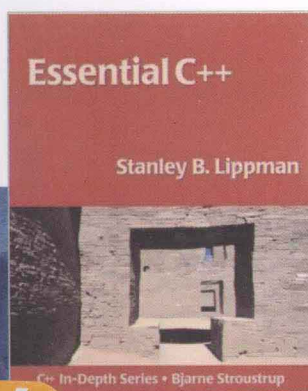


Essential C++ 中文版

[美] **Stanley B. Lippman** 著
侯捷 译



1

Essential C++



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

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

Essential C++

中文版

[美] Stanley B. Lippman 著
侯捷 译



电子工业出版社
Publishing House of Electronics Industry
北京•BEIJING

内容简介

本书以四个面向来表现 C++ 的本质: procedural (面向过程的)、generic (泛型的)、object-based (基于对象的)、object-oriented (面向对象的)。全书围绕一系列逐渐繁复的程序问题,以及用以解决这些问题的语言特性来组织。循此方式,你将不只学到 C++ 的功能和结构,也可学到它们的设计目的和基本原理。

本书适合那些已经开始从事软件设计,又抽不出太多时间学习新技术的程序员阅读。

Authorized translation from the English language edition, entitled ESSENTIAL C++, 1E, 9780201485189 by LIPPMAN, STANLEY B., published by Pearson Education, Inc., Publishing as Addison-Wesley Professional, Copyright © 2000 Pearson Education, Inc.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc.

CHINESE SIMPLIFIED language edition published by PEARSON EDUCATION ASIA LTD., and PUBLISHING HOUSE OF ELECTRONICS INDUSTRY Copyright © 2013.

本书简体中文版专有版权由 Pearson Education 培生教育出版亚洲有限公司授予电子工业出版社。未经出版者预先书面许可,不得以任何方式复制或抄袭本书的任何部分。

本书简体中文版贴有 Pearson Education 培生教育出版集团激光防伪标签,无标签者不得销售。

版权贸易合同登记号 图字: 01-2013-4126

图书在版编目 (CIP) 数据

Essential C++ 中文版 / (美) 李普曼 (Lippman, S. B.) 著; 侯捷译.——北京: 电子工业出版社, 2013.8

ISBN 978-7-121-20934-5

I. ①E... II. ①李... ②侯... III. ①C 语言—程序设计 IV. ①TP312

中国版本图书馆 CIP 数据核字 (2013) 第 150792 号

责任编辑: 白 涛

印 刷: 三河市双峰印刷装订有限公司

装 订: 三河市双峰印刷装订有限公司

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本: 787×980 1/16 印 张: 18.75 字 数: 440 千字

印 次: 2013 年 8 月第 1 次印刷

定 价: 65.00 元

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 zlts@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线: (010) 88258888。

悦读上品 得乎益友

孔子云：“取乎其上，得乎其中；取乎其中，得乎其下；取乎其下，则无所得矣”。

对于读书求知而言，这句古训教我们去读好书，最好是好书中的上品——经典书。其中，科技人员要读的技术书，因为直接关乎客观是非与生产效率，阅读选材本更应慎重。然而，随着技术图书品种的日益丰富，发现经典书越来越难，尤其对于涉世尚浅的新读者，更为不易，而他们又往往是最需要阅读、提升的重要群体。

所谓经典书，或说上品，是指选材精良、内容精练、讲述生动、外延丰盈、表现手法体贴入微的读品，它们会成为读者的知识和经验库中的重要组成部分，并且拥有从不断重读中汲取养分的空间。因此，选择阅读上品的问题便成了有效阅读的首要问题。当然，这不只是效率问题，上品促成的既是对某一种技术、思想的真正理解和掌握，同时又是一种感悟或享受，是一种愉悦。

与技术本身类似，经典 IT 技术书多来自国外。深厚的积累、良好的写作氛围，使一批大师为全球技术学习者留下了璀璨的智慧瑰宝。就在那个年代即将远去之时，无须回眸，也能感受到这一部部厚重而深邃的经典著作，在造福无数读者后从未蒙尘的熠熠光辉。而这些凝结众多当今国内技术中坚美妙记忆与绝佳体验的技术图书，虽然尚在国外图书市场上大放异彩，却已逐渐淡出国人的视线。最为遗憾的是，迟迟未有可以填补空缺的新书问世。而无可替代，不正是经典书被奉为主臬的原因？

为了不让国内读者，尤其是即将步入技术生涯的新一代读者，就此错失这些滋养过先行者们的好书，以出版 IT 精品图书，满足技术人群需求为己任的我们，愿意承担这一使命。本次机遇惠顾了我们，让我们有机会携手权威的 Pearson 公司，精心推出“传世经典书丛”。

在我们眼中，“传世经典”的价值首先在于——既适合喜爱科技图书的读者，也符合专家们挑剔的标准。幸运的是，我们的确找到了这些堪称上品的佳作。丛书带给我们的幸运颇多，细数一下吧。

得以引荐大师著作

有恐思虑不周，我们大量参考了国外权威机构和网站的评选结果，并得到了 Pearson 的专业支持，又进

一步对符合标准之图书的国内外口碑与销售情况进行细致分析,也听取了国内技术专家的宝贵建议,才有幸选出对国内读者最富有技术养分的大师上品。

向深邃的技术内涵致敬

中外技术环境存在差异,很多享誉国外的好书未必适用于国内读者;且技术与应用瞬息万变,很容易让人心生迷惘或疲于奔命。本丛书的图书遴选,注重打好思考方法与技术理念的根基,旨在帮助读者修炼内功,提升境界,将技术真正融入个人知识体系,从而可以一通百通,从容面对随时涌现的技术变化。

翻译与评注的双项选择

引进优秀外版著作,将其翻译为中文供国内读者阅读,较为有效与常见。但另有一些外语水平较高、喜好评读原版的读者,苦于对技术理解不足,不能充分体会原文表述的精妙,需要有人指导与点拨。而一批本土技术精英经过长期经典熏陶及实践锤炼,已足以胜任这一工作。有鉴于此,本丛书在翻译版的同时推出融合英文原著与中文点评、注释的评注版,供不同志趣的读者自由选择。

承蒙国内一流译(注)者的扶持

优秀的英文原著最终转化为真正的上品,尚需跨越翻译鸿沟,外版图书的翻译质量一直屡遭国内读者诟病。评注版的增值与含金量,同样依赖于评注者的高卓才具。好在,本丛书得到了久经考验的权威译(注)者的认可和支持,首肯我们选用其佳作,或亲自参与评注工作。正是他们的参与保证了经典的品质,既再次为我们的选材把关,更提供了一流的中文表述。

期望带给读者良好的阅读体验

一本好书带给人的愉悦不止于知识收获,良好的阅读感受同样不可缺少,且对学业不无助益。为了让读者收获与上品相称的体验,我们在图书装帧设计与选材用料上同样不敢轻率,惟愿送到读者手中的除了珠玑章句,还有舒适与熨帖的视觉感受。

所有参与丛书出版的人员,尽管能力有限,却无不心怀严谨之心与完美愿望。如果读者朋友能从潜心阅读这些上品中偶有获益,不啻为对我们工作的最佳褒奖。若有阅读感悟,敬请拨冗告知,以鼓励我们继续在这一道路上贡献绵薄之力。如有不周之处,也请不吝指教。

电子工业出版社博文视点

满汉全席之外 (译序/侯捷)

Stanley B. Lippman 所著的 C++ *Primer* 雄踞书坛历久不衰, 堪称 C++ 最佳教科书。但是走过十个年头之后, 继 1237 页的 C++ *Primer* 第 3 版, Lippman 又返璞归真地写了这本 276 页的 *Essential C++*。有了满汉全席, 为何还眷顾清粥小菜? 完成了伟大的巨著, 何必回头再写这么一本轻薄短小的初学者用书呢?

所有知道 Lippman 和 C++ *Primer* 的人, 脸上都浮现相同的问号。

轻薄短小并不是判断适合初学与否的依据。Lippman 写过 *Inside the C++ Object Model*, 280 页小开本, 崩掉多少 C++ 老手的牙。本书之所以号称适合初学者, 不在于轻薄短小, 在于素材选择与组织安排。

关于 Lippman 重作冯妇的故事, 他自己在前言中有详细的介绍。他的转折, 他的选择, 他的职责, 乃至于这本书的纲要和组织, 前言中都有详细的交待。这方面我不必再置一词。

身为 C++ *Primer*, 3rd Edition 的译者, 以及多本进阶书籍的作者, 我必须努力说服自己, 才能心甘情愿地将精力与时间用来重复过去的足迹。然而, 如果连 Lippman 都愿意为初学者再铺一条红地毯, 我也愿意为初学者停留一下我的脚步。

我是一名信息教育者, 写译书籍, 培训人员, 在大学开课……。我真正第一线面对大量学习者。借此机会我要表达的是, 所谓“初学者”实在是个过于笼统的名词与分类(呃, 谈得上分类吗)。一般所谓“初学者”, 多半想象是大一新生程度。其实 C++ 语言存在各种“初学者”, 有 13 岁的, 有 31 岁的(当然也有 41 岁的)。只要是第一次接触这个语言, 就是这个语言的初学者, 他可能才初次接触计算机, 可能浸淫 Pascal/C 语言十年之久, 也可能已有 Smalltalk/Java 三年经验。有人连计算机基本概念都没有, 有人已经是经验丰富的软件工程师。这些人面对 C++, 学习速度、教材需求、各人领悟, 相同吗?

大不同矣！

每个人都以自己的方式来诠释“初学者”这个字眼，并不经意地反映出自己的足迹。初学者有很多很多种，“初学者”一词却无法反映他们的真实状态。

固然，轻薄短小的书籍乍见之下让所有读者心情轻松，但如果舍弃太多应该深入的地方不谈，也难免令人行止失据，进退两难。这本小书可以是你的起点，但绝不能够是你的终点。

作为一本优秀教科书，轻薄短小不是重点，素材选择与组织安排，表达的精准与阅读的顺畅，才是重点。

作为一个好的学习者，背景不是重点，重要的是，你是否具备正确的学习态度。起步固然可从轻松小品开始，但如果碰上大部头巨著就退避三舍逃之夭夭，面对任何技术只求快餐速成，学编程语言却从来不编写程序，那就绝对没有成为高手乃至专家的一天。

有些人的学习，自练一身铜筋铁骨，可以在热带丛林中披荆斩棘，在莽莽草原中追奔逐北。有些人的学习，既未习惯大部头书，也未习惯严谨格调，更未习惯自修勤学，是温室里的一朵花，没有自立自强的本钱。

章节的安排，篇幅的份量，索引的保留，习题加解答，以及网上的服务，都使这本小书成为自修妙品、C++专业课程的适当教材。诚挚希望《Essential C++中文版》的完成，帮助更多人获得 C++ 的学习乐趣——噢，是的，OOP（面向对象编程）可以带给你很多乐趣，我不骗你 ☺

侯捷

2012/09/22

敬请注意：

1. 本书与英文版页页对译，从而得以保留原书索引。
2. 本书附加“中英术语对照表”于附录 C，并于其中说明中英术语的采用原则。

前言

Preface

天啊，这本书竟是如此轻薄短小。我真想大叫一声“哇欧”！C++ *Primer* 加上索引、扉页、谢词之后，厚达 1237 页，而此书却只有薄薄 276 页。套用拳击术语，这是一部“轻量级”作品。

每个人都会好奇这究竟是怎么回事。的确，这里头有一段故事。

过去数年来，我不断缠着迪斯尼电影动画公司（Disney Feature Animation）的每一个人，请求让我亲身参与一部电影的制作。我缠着导演，甚至 Mickey 本人（如果我可以说不出的话），要求一份管理工作。我会如此疯狂，部分原因是深陷于好莱坞大屏幕那令人神往的无尽魔力而难以自拔。除了计算机科学方面的学位，我还拥有艺术硕士的头衔，而电影工作似乎可以为我带来个人专长的某种整合。我要求管理工作，为的是从制片过程中获取经验，以便提供实际有用的工具。身为一个 C++ 编译器编写者，我一直都是自己最主要的用户之一。而你知道，当你是自己软件的主力抱怨者时，你就很难再为自己辩护或觉得受到不公平的责难。

《幻想曲 2000》（*Fantasia 2000*）片中有一段火鸟（Firebird）的特效镜头。其计算机特效指导对于我的加盟颇感兴趣。不过，为了掂掂我的斤两，他要求我先写个工具，读入为某段场景所摄的原始数据，再由此产生可嵌入 Houdini 动画套件中的摄影机节点（camera node）。当然，我用 C++ 把它顺利搞定了。他们爱死它了，我也因此得到了我梦寐以求的工作。

有一次，在制片过程中（在此特别感谢 Jinko 和 Chyuan），我被要求以 Perl 重写那个工具。其他的 TD 并非编程高手，仅仅知道 Perl、Tcl 之类的程序语言。（TD 是电影界的术语，指的是技术导演。我是这部片子的软件 TD，我们还有一位灯光 TD [你好，Mira]，一位模型 TD [你好，Tim]，以及电影特效动画师 [你好，Mike, Steve, Tonya]。）而且，喔，天啊，我得赶着点，因为我们想要获得一些观念上的实证，而导演（你好，Paul 和 Gaetan）及特效总监（你好，Dave）正等着结果，准备呈给公司大头目（你好，Peter）。这虽然不是什么紧急要务，可是，你知道的……，唉。

这令我感到些许为难。我自信可以用 C++ 快速完成，但我不懂 Perl。好吧，我想，我去找本书抱抱佛脚好了——前提是这本书不能太厚，起码此刻不能太厚。而且它最好不要告诉我太多东西，虽然我知道我应该知道每一样东西，不过暂且等等吧。毕竟这只是一场表演：导演们需要一些经过证实的概念，艺术家需要一些东西协助证实其概念，而制片（你好，heck），她需要的是一天 48 小时。此刻我不需要全世界最棒的 Perl 大全，我需要的是一本能妥善引导我前进，使我不致偏离正轨过远的小书。

我找到了 Randal Schwartz 的 *Learning Perl*，它让我立即上手并进展神速，而且颇具阅读趣味。不过，就像其他有趣的计算机书籍一样，它也略去了不少值得一读的内容——尽管在那个时间点，我并不需要了解所有内容，我只需要让我的 Perl 程序乖乖动起来。

我终于在感伤的心境中明白，C++ *Primer* 第三版其实无法扮演人们在初学 C++ 时的导师角色。它太庞大了。当然，我还是认为它是一本让我骄傲的巨著——特别是由于邀请到 Josée Lajoie 共同完成。但是，对于想立刻学会 C++ 程序语言的人来说，这本巨著实在过于庞大复杂。这正是本书的由来。

你或许会想，C++ 又不是 Perl。完全正确！本书也非 *Learning Perl*，它谈的是如何学习 C++。真正的问题在于，谁能够在散尽千页篇幅之后，犹敢自称教导了所有的东西呢？

1. **精细度。**在计算机绘图领域中，精细度指的是影像被描绘出来的鲜明程度。画面左上角那位骑在马背上的匈奴人，需要一张看得清楚眼睛的脸、头发、五点钟方向的影子、衣服……。匈奴人的背后——不，不是那块岩石，老天——唔，相较之下无关紧要。因此我们不会以相同的精细度来描绘这两个影像。同样道理，本书的精细度在相当程度上做了降低。依我看，C++ *Primer* 除了在运算符重载（operator overloading）方面的实例讨论稍嫌不足外，可说极其完备了（我敢这么说是因为 Josée 也有一份功劳）。但尽管如此，C++ *Primer* 还花了 46 页篇幅讨论操作符重载，并附上了范例，而本书却仅以两页带过。
2. **语言核心。**当我还是 C++ *Report* 的编辑时，我常说，杂志编辑有一半工作花在决定哪些题材应该放入，哪些不要。这句话对本书一样成立。本书内容是围绕在编程过程中所发生的一系列问题组织的。我介绍编程语言本身的特性，借此来为不同的问题提供解决之道。书中并未提及任何一个可由多继承或虚继承解决的问题，所以我也就完全没有讨论这两个主题。然而，为了实现 iterator class，我必须引入嵌套类型（nested type）。Class 的类型转换操作符很容易被错用，解释起来也很复杂，所以我不打算在书中提到它。诸如此类。我对题材的选择以及对语言特性的呈现顺序，欢迎大家指教批评。这是我的选择，也是我的职责。
3. **范例的数量。**C++ *Primer* 有数百页代码，巨细靡遗，其中甚至包括一套面向对象的（Object Oriented）文本检索系统，以及十个左右的完整 class。虽然本书也有代码，但数量远不及 C++ *Primer*。为了弥补这项缺憾，我将所有习题解答都置于附录 A。诚如我的编辑 Deborah Lafferty 所言，“如果你想提高教学速度，唾手可得的解答对于学习的强化极有帮助。”

结构与组织

本书由七章和两份附录构成。第 1 章借着撰写一个具有互动性质的小程序，描绘 C++ 语言预先定义的部分。这一章涵盖了内置的数据类型、语言预定义的运算符（operator）、标准库中的 `vector` 和 `string`、条件语句和循环语句、输入和输出用的 `iostream` 库。我之所以在本章介绍 `vector` 和 `string` 这两个 class，是因为我想鼓励读者多多利用它们取代语言内置的数组（array）和 C-style 字符串。

第 2 章解释函数的设计与使用，并逐一查看 C++ 函数的多种不同风貌，包括 inline 函数、重载（overloaded）函数、function template，以及函数指针（pointers to functions）。

第 3 章涵盖了所谓的 Standard Template Library（STL）：一组容器类（包括 `vector`、`list`、`set`、`map`，等等）、一组作用于容器上的泛型算法（包括 `sort()`、`copy()`、`merge()`，等等）。附录 B 按字典顺序列出了最广为运用的泛型算法，并逐一附上了使用实例。

身为一个 C++ 程序员，你的主要任务便是提交 class 以及面向对象的 class 层次体系。第 4 章将带领你亲身了解 class 机制的设计与使用过程。在这个过程中，你会看到如何为自身的应用系统建立起专属的数据类型。第 5 章介绍如何扩展 class，使多个相关的 class 形成族系，支持面向对象的 class 层次体系。以我在梦工厂动画电影公司（Dreamworks Animation）担任顾问的经验为例，那时我们设计了一些 class，用来进行四个频道影像合成之类的工作。我们使用了继承和动态绑定（dynamic binding）技术，定义影像合成所需的 class 层次体系，而不只是设计八个独立的 class。

第 6 章的重头戏是 class template，那是建立 class 时的一种先行描述，让我们得以将 class 用到的（或多个）数据类型或数据值，抽离并参数化。以 `vector` 为例，可能需要将其元素的类型加以参数化，而 `buffer` 的设计不仅得将元素类型参数化，还得将其缓冲区容量参数化。本章的行进路线围绕在二分树（binary tree）class template 的实现上。

最后一章，第 7 章，介绍如何使用 C++ 的异常处理机制（exception handling facility），并示范如何将它融入标准库所定义的异常体系中。附录 A 是本书习题解答。附录 B 提供了关于最广为运用的一些泛型算法的相关讨论与使用实例。

关于源代码

本书的所有程序，以及习题解答中的完整代码，都可从网上获得。你可以在 Addison Wesley Longman 的网站（www.awl.com/cseng/titles/0-201-48518-4）或我的个人首页（www.objectwrite.com）中取得。所有程序均在 Visual C++ 5.0 环境中以 Intel C++ 编译器测试过，也在 Visual C++ 6.0 环

境中以 Microsoft C++ 编译器测试过。你或许需要稍微修改一下代码才能在自己的系统上编译成功。如果你需要做一些修改并且做了, 请将修改结果寄一份给我 (slippman@objectwrite.com), 我会将它们附上你的大名, 附于习题解答代码中。请注意, 本书并未显现所有代码。

致谢

在这里, 我要特别感谢 C++ *Primer* 第三版的共同作者 Josée Lajoie。不仅因为她为本书初稿提供了许多深入见解, 更因为她在背后不断地给我鼓舞。我也要特别感谢 Dave Slayton 以他那犀利的绿色铅笔, 彻底审阅了文本内容与程序范例。Steve Vinoski 则以同情但坚决的口吻, 为本书初稿提供了许多宝贵意见。

特别感谢 Addison-Wesley 编辑团队: 全书编辑 Deborah Lafferty 从头到尾支持这个项目; 审稿编辑 Besty Hardinger 对本书文字的可读性贡献最大; 产品经理 John Fuller 带领我们把一堆文稿化为一本完整的图书。

撰写本书的过程中, 我同时还担任独立顾问工作, 因此必须兼顾书稿和客户。感谢我的客户对我如此体谅和宽容。我要感谢 Colin Lipworth、Edwin Leonard、Kenneth Meyer, 因为你们的耐心与信赖, 本书才得以完成。

更多读物

内举不避亲, 我要推荐 C++ 书籍中最好的两本, 那便是 Lippman 与 Lajoie 合著的 C++ *Primer*, 以及 Stroustrup 的著作 *The C++ Programming Language*。这两本书目前均为第 3 版。我会在本书各主题内提供其他更深入的参考书目。以下是本书的参考书目。(你可以在 C++ *Primer* 和 *The C++ Programming Language* 中找到更广泛的参考文献。)

[LIPPMAN98] Lippman, Stanley and Josée Lajoie, *C++ Primer, 3rd Editoin*, Addison Wesley Longman, Inc., Reading, MA (1998) ISBN 0-201-82470-1.

[LIPPMAN96a] Lippman, Stanley, *Inside the C++ Object Model*, Addison Wesley Longman, Inc., Reading, MA (1996) ISBN 0-201-83454-5.

[LIPPMAN96b] Lippman, Stanley, Editor, *C++ Gems*, a SIGS Books imprint, Cambridge University Press, Cambridge, England (1996) ISBN 0-13570581-9.

[STROUSTRUP97] Stroustrup, Bjarne, *The C++ Programming Language, 3rd Editoin*, Addison Wesley Longman, Inc., Reading, MA (1997) ISBN 0-201-88954-4.

[SUTTER99] Sutter, Herb, *Exceptional C++*, Addison Wesley Longman, Inc., Reading, MA (2000) ISBN 0-201-61562-2.

排版约定

本书字体（英文版）为 10.5 pt Palatino。代码和语言关键字为 8.5 pt `lucida`。书中出现的标识符如果后面紧接着 C++ 的 function call 运算符（也就是一对圆括号 `()`），即代表某个函数名称。因此，`foo` 代表程序中的某个 object，`bar()` 代表程序中的某个函数。各个 class 的名称以 Palatino 字形呈现。

简体中文版的排版约定是：内文中的一般英文字为 9 pt Times New Roman。代码和语言关键字为 8 pt Courier New。各个 class 的名称亦为 8 pt Courier New。异常类（exception class）以 8 pt Lucida Sans 呈现。英文长术语（例如 template parameter list, by reference, exception safe）采用 8 pt Arial。运算符名称采用 9 pt Footlight MT Light。译注均以楷体呈现。

目录

Contents

满汉全席之外（译序/侯捷）	v
前言 Preface	xi
结构与组织	xiii
关于源代码	xiii
致谢	xiv
更多读物	xiv
排版约定	xv
第 1 章 C++编程基础 Basic C++ Programming	1
1.1 如何撰写 C++程序	1
1.2 对象的定义与初始化	7
1.3 撰写表达式	10
1.4 条件语句和循环语句	15
1.5 如何运用 Array 和 Vector	22
1.6 指针带来弹性	26
1.7 文件的读写	30
第 2 章 面向过程的编程风格 Procedural Programming	35
2.1 如何编写函数	35
2.2 调用函数	41
2.3 提供默认参数值	50
2.4 使用局部静态对象	53
2.5 声明 inline 函数	55

2.6 提供重载函数.....	56
2.7 定义并使用模板函数.....	58
2.8 函数指针带来更大的弹性.....	60
2.9 设定头文件.....	63
第 3 章 泛型编程风格 Generic Programming	67
3.1 指针的算术运算.....	68
3.2 了解 Iterator (泛型指针)	73
3.3 所有容器的共通操作.....	76
3.4 使用顺序性容器.....	77
3.5 使用泛型算法.....	81
3.6 如何设计一个泛型算法.....	83
3.7 使用 Map	90
3.8 使用 Set.....	91
3.9 如何使用 Iterator Inserter.....	93
3.10 使用 iostream Iterator	95
第 4 章 基于对象的编程风格 Object-Based Programming	99
4.1 如何实现一个 Class	100
4.2 什么是构造函数和析构函数.....	104
4.3 何谓 mutable (可变) 和 const (不变)	109
4.4 什么是 this 指针.....	113
4.5 静态类成员.....	115
4.6 打造一个 Iterator Class	118
4.7 合作关系必须建立在友谊的基础上.....	123
4.8 实现一个 copy assignment operator.....	125
4.9 实现一个 function object.....	126
4.10 重载 iostream 运算符.....	128
4.11 指针, 指向 Class Member Function.....	130
第 5 章 面向对象编程风格 Object-Oriented Programming	135
5.1 面向对象编程概念.....	135
5.2 漫游: 面向对象编程思维.....	138
5.3 不带继承的多态.....	142

5.4	定义一个抽象基类.....	145
5.5	定义一个派生类.....	148
5.6	运用继承体系.....	155
5.7	基类应该多么抽象.....	157
5.8	初始化、析构、复制.....	158
5.9	在派生类中定义一个虚函数.....	160
5.10	运行时的类型鉴定机制.....	164
第 6 章 以 template 进行编程 Programming with Templates		167
6.1	被参数化的类型.....	169
6.2	Class Template 的定义	171
6.3	Template 类型参数的处理.....	172
6.4	实现一个 Class Template	174
6.5	一个以 Function Template 完成的 Output 运算符.....	180
6.6	常量表达式与默认参数值.....	181
6.7	以 Template 参数作为一种设计策略	185
6.8	Member Template Function	187
第 7 章 异常处理 Exception Handling		191
7.1	抛出异常.....	191
7.2	捕获异常.....	193
7.3	提炼异常.....	194
7.4	局部资源管理.....	198
7.5	标准异常.....	200
附录 A 习题解答 Exercises Solutions.....		205
附录 B 泛型算法参考手册 Generic Algorithms Handbook		255
附录 C 中英术语对照 侯捷		271
	英文术语的采用原则.....	271
	中英术语对照（按字母顺序排列）	272
索引 Index.....		277

C++ 编程基础

Basic C++ Programming

本章，我们将从一个小程序开始，通过它来练习 C++ 程序语言的基本组成。其中包括：

1. 一些基础数据类型：布尔值 (Boolean)、字符 (character)、整数 (integer)、浮点数 (floating point)。
2. 算术运算符、关系运算符以及逻辑运算符，用以操作上述基础数据类型。这些运算符不仅包括一般常见的加法运算符、相等运算符 (==)、小于等于 (<=) 运算符以及赋值 (assignment, =) 运算符，也包含比较特殊的递增 (++) 运算符、条件运算符 (?:)，以及复合赋值 (+= 等) 运算符。
3. 条件分支和循环控制语句，例如 if 语句和 while 循环，可用来改变程序的控制流程。
4. 一些复合类型，例如指针及数组。指针可以让我们间接参考一个已存在的对象，数组则用来定义一组具有相同数据类型的元素。
5. 一套标准的、通用的抽象化库，例如字符串和向量 (vector)。

1.1 如何撰写 C++ 程序

How to Write a C++ Program

此刻，假设我们需要撰写一个简易程序，必须能够将一段信息送至用户的终端 (terminal)。信息的内容则是要求用户输入自己的名字。然后程序必须读取用户所输入的名字，将这个名字储存起来，以便后续操作使用。最后，送出一个信息，以指名道姓的方式向用户打招呼。

那么，该从何处着手呢？每个 C++ 程序都是从一个名为 main 的函数开始执行，我们就从这个地方着手吧！main 是个由用户自行撰写的函数，其通用形式如下：

```
int main()
{
    // 我们的程序代码置于此处
}
```

`int` 是 C++ 程序语言的关键字。所谓关键字 (keyword)，就是程序语言预先定义的一些具有特殊意义的名称。`int` 用来表示语言内置的整数数据类型。(下一节我将针对数据类型做更详细的说明。)

函数 (function) 是一块独立的程序代码序列 (code sequence)，能够执行一些运算。它包含四个部分：返回值类型 (return type)、函数名称、参数列表 (parameter list)，以及函数体 (function body)。下面依次简要介绍每一部分。

函数的返回值通常用来表示运算结果。`main()` 函数返回整数类型。`main()` 的返回值用来告诉调用者，这个程序是否正确执行。习惯上，程序执行无误时我们令 `main()` 返回零。若返回一个非零值，表示程序在执行过程中发生了错误。

函数的名称由程序员选定。函数名最好能够提供某些信息，让我们容易了解函数实际上在做些什么。举例来说，`min()` 和 `sort()` 便是极佳的命名。`f()` 和 `g()` 就没有那么好了。为什么？因为后两个名称相形之下无法告诉我们函数的实际执行操作。

`main` 并非程序语言定义的关键字。但是，执行我们这个 C++ 程序的编译系统，会假设程序中定义有 `main()` 函数。如果我们没有定义，程序将无法执行。

函数的参数列表 (parameter list) 由两个括号括住，置于函数名之后。空的参数列表，如 `main()`，表示函数不接受任何参数。

参数列表用来表示“函数执行时，调用者可以传给函数的类型列表”。列表之中以逗号隔开各个类型（通常会说用户“调用 (call 或是 *invoke*)”某个函数)。举例来说，如果我们编写 `min()` 函数，使其返回两数中较小者，那么它的参数列表应该注明两个即将被拿来比较的数值的类型。这样一个用来比较两整数值的 `min()` 函数，可能会以如下形式定义：

```
int min(int val1, int val2)
{
    // 程序代码置于此处
}
```

函数的主体 (body) 由大括号 ({}) 标出，其中含有“提供此函数之运算”的程序代码。双斜线 (//) 表示该行内容为注释，也就是程序员对程序代码所做的某些说明。注释的撰写是为了便于阅读者更容易理解程序。编译过程中，注释会被忽略掉。双斜线之后直至行末的所有内容，都会被当作程序注释。

我们的第一件工作就是要将信息送至用户终端。数据的输入与输出，并非 C++ 程序语言本身定义的一部分（此精神同 C 语言，见 K&R 第 7 章），而是由 C++ 的一套面向对象的类层次体系 (classes hierarchy) 提供支持，并作为 C++ 标准库 (standard library) 的一员。

所谓类 (class)，是用户自定义的数据类型 (user-defined data type)。class 机制让我们得以将数据类型加入我们的程序中，并有能力识别它们。面向对象的类层次体系 (class hierarchy) 定义了整个家