425 |
| E.3 Card 类 .....                | 427 |
| E.4 Concordance 类 .....         | 430 |
| E.5 Date 类 .....                | 432 |
| E.6 Deck 类 .....                | 440 |
| E.7 Hand 类 .....                | 441 |
| E.8 Hash 函数结构模板 .....           | 442 |
| E.9 HashTable 类模板 .....         | 442 |
| E.10 Line 类 .....               | 445 |
| E.11 List 类模板 .....             | 447 |
| E.12 Matrix 类模板 .....           | 454 |
| E.13 OrderedList 类 .....        | 456 |
| E.14 Person 类 .....             | 456 |
| E.15 Point 类 .....              | 460 |
| E.16 Polynomial 类 .....         | 463 |
| E.17 PriorityQueue 类模板 .....    | 470 |
| E.18 Purse 类 .....              | 472 |
| E.19 Queue 类 .....              | 474 |
| E.20 Random 类 .....             | 476 |
| E.21 RandomLine 类 .....         | 477 |
| E.22 RandomPoint 类 .....        | 477 |
| E.23 Ratio 类 .....              | 478 |
| E.24 Rational 类 .....           | 481 |
| E.25 SelfOrganizingList 类 ..... | 484 |
| E.26 Stack 类模板 .....            | 484 |
| E.27 Tree 类 .....               | 486 |

# 第 1 章 C++ 复习

本章复习了 C++ 的基本特性。有关更详细的信息，参阅附录 A 中列出的 [Stroustrup2] 和 [Hubbard1]。

## 1.1 标准 C++ 程序设计语言

C++ 程序设计语言是由 Bjarne Stroustrup 在 1980 年构建一种分布式计算系统时发明的。他把 C++ 建立在由 Dennis Ritchie 于 1972 年在贝尔实验室发明的 C 程序设计语言的基础之上。之所以使用 C 这个名称，是因为该语言是一种名称为 B 的语言的后继语言，B 语言是由 Ken Thompson 发明的一种无类型的程序设计语言，它又是 Martin Richards 于 1967 年发明的一种名称为 BCPL (Basic Combined Programming Language, 基本组合式程序设计语言) 的语言的后继语言。名称 C++ 用于暗示它是 C 的增补版本。Stroustrup 通过添加类来增补 C。的确，他最初把该语言命名为“带有类的 C”。促进面向对象程序设计的类特性来自于 20 世纪 60 年代早期开发的 Simula 程序设计语言。

1998 年，C++ 程序设计语言由 ISO (International Standards Organization, 国际标准化组织) 和 ANSI (American National Standards Institute, 美国国家标准协会) 标准化。这种新的标准包括最初由 Alexander Stepanov 于 1979 年开发的 STL (Standard Template Library, 标准模板库)。术语“标准 C++”就是指该语言的这个标准化版本。

### 示例 1.1 标准 C++ 的“Hello World”程序

```
#include <iostream>           // defines the std::cout and std::endl objects
int main()
{    // prints "Hello, World!"
    std::cout << "Hello, World!" << std::endl;
}
Hello, World!
```

第 1 行中的预处理器指令告诉 C++ 编译器包括来自标准头 (standard header) 文件 `iostream` 中的定义，该文件是标准 C++ 库的一部分。它在 `std` 命名空间中定义了 `cout` 对象和 `endl` 对象。作用域解析运算符 (scope resolution operator) `::` 用于指示这些定义的位置。

### 示例 1.2 使用标准的 `std` 命名空间

```
#include <iostream>           // defines the std::cout and std::endl objects
```

```
using namespace std;           // renders the std:: prefix unnecessary
int main()
{   // prints "Hello, World!"
    cout << "Hello, World!" << endl;
}
Hello, World!
```

第 2 行中的 `using` 声明把名称 `std` 添加到本地作用域中, 从而消除了在 `cout` 和 `endl` 对象前添加作用域解析前缀的需要。

本书中其余的所有程序都假定开始于下面两行:

```
#include <iostream>
using namespace std;
```

如果使用标准化之前的编译器, 可代之以使用下面这一行语句:

```
#include <iostream.h>
```

### 示例 1.3 求解二次方程

```
#include <cmath>                // defines the sqrt() function
int main()
{   // implements the quadratic formula
    double a, b, c;
    cout << "Enter the coefficients of a quadratic equation:\n";
    cout << "\ta: "; cin >> a;
    cout << "\tb: "; cin >> b;
    cout << "\tc: "; cin >> c;
    cout << "The equation is: " << a << "x*x + " << b
        << "x + " << c << " = 0\n";
    double d = b*b - 4*a*c;      // discriminant
    double sqrted = sqrt(d);
    double x1 = (-b + sqrted)/(2*a);
    double x2 = (-b - sqrted)/(2*a);
    cout << "The solutions are:\n";
    cout << "\tx1 = " << x1 << endl;
    cout << "\tx2 = " << x2 << endl;
    cout << "Check:\ta*x1*x1 + b*x1 + c = " << a*x1*x1 + b*x1 + c;
    cout << "\n \ta*x2*x2 + b*x2 + c = " << a*x2*x2 + b*x2 + c;
}
```

在第一次运行该程序时, 输入 2、1 和 -3, 解二次方程  $2x^2 + x - 3 = 0$ :

```
Enter the coefficients of a quadratic equation:
```



```

a: 2
b: 1
c: -3
The equation is: 2*x*x + 1*x + -3 = 0
The solutions are:
x1 = 1
x2 = -1.5
Check: a*x1*x1 + b*x1 + c = 0
a*x2*x2 + b*x2 + c = 0

```

程序会计算出正确的解：1 和-1.5，然后通过把它们代入回二次表达式中得到 0 来校验它们。

在第二次运行时，我们输入 2、1 和 3，解二次方程  $2x^2 + x + 3 = 0$ 。程序为  $x_1$  和  $x_2$  的解以及由之产生的校验计算输出符号 `nan`。该符号代表“不是一个数字”。它是由于判别式  $d = b^2 - 4ac < 0$  这一事实产生的。因此，对平方根函数的调用 `sqrt(d)` 会失败，从而返回值 `nan`。这是 `float` 和 `double` 变量的有效值。但它是幂等（idempotent）：当以算术方式与任何其他值结合时，得到的值也是 `nan`。

```

Enter the coefficients of a quadratic equation:
a: 2
b: 1
c: 3
The equation is: 2*x*x + 1*x + 3 = 0
The solutions are:
x1 = nan
x2 = nan
Check: a*x1*x1 + b*x1 + c = nan
a*x2*x2 + b*x2 + c = nan

```

## 1.2 条件语句

我们可以通过使用 `if` 语句单独处理判别式为负的情况，改进示例 1.3 中的程序。

### 示例 1.4 求解二次方程的更健壮实现

```

#include <cmath> // defines the function sqrt() function
int main()
{ // implements the quadratic formula
  double a, b, c;
  cout << "Enter the coefficients of a quadratic equation:" << endl;

```