

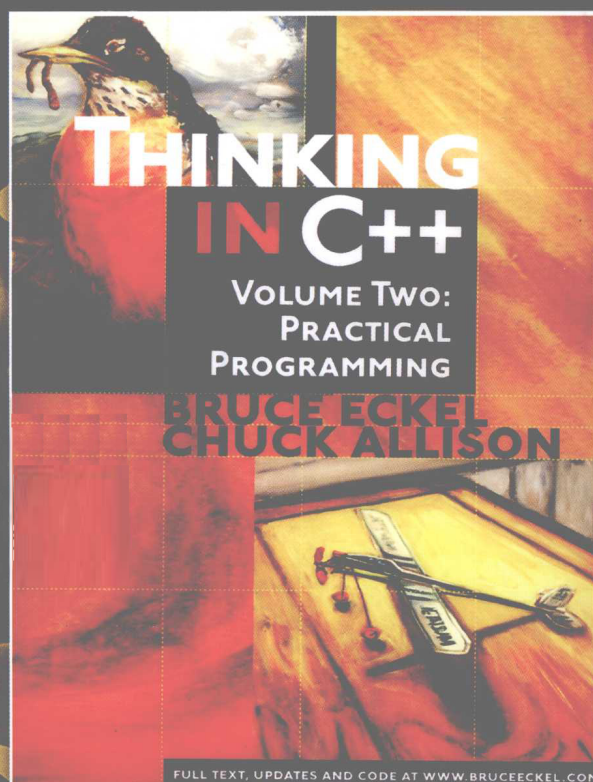
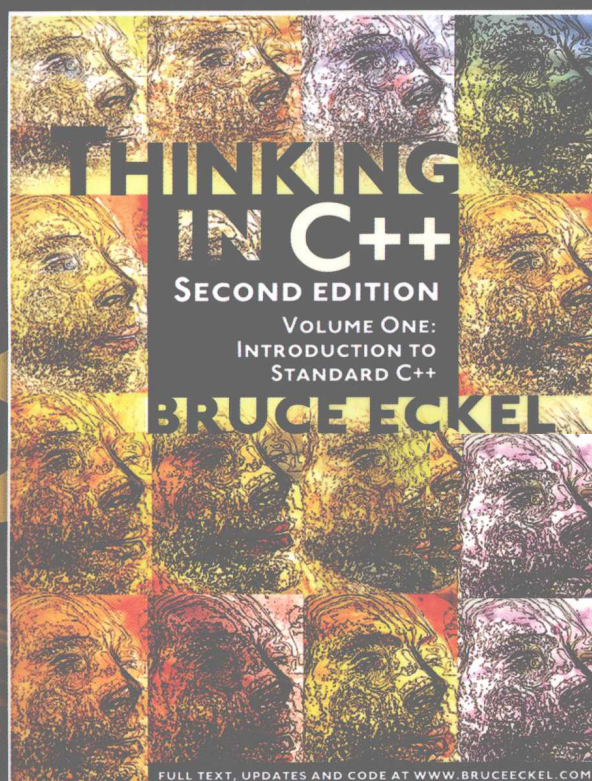
C++编程思想

第1卷 标准C++导引（原书第2版）&
第2卷 实用编程技术

（美）Bruce Eckel Chuck Allison 著 刘宗田 袁兆山 潘秋菱 刁成嘉 等译

Thinking in C++

Volume One: Introduction to Standard C++, Second Edition
& Volume Two: Practical Programming



计 算 机 科 学 丛 书

两卷合订本

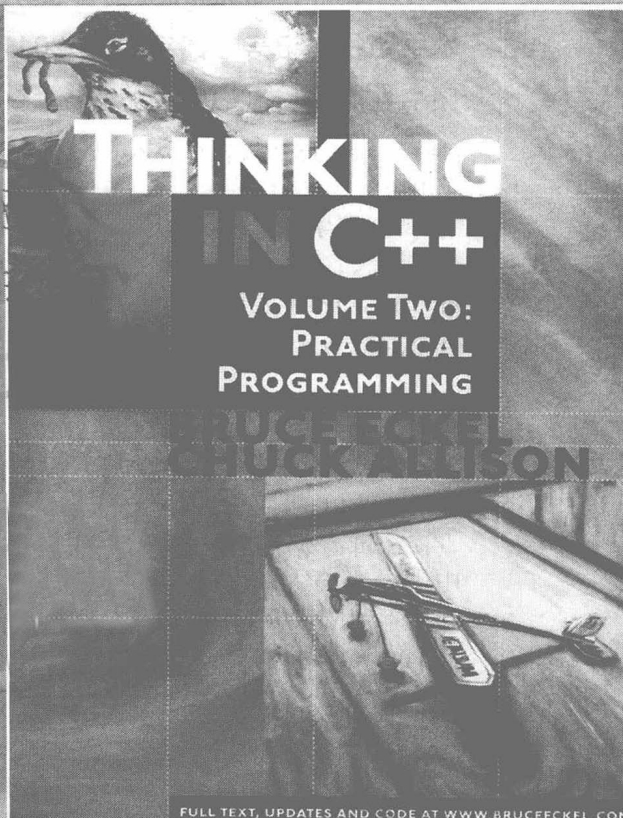
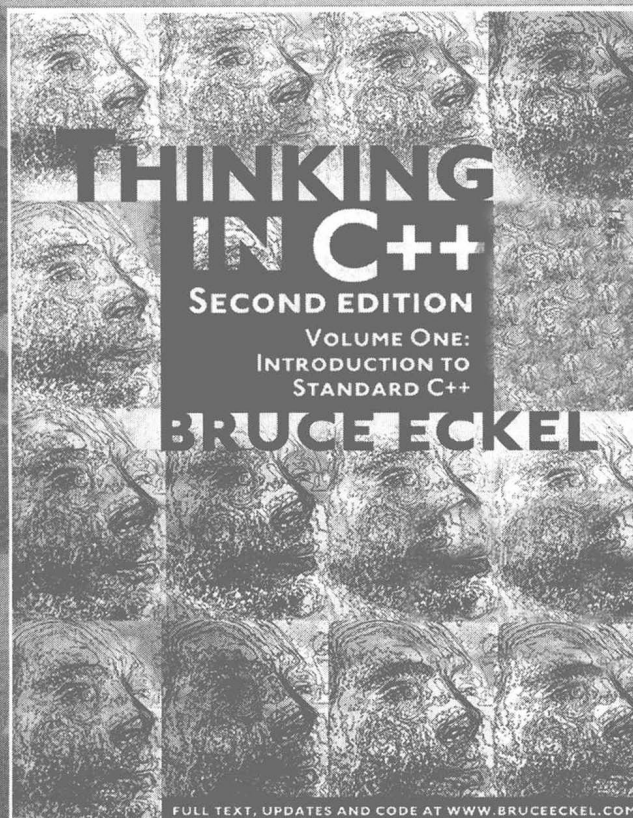
C++编程思想

第1卷 标准C++导引（原书第2版）&
第2卷 实用编程技术

（美）Bruce Eckel Chuck Allison 著 刘宗田 袁兆山 潘秋菱 刁成嘉 等译

Thinking in C++

Volume One: Introduction to Standard C++, Second Edition
& Volume Two: Practical Programming



机械工业出版社
China Machine Press

本书曾荣获美国《软件开发》杂志评选的1996年Jolt生产力大奖，中文版自2000年推出以来，经久不衰，获得了读者的充分肯定和高度评价。

本书的第1卷是在第1版的基础上进行了更加深入的分析 and 修改后得到的第2版，其内容更加集中，可以供不同程度的读者选择阅读。本书第2卷介绍了C++实用的编程技术和最佳的实践方法，深入探究了异常处理方法和异常安全设计；介绍C++的字符串、输入输出流的现代用法；解释多重继承问题的难点，描述了典型的设计模式及其实现，特别介绍了多线程处理编程技术。

本书是C++领域内一本权威的著作，书中的内容、讲授方法、练习既适合课堂教学，又适合读者自学。本书适合作为高等院校计算机及相关专业的本科生、研究生的教材，也可供从事软件开发的研究人员和科技工作者参考。

Authorized translation from the English language edition, entitled THINKING IN C++: INTRODUCTION TO STANDARD C++, VOLUME ONE, 2E, 9780139798092 by ECKEL, BRUCE, published by Pearson Education, Inc., publishing as Prentice Hall, Copyright © 1999 and THINKING IN C++, VOLUME Two: PRACTICAL PROGRAMMING, 1E, 0130353132, by ECKEL, BRUCE, ALLISON, CHUCK, Copyright © 2003.

All rights reserved.

CHINESE SIMPLIFIED language edition published by PEARSON EDUCATION ASIA LTD., and CHINA MACHINE PRESS Copyright © 2011.

本书封面贴有Pearson Education（培生教育出版集团）激光防伪标签，无标签者不得销售。

封底无防伪标均为盗版

版权所有，侵权必究

本书法律顾问 北京市展达律师事务所

本书版权登记号：图字：01-2011-3363

图书在版编目（CIP）数据

C++编程思想（两卷合订本）/（美）埃克尔（Eckel, B.）等著；刘宗田等译. —北京：机械工业出版社，2011.7

（计算机科学丛书）

书名原文：Thinking in C++: Volume One: Introduction to Standard C++, Second Edition & Volume Two: Practical Programming

ISBN 978-7-111-35021-7

I. C… II. ①埃… ②刘… III. C语言—程序设计 IV. TP312

中国版本图书馆CIP数据核字（2011）第110972号

机械工业出版社（北京市西城区百万庄大街22号 邮政编码 100037）

责任编辑：秦 健

中国电影出版社印刷厂印刷

2011年7月第1版第1次印刷

185mm×260mm · 59印张

标准书号：ISBN 978-7-111-35021-7

定价：116.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

客服热线：(010) 88378991；88361066

购书热线：(010) 68326294；88379649；68995259

投稿热线：(010) 88379604

读者信箱：hzjsj@hzbook.com

文艺复兴以降，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域取得了垄断性的优势；也正是这样的传统，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅擘划了研究的范畴，还揭示了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界都既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短的现状下，美国等发达国家在其计算机科学发展的几十年间积淀和发展的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起到积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章公司较早意识到“出版要为教育服务”。自1998年开始，我们就将工作重点放在了遴选、移译国外优秀教材上。经过多年的不懈努力，我们与Pearson, McGraw-Hill, Elsevier, MIT, John Wiley & Sons, Cengage等世界著名出版公司建立了良好的合作关系，从他们现有的数百种教材中甄选出Andrew S. Tanenbaum, Bjarne Stroustrup, Brain W. Kernighan, Dennis Ritchie, Jim Gray, Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullman, Abraham Silberschatz, William Stallings, Donald E. Knuth, John L. Hennessy, Larry L. Peterson等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及珍藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力襄助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工作；而原书的作者也相当关注其作品在中国的传播，有的还专程为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近两百个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍。其影印版“经典原版书库”作为姊妹篇也被越来越多实施双语教学的学校所采用。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证。随着计算机科学与技术专业学科建设的不断完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都将步入一个新的阶段，我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。华章公司欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方式如下：

华章网站：www.hzbook.com

电子邮件：hzjsj@hzbook.com

联系电话：(010) 88379604

联系地址：北京市西城区百万庄南街1号

邮政编码：100037



华章教育

华章科技图书出版中心

出版说明

Thinking in C++: Volume One: Introduction to Standard C++, Second Edition & Volume Two: Practical Programming

为更好地满足读者的不同需求，这次特别策划出版《C++编程思想（两卷合订本）》。根据读者对前一版本的意见，我们对图书进行了认真的更正和完善；同时，将光盘中的内容作为网络下载服务免费提供，以方便更多的学习者。此外，合订本的定价也将略低于单独购买两本图书的总价。

出版者的话
出版说明

第1卷 标准C++导引

译者序	3
前言	5
第1章 对象导言	11
1.1 抽象的过程	11
1.2 对象有一个接口	12
1.3 实现的隐藏	14
1.4 实现的重用	15
1.5 继承：重用接口	15
1.5.1 is-a 关系和is-like-a 关系	18
1.6 具有多态性的可互换对象	18
1.7 创建和销毁对象	21
1.8 异常处理：应对错误	22
1.9 分析和设计	22
1.9.1 第0阶段：制定计划	24
1.9.2 第1阶段：我们在做什么	24
1.9.3 第2阶段：我们将如何建立对象	26
1.9.4 第3阶段：创建核心	28
1.9.5 第4阶段：迭代用例	29
1.9.6 第5阶段：进化	29
1.9.7 计划的回报	30
1.10 极限编程	30
1.10.1 先写测试	31
1.10.2 结对编程	32
1.11 为什么C++会成功	32
1.11.1 一个较好的C	32
1.11.2 延续式的学习过程	33
1.11.3 效率	33
1.11.4 系统更容易表达和理解	33

1.11.5 尽量使用库	33
1.11.6 利用模板的源代码重用	34
1.11.7 错误处理	34
1.11.8 大型程序设计	34
1.12 为向OOP转变而采取的策略	34
1.12.1 指导方针	35
1.12.2 管理的障碍	35
1.13 小结	37
第2章 对象的创建与使用	38
2.1 语言的翻译过程	38
2.1.1 解释器	38
2.1.2 编译器	39
2.1.3 编译过程	39
2.2 分段编译工具	40
2.2.1 声明与定义	40
2.2.2 连接	44
2.2.3 使用库文件	44
2.3 编写第一个C++程序	45
2.3.1 使用iostream类	45
2.3.2 名字空间	46
2.3.3 程序的基本结构	47
2.3.4 “Hello, World!”	47
2.3.5 运行编译器	48
2.4 关于输入输出流	48
2.4.1 字符数组的拼接	49
2.4.2 读取输入数据	49
2.4.3 调用其他程序	50
2.5 字符串简介	50
2.6 文件的读写	51
2.7 vector简介	52
2.8 小结	55
2.9 练习	56

第3章 C++中的C	57	3.7.4 逻辑运算符	85
3.1 创建函数	57	3.7.5 位运算符	85
3.1.1 函数的返回值	58	3.7.6 移位运算符	86
3.1.2 使用C的函数库	59	3.7.7 一元运算符	88
3.1.3 通过库管理器创建自己的库	59	3.7.8 三元运算符	88
3.2 执行控制语句	60	3.7.9 逗号运算符	89
3.2.1 真和假	60	3.7.10 使用运算符时的常见问题	89
3.2.2 if-else 语句	60	3.7.11 转换运算符	90
3.2.3 while 语句	61	3.7.12 C++的显式转换	90
3.2.4 do-while 语句	61	3.7.13 sizeof ——独立运算符	93
3.2.5 for 语句	62	3.7.14 asm 关键字	94
3.2.6 关键字 break 和 continue	63	3.7.15 显式运算符	94
3.2.7 switch 语句	64	3.8 创建复合类型	94
3.2.8 使用和滥用 goto	65	3.8.1 用 typedef 命名别名	95
3.2.9 递归	65	3.8.2 用 struct 把变量结合在一起	95
3.3 运算符简介	66	3.8.3 用 enum 提高程度清晰度	97
3.3.1 优先级	66	3.8.4 用 union 节省内存	98
3.3.2 自增和自减	67	3.8.5 数组	99
3.4 数据类型简介	67	3.9 调试技巧	106
3.4.1 基本内建类型	67	3.9.1 调试标记	106
3.4.2 bool 类型与 true 和 false	68	3.9.2 把变量和表达式转换成字符串	108
3.4.3 说明符	69	3.9.3 C语言 assert() 宏	108
3.4.4 指针简介	70	3.10 函数地址	109
3.4.5 修改外部对象	72	3.10.1 定义函数指针	109
3.4.6 C++引用简介	74	3.10.2 复杂的声明和定义	109
3.4.7 用指针和引用作为修饰符	75	3.10.3 使用函数指针	110
3.5 作用域	76	3.10.4 指向函数的指针数组	111
3.5.1 实时定义变量	77	3.11 make : 管理分段编译	111
3.6 指定存储空间分配	78	3.11.1 make 的行为	112
3.6.1 全局变量	78	3.11.2 本书中的 makefile	114
3.6.2 局部变量	79	3.11.3 makefile 的一个例子	114
3.6.3 静态变量	80	3.12 小结	116
3.6.4 外部变量	81	3.13 练习	116
3.6.5 常量	82	第4章 数据抽象	119
3.6.6 volatile 变量	83	4.1 一个袖珍C库	119
3.7 运算符及其使用	83	4.1.1 动态存储分配	122
3.7.1 赋值	83	4.1.2 有害的猜测	124
3.7.2 数学运算符	83	4.2 哪儿出问题	125
3.7.3 关系运算符	85	4.3 基本对象	126

4.4 什么是对象	130	6.6 聚合初始化	166
4.5 抽象数据类型	131	6.7 默认构造函数	168
4.6 对象细节	131	6.8 小结	169
4.7 头文件形式	132	6.9 练习	169
4.7.1 头文件的重要性	132	第7章 函数重载与默认参数	171
4.7.2 多次声明问题	133	7.1 名字修饰	172
4.7.3 预处理器指示 #define 、 #ifdef 和 #endif	134	7.1.1 用返回值重载	172
4.7.4 头文件的标准	134	7.1.2 类型安全连接	172
4.7.5 头文件中的名字空间	135	7.2 重载的例子	173
4.7.6 在项目中使用头文件	135	7.3 联合	176
4.8 嵌套结构	136	7.4 默认参数	178
4.8.1 全局作用域解析	138	7.4.1 占位符参数	179
4.9 小结	139	7.5 选择重载还是默认参数	180
4.10 练习	139	7.6 小结	183
第5章 隐藏实现	142	7.7 练习	183
5.1 设置限制	142	第8章 常量	185
5.2 C++的访问控制	142	8.1 值替代	185
5.2.1 protected 说明符	144	8.1.1 头文件里的 const	186
5.3 友元	144	8.1.2 const 的安全性	186
5.3.1 嵌套友元	146	8.1.3 聚合	187
5.3.2 它是纯面向对象的吗	148	8.1.4 与C语言的区别	187
5.4 对象布局	148	8.2 指针	188
5.5 类	149	8.2.1 指向 const 的指针	189
5.5.1 用访问控制来修改 Stash	151	8.2.2 const 指针	189
5.5.2 用访问控制来修改 Stack	151	8.2.3 赋值和类型检查	190
5.6 句柄类	152	8.3 函数参数和返回值	191
5.6.1 隐藏实现	152	8.3.1 传递 const 值	191
5.6.2 减少重复编译	152	8.3.2 返回 const 值	191
5.7 小结	154	8.3.3 传递和返回地址	193
5.8 练习	154	8.4 类	195
第6章 初始化与清除	156	8.4.1 类里的 const	196
6.1 用构造函数确保初始化	156	8.4.2 编译期间类里的常量	198
6.2 用析构函数确保清除	157	8.4.3 const 对象和成员函数	200
6.3 清除定义块	159	8.5 volatile	204
6.3.1 for 循环	160	8.6 小结	205
6.3.2 内存分配	161	8.7 练习	205
6.4 带有构造函数和析构函数的 Stash	162	第9章 内联函数	207
6.5 带有构造函数和析构函数的 Stack	164	9.1 预处理器的缺陷	207

9.1.1 宏和访问	209	11.3.1 按值传递和返回	257
9.2 内联函数	210	11.3.2 拷贝构造函数	261
9.2.1 类内部的内联函数	210	11.3.3 默认拷贝构造函数	265
9.2.2 访问函数	211	11.3.4 替代拷贝构造函数的方法	266
9.3 带内联函数的 Stash 和 Stack	215	11.4 指向成员的指针	267
9.4 内联函数和编译器	218	11.4.1 函数	269
9.4.1 限制	219	11.5 小结	271
9.4.2 向前引用	219	11.6 练习	271
9.4.3 在构造函数和析构函数里隐藏行为	220	第12章 运算符重载	274
9.5 减少混乱	220	12.1 两个极端	274
9.6 预处理器的更多特征	221	12.2 语法	274
9.6.1 标志粘贴	222	12.3 可重载的运算符	275
9.7 改进的错误检查	222	12.3.1 一元运算符	276
9.8 小结	225	12.3.2 二元运算符	279
9.9 练习	225	12.3.3 参数和返回值	288
第10章 名字控制	227	12.3.4 不常用的运算符	290
10.1 来自C语言中的静态元素	227	12.3.5 不能重载的运算符	295
10.1.1 函数内部的静态变量	227	12.4 非成员运算符	296
10.1.2 控制连接	230	12.4.1 基本方针	297
10.1.3 其他存储类型说明符	232	12.5 重载赋值符	297
10.2 名字空间	232	12.5.1 operator= 的行为	298
10.2.1 创建一个名字空间	232	12.6 自动类型转换	306
10.2.2 使用名字空间	234	12.6.1 构造函数转换	306
10.2.3 名字空间的使用	237	12.6.2 运算符转换	307
10.3 C++中的静态成员	238	12.6.3 类型转换例子	309
10.3.1 定义静态数据成员的存储	238	12.6.4 自动类型转换的缺陷	310
10.3.2 嵌套类和局部类	241	12.7 小结	312
10.3.3 静态成员函数	242	12.8 练习	312
10.4 静态初始化的相依性	244	第13章 动态对象创建	315
10.4.1 怎么办	245	13.1 对象创建	315
10.5 替代连接说明	250	13.1.1 C从堆中获取存储单元的方法	316
10.6 小结	250	13.1.2 operator new	317
10.7 练习	251	13.1.3 operator delete	317
第11章 引用和拷贝构造函数	254	13.1.4 一个简单的例子	318
11.1 C++中的指针	254	13.1.5 内存管理的开销	318
11.2 C++中的引用	254	13.2 重新设计前面的例子	319
11.2.1 函数中的引用	255	13.2.1 使用 delete void* 可能会出错	319
11.2.2 参数传递准则	257	13.2.2 对指针的清除责任	320
11.3 拷贝构造函数	257	13.2.3 指针的 Stash	320

13.3 用于数组的new和delete	324	第15章 多态性和虚函数	364
13.3.1 使指针更像数组	325	15.1 C++程序员的演变	364
13.4 耗尽内存	325	15.2 向上类型转换	365
13.5 重载new和delete	326	15.3 问题	366
13.5.1 重载全局new和delete	327	15.3.1 函数调用捆绑	366
13.5.2 对于一个类重载new和delete	328	15.4 虚函数	366
13.5.3 为数组重载new和delete	330	15.4.1 扩展性	367
13.5.4 构造函数调用	332	15.5 C++如何实现晚捆绑	369
13.5.5 定位new和delete	333	15.5.1 存放类型信息	370
13.6 小结	334	15.5.2 虚函数功能图示	371
13.7 练习	334	15.5.3 撩开面纱	372
第14章 继承和组合	336	15.5.4 安装vpointer	373
14.1 组合语法	336	15.5.5 对象是不同的	373
14.2 继承语法	337	15.6 为什么需要虚函数	374
14.3 构造函数的初始化表达式表	339	15.7 抽象基类和纯虚函数	375
14.3.1 成员对象初始化	339	15.7.1 纯虚定义	378
14.3.2 在初始化表达式表中的内建类型	339	15.8 继承和VTABLE	378
14.4 组合和继承的联合	340	15.8.1 对象切片	380
14.4.1 构造函数和析构函数调用的次序	341	15.9 重载和重新定义	382
14.5 名字隐藏	343	15.9.1 变量返回类型	383
14.6 非自动继承的函数	346	15.10 虚函数和构造函数	385
14.6.1 继承和静态成员函数	349	15.10.1 构造函数调用次序	385
14.7 组合与继承的选择	349	15.10.2 虚函数在构造函数中的行为	386
14.7.1 子类型设置	350	15.11 析构函数和虚拟析构函数	386
14.7.2 私有继承	352	15.11.1 纯虚析构函数	388
14.8 protected	353	15.11.2 析构函数中的虚机制	389
14.8.1 protected继承	353	15.11.3 创建基于对象的继承	390
14.9 运算符的重载与继承	353	15.12 运算符重载	392
14.10 多重继承	355	15.13 向下类型转换	394
14.11 渐增式开发	355	15.14 小结	396
14.12 向上类型转换	356	15.15 练习	397
14.12.1 为什么要“向上类型转换”	357	第16章 模板介绍	400
14.12.2 向上类型转换和拷贝构造函数	357	16.1 容器	400
14.12.3 组合与继承（再论）	359	16.1.1 容器的需求	401
14.12.4 指针和引用的向上类型转换	360	16.2 模板综述	402
14.12.5 危机	360	16.2.1 模板方法	403
14.13 小结	361	16.3 模板语法	404
14.14 练习	361	16.3.1 非内联函数定义	405
		16.3.2 作为模板的IntStack	406

16.3.3 模板中的常量	408
16.4 作为模板的 Stash 和 Stack	409
16.4.1 模板化的指针 Stash	411
16.5 打开和关闭所有权	415
16.6 以值存放对象	417
16.7 迭代器简介	418
16.7.1 带有迭代器的栈	425
16.7.2 带有迭代器的 PStash	427
16.8 为什么使用迭代器	432
16.8.1 函数模板	434
16.9 小结	435
16.10 练习	435
附录A [⊖] 编码风格	
附录B [⊖] 编程准则	
附录C [⊖] 推荐读物	

第2卷 实用编程技术

译者序	441
前言	442

第一部分 建立稳定的系统

第1章 异常处理	448
1.1 传统的错误处理	448
1.2 抛出异常	450
1.3 捕获异常	451
1.3.1 try块	451
1.3.2 异常处理器	451
1.3.3 终止和恢复	452
1.4 异常匹配	453
1.4.1 捕获所有异常	454
1.4.2 重新抛出异常	454
1.4.3 不捕获异常	455
1.5 清理	456
1.5.1 资源管理	457
1.5.2 使所有事物都成为对象	458
1.5.3 auto_ptr	460
1.5.4 函数级的try块	461
1.6 标准异常	462

1.7 异常规格说明	464
1.7.1 更好的异常规格说明	467
1.7.2 异常规格说明和继承	467
1.7.3 什么时候不使用异常规格说明	468
1.8 异常安全	468
1.9 在编程中使用异常	471
1.9.1 什么时候避免异常	471
1.9.2 异常的典型应用	472
1.10 使用异常造成的开销	474
1.11 小结	476
1.12 练习	476
第2章 防御性编程	478
2.1 断言	480
2.2 一个简单的单元测试框架	482
2.2.1 自动测试	483
2.2.2 TestSuite框架	485
2.2.3 测试套件	488
2.2.4 测试框架的源代码	489
2.3 调试技术	493
2.3.1 用于代码跟踪的宏	494
2.3.2 跟踪文件	494
2.3.3 发现内存泄漏	495
2.4 小结	499
2.5 练习	500

第二部分 标准C++库

第3章 深入理解字符串	504
3.1 字符串的内部是什么	504
3.2 创建并初始化C++字符串	505
3.3 对字符串进行操作	508
3.3.1 追加、插入和连接字符串	508
3.3.2 替换字符串中的字符	509
3.3.3 使用非成员重载运算符连接	512
3.4 字符串的查找	513
3.4.1 反向查找	516
3.4.2 查找一组字符第1次或最后一次出现的位置	517
3.4.3 从字符串中删除字符	519
3.4.4 字符串的比较	520

⊖ 附录内容请到华章网站 (www.hzbook.com) 下载。

3.4.5 字符串和字符的特性	523	5.1.2 默认模板参数	582
3.5 字符串的应用	527	5.1.3 模板类型的模板参数	583
3.6 小结	531	5.1.4 typename关键字	587
3.7 练习	531	5.1.5 以template关键字作为提示	588
第4章 输入输出流	534	5.1.6 成员模板	589
4.1 为什么引入输入输出流	534	5.2 有关函数模板的几个问题	591
4.2 救助输入输出流	537	5.2.1 函数模板参数的类型推断	591
4.2.1 插入符和提取符	537	5.2.2 函数模板重载	594
4.2.2 通常用法	540	5.2.3 以一个已生成的函数模板地址 作为参数	595
4.2.3 按行输入	541	5.2.4 将函数应用到STL序列容器中	598
4.3 处理流错误	542	5.2.5 函数模板的半有序	600
4.4 文件输入输出流	544	5.3 模板特化	601
4.4.1 一个文件处理的例子	544	5.3.1 显式特化	601
4.4.2 打开模式	546	5.3.2 半特化	602
4.5 输入输出流缓冲	546	5.3.3 一个实例	604
4.6 在输入输出流中定位	548	5.3.4 防止模板代码膨胀	606
4.7 字符串输入输出流	550	5.4 名称查找问题	609
4.7.1 输入字符串流	551	5.4.1 模板中的名称	609
4.7.2 输出字符串流	552	5.4.2 模板和友元	613
4.8 输出流的格式化	555	5.5 模板编程中的习语	617
4.8.1 格式化标志	555	5.5.1 特征	617
4.8.2 格式化域	556	5.5.2 策略	621
4.8.3 宽度、填充和精度设置	557	5.5.3 奇特的递归模板模式	623
4.8.4 一个完整的例子	557	5.6 模板元编程	624
4.9 操纵算子	560	5.6.1 编译时编程	625
4.9.1 带参数的操纵算子	560	5.6.2 表达式模板	631
4.9.2 创建操纵算子	562	5.7 模板编译模型	636
4.9.3 效用算子	563	5.7.1 包含模型	636
4.10 输入输出流程序举例	565	5.7.2 显式实例化	637
4.10.1 维护类库的源代码	565	5.7.3 分离模型	638
4.10.2 检测编译器错误	568	5.8 小结	639
4.10.3 一个简单的数据记录器	570	5.9 练习	640
4.11 国际化	573	第6章 通用算法	642
4.11.1 宽字符流	574	6.1 概述	642
4.11.2 区域性字符流	575	6.1.1 判定函数	644
4.12 小结	577	6.1.2 流迭代器	646
4.13 练习	577	6.1.3 算法复杂性	647
第5章 深入理解模板	580	6.2 函数对象	648
5.1 模板参数	580	6.2.1 函数对象的分类	649
5.1.1 无类型模板参数	580		

6.2.2 自动创建函数对象	649	7.6 堆栈	743
6.2.3 可调整的函数对象	652	7.7 队列	745
6.2.4 更多的函数对象例子	653	7.8 优先队列	748
6.2.5 函数指针适配器	658	7.9 持有二进制位	755
6.2.6 编写自己的函数对象适配器	662	7.9.1 bitset<n>	756
6.3 STL算法目录	665	7.9.2 vector<bool>	758
6.3.1 实例创建的支持工具	666	7.10 关联式容器	760
6.3.2 填充和生成	669	7.10.1 用于关联式容器的发生器和 填充器	763
6.3.3 计数	670	7.10.2 不可思议的映像	765
6.3.4 操作序列	671	7.10.3 多重映像和重复的关键字	766
6.3.5 查找和替换	674	7.10.4 多重集合	768
6.3.6 比较范围	679	7.11 将STL容器联合使用	771
6.3.7 删除元素	681	7.12 清除容器的指针	773
6.3.8 对已排序的序列进行排序和运算	684	7.13 创建自己的容器	774
6.3.9 堆运算	691	7.14 对STL的扩充	776
6.3.10 对某一范围内的所有元素进行 运算	691	7.15 非STL容器	777
6.3.11 数值算法	697	7.16 小结	781
6.3.12 通用实用程序	699	7.17 练习	781
6.4 创建自己的STL风格算法	700	第三部分 专 题	
6.5 小结	701	第8章 运行时类型识别	785
6.6 练习	702	8.1 运行时类型转换	785
第7章 通用容器	706	8.2 typeid 操作符	789
7.1 容器和迭代器	706	8.2.1 类型转换到中间层次类型	790
7.2 概述	707	8.2.2 void型指针	791
7.2.1 字符串容器	711	8.2.3 运用带模板的RTTI	792
7.2.2 从STL容器继承	712	8.3 多重继承	793
7.3 更多迭代器	714	8.4 合理使用RTTI	793
7.3.1 可逆容器中的迭代器	715	8.5 RTTI的机制和开销	797
7.3.2 迭代器的种类	716	8.6 小结	797
7.3.3 预定义迭代器	717	8.7 练习	798
7.4 基本序列容器: vector、list和deque	721	第9章 多重继承	800
7.4.1 基本序列容器的操作	721	9.1 概论	800
7.4.2 向量	723	9.2 接口继承	801
7.4.3 双端队列	728	9.3 实现继承	803
7.4.4 序列容器间的转换	730	9.4 重复子对象	807
7.4.5 被检查的随机访问	731	9.5 虚基类	810
7.4.6 链表	732	9.6 名字查找问题	817
7.4.7 交换序列	736	9.7 避免使用多重继承	819
7.5 集合	737	9.8 扩充一个接口	820

9.9 小结	823	11.2 C++中的并发	876
9.10 练习	823	11.3 定义任务	878
第10章 设计模式	825	11.4 使用线程	879
10.1 模式的概念	825	11.4.1 创建有响应的用户界面	880
10.2 模式分类	826	11.4.2 使用执行器简化工作	882
10.3 简化习语	827	11.4.3 让步	884
10.3.1 信使	827	11.4.4 休眠	885
10.3.2 收集参数	828	11.4.5 优先权	886
10.4 单件	829	11.5 共享有限资源	887
10.5 命令：选择操作	833	11.5.1 保证对象的存在	887
10.6 消除对象耦合	836	11.5.2 不恰当地访问资源	890
10.6.1 代理模式：作为其他对象的前端	837	11.5.3 访问控制	892
10.6.2 状态模式：改变对象的行为	838	11.5.4 使用保护简化编码	893
10.7 适配器模式	840	11.5.5 线程本地存储	896
10.8 模板方法模式	841	11.6 终止任务	897
10.9 策略模式：运行时选择算法	842	11.6.1 防止输入/输出流冲突	897
10.10 职责链模式：尝试采用一系列策略模式	843	11.6.2 举例观赏植物园	898
10.11 工厂模式：封装对象的创建	845	11.6.3 阻塞时终止	901
10.11.1 多态工厂	847	11.6.4 中断	902
10.11.2 抽象工厂	849	11.7 线程间协作	906
10.11.3 虚构造函数	851	11.7.1 等待和信号	906
10.12 构建器模式：创建复杂对象	855	11.7.2 生产者-消费者关系	909
10.13 观察者模式	860	11.7.3 用队列解决线程处理的问题	912
10.13.1 “内部类”方法	862	11.7.4 广播	916
10.13.2 观察者模式举例	864	11.8 死锁	921
10.14 多重派遣	867	11.9 小结	925
10.15 小结	873	11.10 练习	926
10.16 练习	873		
第11章 并发	875		
11.1 动机	875		
		附录	
		附录A 推荐读物	
		附录B 其他	

附录^①

附录A 推荐读物

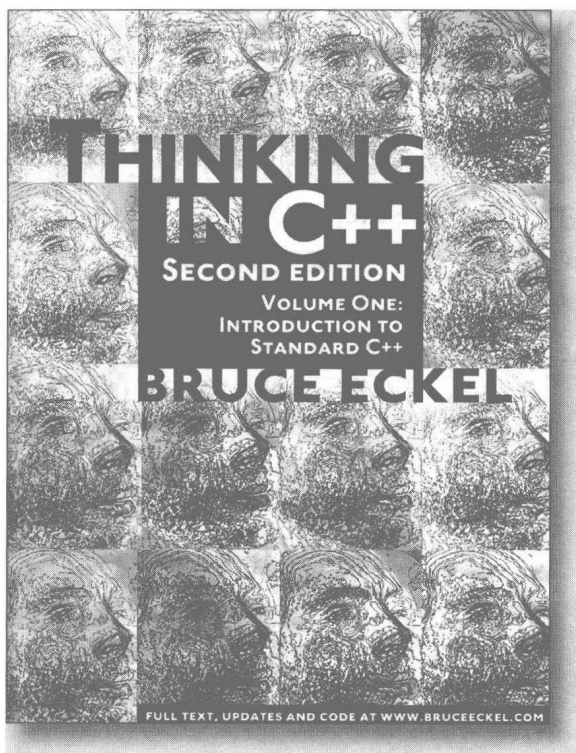
附录B 其他

第1卷 标准C++导引

(原书第2版)

Volume One: Introduction to Standard C++
Second Edition

(美) Bruce Eckel 著 刘宗田 袁兆山 潘秋菱 等译



本书为**1996**年软件开发图书**Jolt**生产力大奖得主。

“这本书是一项巨大的成就。你的书架上早就该有这本书了。讲述输入输出流的这一章是我至今见过的对这个主题最全面最易懂的叙述。”

——《**Doctor Dobbs Journal**》杂志的资深编辑**Al Stevens**

“Eckel的作品是惟一一本能如此清晰地叙述以面向对象方法构造程序的书籍。这本书也是一本优秀的C++入门指南。”

——《**Unix Review**》杂志的编辑**Andrew Binstock**

“Bruce继续用他对C++的洞察力让我惊叹。《C++编程思想》是他思想的精华荟萃。如果你需要C++中难题的清晰解答，就去买这部杰作吧。”

——《**The Tao of Objects**》一书的作者**Gary Entsminger**

“《C++编程思想》耐心而系统地探讨了C++的一些重要问题，例如内联、引用、运算符重载、继承和动态对象，还有一些更深奥的主题，例如模板的合理使用、异常和多继承等。所有这些内容，连同Eckel本人的关于对象和程序设计的观点，完美地编织为一体，成为每个C++开发者案头必备之书。如果你正在用C++开发软件，《C++编程思想》应当是你必须拥有的一本书。”

——《**PC Magazine**》的特邀编辑**Richard Hale Shaw**

作为译者，我有幸组织翻译了《C++编程思想》第1版。在这之前，我仅仅耳闻这是一本别具特色的畅销书，至于如何别具特色，如何得以畅销，并不十分清楚。在第1版的翻译过程中，我逐渐领悟了Eckel编写技巧的真谛。在第1版中文版的译者序中，我曾这样总结他的技巧：“其内容、讲授方法、选用例子和跟随的练习，别具特色。原书作者不是按传统的方法讲解C++的概念和编程方法，而是根据他自己过去学习C++的亲身体会，根据他多年教学中从他的学生们的学习中发现的问题，用一些非常简单的例子和简练的叙述，阐明了在学习C++中特别容易混淆的概念。特别是，他经常通过例子引导读者从C++编译实现的汇编代码的角度反向审视C++的语法和语义，常常使读者有‘心有灵犀一点通’的奇特效果，这在以往的C++书中并不多见。”

《C++编程思想》第1版的中文版自2000年1月第1次印刷以来，在中国市场上的畅销势头经久不衰。这充分说明了这本书在中国读者心目中的地位。

Eckel致力于计算机程序设计语言教学数十年，而且是全心全意地从事这项工作，这本身就是难能可贵的，是他成功的根本原因。另外，他的成功还有赖于他的精益求精的精神，这不仅表现在第1版的与众不同的精心选材和认真推敲的叙述方面，也体现在第2版与第1版的不同点上。

表面上，第2版与第1版并无太多的变化，但是通过分析，可以看出，其中的任何变化都是经过深思熟虑的。从章节上看，最大的区别是增加了两章和去掉了四章。增加的两章分别是“对象的创建与使用”和“C++中的C”。前者与“对象导言”实际上是第1版的“对象的演化”一章的彻底重写，增加了近几年面向对象方法和编程方法的最新研究与实践的丰硕成果。后者的添加不仅使不熟悉C的读者直接使用这本书成为可能，而且C本身就是C++的组成部分，这是C++得以成功的主要原因之一。删去的四章是“输入输出流介绍”、“多重继承”、“异常处理”和“运行时类型识别”。这四章属于C++中较复杂的主题，作者将它们连同C++标准完成后又增加的一些内容放到这本书的第2卷中。这样就使得这本书的第1卷内容更加集中，一般的读者可以不被这些复杂内容所困扰，而需要这些复杂知识的读者可以阅读这本书的第2卷。

实际上，第2版的改变不仅仅在于这些章节的调整，更多的改变体现在每一章中，包括例子的调整和练习的补充。这本书更成熟了。

受机械工业出版社华章公司计算机编辑部委托，我又承担起《C++编程思想》第2版的翻译组织任务。翻译这样的成功之作，既是机遇，更是压力。有如此众多的读者阅读我们翻译的作品，无论如何这是令人高兴的事情。诚然，吸引读者的魅力来源于原作，而不是我们的翻译技巧，但是能将如此光辉灿烂的作品变成中文版本，奉献给中国的读者，这其中毕竟融入了我们的心血，而且，第1版中文版的畅销，已经充分证明，并未因我们翻译水平的限制而黯淡了原作的光芒，这对我们已经够宽慰的了。然而，百万双眼睛在阅读这本书的同时也在审视我们的翻译水平，这就足以使我们诚惶诚恐的了。翻译在某种意义上是再创作的过程，