



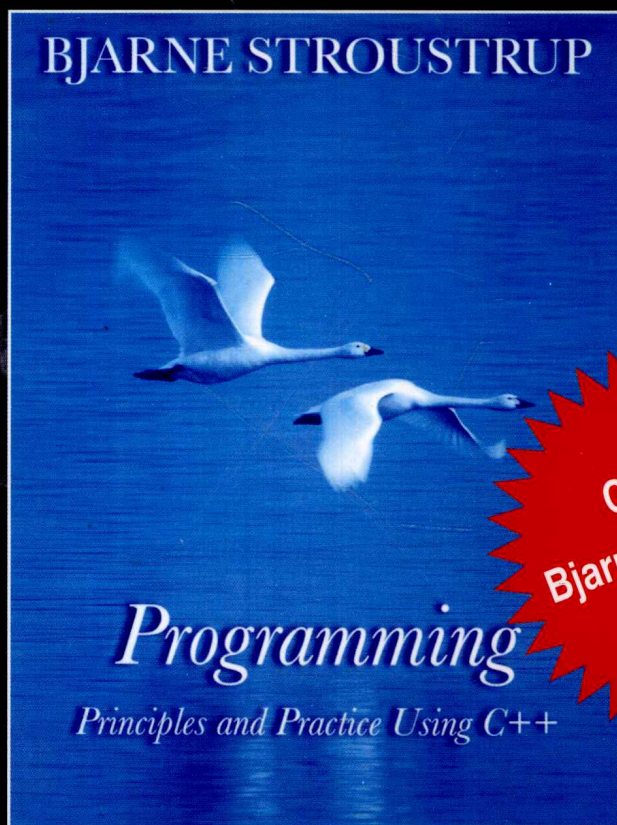
HZ BOOKS  
华章教育

PEARSON

计 算 机 科 学 丛 书

# C++程序设计原理与实践

(美) Bjarne Stroustrup 著 王刚 刘晓光 吴英 李涛 译



C++之父  
Bjarne Stroustrup  
的最新力作

**Programming**  
Principles and Practice Using C++



机械工业出版社  
China Machine Press

本书是 C++ 之父 Bjarne Stroustrup 的最新力作。书中广泛地介绍了程序设计的基本概念和技术,包括类型系统、算术运算、控制结构、错误处理等;介绍了从键盘和文件获取数值和文本数据的方法以及以图形化方式表示数值数据、文本和几何图形;介绍了 C++ 标准库中的容器(如向量、列表、映射)和算法(如排序、查找和内积)的设计和使用。同时还对 C++ 思想和历史进行了详细的讨论,很好地拓宽了读者的视野。

本书语言通俗易懂、实例丰富,可作为大学计算机、电子工程、信息科学等相关专业的教材,也可供相关专业人员参考。

Simplified Chinese edition copyright © 2010 by Pearson Education Asia Limited and China Machine Press.

Original English language title: *Programming: Principles and Practice Using C++* (ISBN 978-0-321-54372-1) by Bjarne Stroustrup, Copyright © 2009.

All rights reserved.

Published by arrangement with the original publisher, Pearson Education, Inc., publishing as Addison-Wesley.

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:01-2009-1608

### 图书在版编目(CIP)数据

C++ 程序设计原理与实践/(美)斯特劳斯特鲁普(Stroustrup, B.)著;王刚等译. —北京:机械工业出版社,2010.6

(计算机科学丛书)

书名原文: *Programming: Principles and Practice Using C++*

ISBN 978-7-111-30322-0

I. C… II. ①斯… ②王… III. C 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字(2010)第 061970 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:李俊竹

北京京北印刷有限公司印刷

2010 年 6 月第 1 版第 1 次印刷

184mm × 260mm 41.75 印张

标准书号: ISBN 978-7-111-30322-0

定价:108.00 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

## 出版者的话

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，不仅肇划了研究的范畴，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, Brian 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](http://www.hzbook.com)

电子邮件：[hzjsj@hzbook.com](mailto: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



华章科技图书出版中心

## 译者序

程序设计是打开计算机世界大门的金钥匙，它使五彩斑斓的软件对你来说不再是“魔术”。C++ 语言则是学习掌握这把金钥匙的有力武器，它优美、高效，从大洋深处到火星表面，从系统核心到高层应用，从掌中的手机到超级计算机，到处都有 C++ 程序的身影。本书适合那些从未有过程序设计经验的初学者，如果你愿意努力学习，本书能帮助你理解使用 C++ 语言进行程序设计的基本原理及大量实践技巧。你所学到的思想，大多数也都可直接用于其他程序设计语言。本书不是初学程序设计语言的简单入门教材，它的目标是能让读者学到基本的实用程序设计技术，因此也可以作为程序设计方面的“第二本书”。基于这样一个目标，注重实践是本书的明显特点。它希望教会你编写真正能被他人所使用的“有用的程序”，而非“玩具程序”。因此，除了基本的 C++ 程序特性之外，本书还介绍了大量的求解实际问题的程序设计技术：如语法分析器程序的设计、图形化程序设计、利用正则表达式处理文本、数值计算程序设计以及嵌入式程序设计等。在其他大多数程序设计入门书籍中，是找不到这些内容的，像调试技术、测试技术等其他程序设计书籍着墨不多的话题，本书也有详细的介绍。程序设计远非遵循语法规则和阅读手册那么简单，而在于理解基本思想、原理和技术，并进行大量实践。本书阐述了这一理念，为读者指引了明确的方向，教会读者如何才能达到编写有用的、优美的程序这一最终目标。

本书的作者 Bjarne Stroustrup 是 C++ 语言的设计者和最初的实现者，也是《The C++ Programming Language》(Addison-Wesley 出版社)一书的作者。他现在德州农工大学计算机科学首席教授，美国国家工程院的会员和 AT&T 院士。在进入学术界之前，他在 AT&T 贝尔实验室工作多年。他是 ISO C++ 标准委员会的创始人之一。本书是他在 C++ 程序设计领域奉献给广大读者的又一经典著作。

本书分为五个部分。第一部分介绍基本的 C++ 程序设计知识，包括第一个“Hello, World!”程序，对象、类型和值，运算，错误处理，函数，类等内容，以及一个计算器程序实例。第二部分介绍输入和输出，首先介绍了输入/输出流的基本概念和格式化输出方法，然后第 12~16 章重点介绍了图形/GUI 类和图形化程序设计。第三部分介绍数据结构和算法，重点介绍了向量、自由内存空间、数组、模板和异常、容器和迭代器以及算法和映射。第四部分希望拓宽读者的视野，介绍了程序设计语言理念和历史、文本处理技术、数值计算、嵌入式程序设计技术及测试技术，此外还较为详细地介绍了 C 语言与 C++ 的异同。第五部分为附录，包括 C++ 语言概要、标准库概要、Visual Studio 简要入门、FLTK 安装以及 GUI 实现等内容。

本书的序、第 0 章、8~11 章、23~25 章、27 章、附录、术语表由王刚翻译，第 4、5、22、26 章由刘晓光翻译，第 1~3 章、17~21 章由吴英翻译，第 6~7 章、12~16 章由李涛翻译。翻译大师经典，难度超乎想象。接受任务之初，诚惶诚恐；翻译过程中，如履薄冰；完成后，忐忑不安。虽然竭尽全力，但肯定还有很多错漏之处，敬请读者批评指正。

译者

2010 年 4 月于南开大学

# 前 言

“该死的鱼雷！全速前进。”

——海军上将 Farragut

程序设计是这样一门艺术，它将问题求解方案描述成计算机可以执行的形式。程序设计中很多工作都花费在寻找求解方案以及对其求精上。通常，只有在真正编写程序求解一个问题的过程中才会对问题本身理解透彻。

本书适合于那些从未有过编程经验但愿意努力学习程序设计的初学者，它能帮助你理解使用 C++ 语言进行程序设计的基本原理并获得实践技巧。我的目标是使你获得足够多的知识和经验，以便能使用最新最好的技术进行简单有用的编程工作。达到这一目标需要多长时间呢？作为大学一年级课程的一部分，你可以在一个学期内完成这本书的学习（假定你有另外四门中等难度的课程）。如果你是自学的话，不要期望能花费更少的时间完成学习（一般来说，每周 15 个小时，共 14 周是合适的学时安排）。

三个月可能看起来是一段很长的时间，但要学习的内容很多，写第一个简单程序之前，就要花费一个小时。而且，所有学习过程都是渐进的：每一章都会介绍一些新的有用的概念，并通过从实际应用中获取的例子来阐述这些概念。随着学习进程的推进，你通过程序代码表达思想的能力——也就是让计算机按你的期望工作的能力，会逐渐稳步地提高。我从不会说：“先学习一个月的理论知识，然后看看你是否能使用这些理论吧。”

为什么要学习程序设计呢？因为计算机文化是建立在软件之上的。如果不理解软件，那么你将退化到只能相信“魔术”的境地，并且将被排除在很多最为有趣、最具经济效益和社会效益的领域之外。当谈论程序设计时，我所想到的是整个计算机程序家族，从带有 GUI（图形用户界面）的个人计算机程序，到工程计算和嵌入式系统控制程序（如数码相机、汽车和手机中的程序），以及文字处理程序等，在很多日常应用和商业应用中都能看到这些程序。程序设计与数学有些相似，如果认真去做的话，它会是一种非常有用的智力训练，可以锻炼我们的思考能力。然而，由于计算机能做出反馈，程序设计又不像大多数数学形式那么抽象，因而对更多人来说更容易接受。可以说，程序设计是一条能够打开你的眼界，将世界变得更美好的途径。最后，程序设计非常有趣。

为什么学习 C++ 这门程序设计语言呢？学习程序设计不可能不借助一门程序设计语言，而 C++ 直接支持现实世界中的软件所使用的那些关键概念和技术。C++ 是使用最为广泛的程序设计语言之一，其应用领域几乎没有局限。从大洋深处到火星表面，到处都能发现 C++ 程序的身影。C++ 是由一个开放的国际标准组织全面考量、精心设计的。在任何一种计算机平台上都能找到高质量的和免费的 C++ 实现。而且，你用 C++ 所学到的程序设计思想，大多数都可直接用于其他程序设计语言，如 C、C#、Fortran 以及 Java。最后一个原因，我喜欢 C++ 适合编写优美、高效的代码这一特点。

本书不是初学程序设计的简单入门教材，我写此书的用意也不在此。我为本书设定的目标

是：能让你学到基本的实用编程技术的最简单的书籍。这是一个雄心勃勃的目标，因为很多现代软件所依赖的技术，不过才出现短短几年时间。

我的基本假设是，你希望编写供他人使用的程序，并愿意认真负责地、较高质量地完成这个工作；也就是说，我假定你希望达到专业水准。因此，我为本书选择的主题覆盖了开始学习实用编程技术所需要的内容，而不只是那些容易讲授和容易学习的内容。如果某种技术是你做好基本编程工作所需要的，那么本书就会介绍它，同时展示用以支持这种技术的编程思想和语言工具，并提供相应的练习，期望你通过做这些练习来熟悉这种技术。但如果你只想了解“玩具程序”，那么你能学到的将远比我所提供的少得多。另一方面，我不会用一些实用性很低的内容来浪费你的时间，本书介绍的内容都是你在实践中几乎肯定会用到的。

如果你只是希望直接使用别人编写的程序，而不想了解其内部原理，也不想亲自向代码中加入重要的内容，那么本书不适合你。请考虑是否采用另一本书或另一种程序设计语言会更好些。如果这大概就是你对程序设计的看法，那么请同时考虑一下你从何得来的这种观点，它真的满足你的需求吗？人们常常低估程序设计的复杂程度和它的重要性。我不愿看到你不喜欢程序设计，只是因为你的需求与我所描述的部分软件之间不匹配。信息技术世界中还有很多部分是不要求程序设计知识的，那些领域可能适合你。本书面向的是那些确实希望编写和理解复杂计算机程序的人。

考虑到本书的结构和注重实践的特点，它也可以作为程序设计方面的第二本书，适合那些已经了解一点 C++ 的人，和那些会用其他语言编程，现在想学习 C++ 的人。如果你属于其中一类，我不好估计你学习这本书要花费多长时间。但我可以给你的建议是，多做练习。因为你在学习中常见的一个问题是习惯用熟悉的、旧的方式编写程序，而不是在适当的地方采用新技术，多做练习会帮助你解决这个问题。如果你曾经按某种更为传统的方式学习过 C++，那么在进行到第 7 章之前，你会发现一些令你惊奇的和有用的内容。除非你的名字是 Stroustrup，否则你会发现我在本书中所讨论的内容不是“你父辈的 C++”。

学习程序设计要靠编程实践。在这一点上，程序设计与其他需要实践学习的技能是相似的。你不可能仅仅通过读书就学会游泳、演奏乐器或者开车，你必须进行实践。同样，不读程序、不写程序就不可能学会程序设计。本书给出了大量代码实例，都配合有说明文字和图表。你需要通过读这些代码来理解程序设计的思想、概念和原理，并掌握用来表达这些思想、概念和原理的程序设计语言的特性。但有一点很重要，仅仅读代码是不能学会编程实践技巧的。为此，你必须进行编程练习，通过编程工具熟悉编写、编译和运行程序。你需要亲身体验编程中会出现的错误，学习如何修改它们。总之，在学习程序设计的过程中，编写代码的练习是不可替代的。而且，这也是乐趣所在！

另一方面，程序设计远非只是遵循一些语法规则和阅读手册那么简单。本书的重点不在于 C++ 的语法，而在于理解基础思想、原理和技术，这是一名好程序员所必备的。只有设计良好的代码才有机会成为一个正确、可靠和易维护的系统的一部分。而且，“基础”意味着延续性：当现在的程序设计语言和工具演变甚至被取代后，这些基础知识仍会保持其重要性。

那么计算机科学、软件工程、信息技术等又如何呢？它们都属于程序设计范畴吗？当然不是！但程序设计是一门基础性的学科，是所有计算机相关领域的基础，在计算机科学领域占有重要的地位。本书对算法、数据结构、用户接口、数据处理和软件工程等领域的重要概念和技术进行了简要介绍。但本书不能取代对这些领域全面、均衡的学习。

代码可以很有用，同样也可以很优美。本书会帮你了解优美的代码意味着什么，并帮你掌握构造优美代码的原理和实践技巧。祝你学习顺利！

## 致学生

到目前为止，我在德州农工大学已经用本书的初稿教过 1000 名以上的大一新生，其中 60% 曾经有过编程经历，而剩余 40% 从未见过哪怕一行代码。大多数学生的学习是成功的，所以你也可以成功。

你不一定是在某门课程中来学习本书，我认为本书会广泛用于自学。然而，不管你学习本书是作为课程的一部分还是自学，都要尽量与他人协作。程序设计有一个不好的名声——它是一种个人活动，这是不公正的。大多数人在作为一个有共同目标的团体的一份子时，工作效果更好，学习得更快。与朋友一起学习和讨论问题不是作弊！而是取得进步最有效，同时也是最快乐的途径。如果没有特殊情况的话，与朋友一起工作会促使你表达出你的思想，这正是测试你对问题理解和确认你的记忆的最有效的方法。你没有必要独自解决所有编程语言和编程环境中的难题。但是，请不要自欺欺人，不去完成那些简单练习和大量的习题（即使没有老师督促你，你也不应这样做）。记住，程序设计（尤其）是一种实践技能，需要通过实践来掌握。如果你不编写代码（完成每章的若干习题），那么阅读本书就纯粹是一种无意义的理论学习。

大多数学生，特别是那些爱思考的好学生，有时会对自己的努力工作是否值得产生疑问。当（不是如果）你产生这样的疑问时，休息一会儿，重新阅读这篇前言，阅读一下第 1 章（“计算机、人和程序设计”）和第 22 章（“思想和历史”）。在那里，我试图阐述我在程序设计中发现了哪些令人兴奋的东西，以及为什么我会认为程序设计是能为世界带来积极贡献的重要工具。如果你对我的教学理念和一般方法有疑问，请阅读第 0 章（“致读者”）。

你可能会对本书的厚度感到担心。本书如此之厚的一部分原因是，我宁愿反复重复一些解释说明或增加一些实例，而不是让你自己到处找这些内容，这应该令你安心。另外一个主要原因是，本书的后半部分是一些参考资料和补充资料，供你想要深入了解程序设计的某个特定领域（如嵌入式系统程序设计、文本分析或数值计算）时查阅。

还有，学习中请耐心些。学习任何一种重要的、有价值的新技能都要花费一些时间，而这是值得的。

## 致教师

本书不是一门传统的计算机科学的 101 课程，而是一本关于如何构造能实际工作的软件的书。因此本书省略了很多计算机科学系学生按惯例要学习的内容（图灵完全、状态机、离散数学、乔姆斯基文法等）。硬件相关的内容也省略了，因为我假定学生从幼儿园时代就已经通过不同途径使用过计算机了。本书也不准备涉及一些计算机科学领域最重要的主题。本书是关于程序设计的（或者更一般地，是关于如何开发软件的），因此关注的是少量主题的更深入的细节，而不是像传统计算机课程那样讨论很多主题。本书试图只做好一件事，计算机科学不是一门课程可以囊括的。如果本书（本课程）被计算机科学、计算机工程、电子工程（很多我们最早的学生都是电子专业的）、信息科学或者其他相关专业所采用，我希望这门课程能和其他一些课程一起进行，共同形成对计算机科学的完整介绍。

请阅读第 0 章，那里有对我的教学理念、一般教学方法等的介绍。请在教学过程中尝试将这

些观点传达给你的学生。

## 资源

本书网站的网址为 [www.stroustrup.com/Programming](http://www.stroustrup.com/Programming)，其中包含了各种使用本书讲授和学习程序设计所需的辅助资料。这些资料可能会随着时间推移不断改进，但对于初学者，现在可以找到下面一些资料：

- 基于本书的讲义的幻灯片。
- 一本教师指南。
- 本书中使用的库的头文件和实现。
- 本书中实例的代码。
- 某些习题的解答。
- 可能有用的一些链接。
- 勘误表。

欢迎随时提出对这些资料的改进意见。

## 致谢

我要特别感谢我已故的同事和联合导师 Lawrence “Pete” Peterson，很久以前，在我还未感受到教授初学者的惬意时，是他鼓励我承担这项工作，并提供了很多能令课程成功的教学经验。没有他，这门课程的首次尝试就会失败。他参与了这门课程最初的建设，本书就是为这门课程所著。他还和我一起反复讲授这门课程，汲取经验，不断改进课程和本书。在本书中我使用的“我们”这个字眼，最初的意思就是指“Pete 和我”。

我要感谢那些直接或间接帮助过我撰写本书的学生、助教以及德州农工大学讲授 ENGR 112 课程的教师，以及 Walter Daugherty，他曾讲授过这门课程。还要感谢 Damian Dechev、Tracy Hammond、Arne Tolstrup Madsen、Gabriel Dos Reis、Nicholas Stroustrup、J. C. van Winkel、Greg Versoonder、Ronnie Ward 和 Leor Zolman，他们对本书初稿提出了一些建设性意见。感谢 Mogens Hansen 为我解释引擎控制软件。感谢 Al Aho、Stephen Edwards、Brian Kernighan 和 Daisy Nguyen，他们帮助我在夏天躲开那些分心的事来完成本书。

感谢 Addison-Wesley 公司为我安排的审阅人：Richard Enbody、David Gustafson、Ron McCarty 和 K. Narayanaswamy，他们基于自身讲授 C++ 课程或者大学计算机科学系 101 课程的经验，对本书提出了宝贵的意见。还要感谢我的编辑 Peter Gordon 为本书提出的很多有价值的意见以及他极大的耐心。我非常感谢 Addison-Wesley 公司为本书组织的制作团队的同仁，他们为本书的高质量出版做出了很多贡献，他们是：Julie Grady（校对）、Chris Keane（排版）、Rob Mauhar（插图）、Julie Nahil（制作编辑）和 Barbara Wood（文字编辑）。

另外，我本人对本书代码的检查很不系统，Bashar Anabtawi、Yinan Fan 和 Yuriy Solodkyy 使用微软 C++ 7.1 版（2003）和 8.0 版（2005）以及 GCC 3.4.4 版检查了所有代码片段。

我还要感谢 Brian Kernighan 和 Doug McIlroy 为程序设计类书籍的撰写定下了非常高的标准，以及 Dennis Ritchie 和 Kristen Nygaard 为实用编程语言设计提供的非常有价值的经验。



# 目 录

## 第一部分 基本知识

出版者的话  
译者序  
前言

第0章 致读者	1	第2章 Hello, World!	25
0.1 本书结构	1	2.1 程序	25
0.1.1 一般方法	2	2.2 经典的第一个程序	26
0.1.2 简单练习、习题等	2	2.3 编译	27
0.1.3 进阶学习	3	2.4 链接	29
0.2 讲授和学习本书的方法	4	2.5 编程环境	30
0.2.1 本书内容顺序的安排	6	第3章 对象、类型和值	34
0.2.2 程序设计和程序设计语言	7	3.1 输入	34
0.2.3 可移植性	7	3.2 变量	35
0.3 程序设计和计算机科学	8	3.3 输入和类型	36
0.4 创造性和问题求解	8	3.4 运算和运算符	37
0.5 反馈方法	8	3.5 赋值和初始化	39
0.6 参考文献	8	3.5.1 实例：删除重复单词	41
0.7 作者简介	9	3.6 组合赋值运算符	42
第1章 计算机、人与程序设计	11	3.6.1 实例：重复单词统计	42
1.1 介绍	11	3.7 命名	43
1.2 软件	11	3.8 类型和对象	44
1.3 人	13	3.9 类型安全	45
1.4 计算机科学	15	3.9.1 安全类型转换	46
1.5 计算机已无处不在	15	3.9.2 不安全类型转换	46
1.5.1 有屏幕和没有屏幕	16	第4章 计算	51
1.5.2 船舶	16	4.1 计算	51
1.5.3 电信	17	4.2 目标和工具	52
1.5.4 医疗	18	4.3 表达式	53
1.5.5 信息领域	19	4.3.1 常量表达式	54
1.5.6 一种垂直的视角	20	4.3.2 运算符	55
1.5.7 与C++程序设计有何联系	21	4.3.3 类型转换	56
1.6 程序员的理想	21	4.4 语句	56
		4.4.1 选择语句	57
		4.4.2 循环语句	61
		4.5 函数	64

4.5.1	使用函数的原因	65	6.3.3	实现单词	106
4.5.2	函数声明	66	6.3.4	使用单词	107
4.6	向量	67	6.3.5	重新开始	108
4.6.1	向量空间增长	67	6.4	文法	109
4.6.2	一个数值计算的例子	68	6.4.1	英文文法	112
4.6.3	一个文本处理的例子	70	6.4.2	设计一个文法	113
4.7	语言特性	71	6.5	将文法转换为程序	114
第5章	错误	75	6.5.1	实现文法规则	114
5.1	介绍	75	6.5.2	表达式	115
5.2	错误的来源	76	6.5.3	项	117
5.3	编译时错误	77	6.5.4	基本表达式	118
5.3.1	语法错误	77	6.6	试验第一个版本	119
5.3.2	类型错误	77	6.7	试验第二个版本	122
5.3.3	警告	78	6.8	单词流	123
5.4	连接时错误	78	6.8.1	实现 Token_stream	124
5.5	运行时错误	79	6.8.2	读单词	125
5.5.1	调用者处理错误	80	6.8.3	读数值	126
5.5.2	被调用者处理错误	81	6.9	程序结构	127
5.5.3	报告错误	82	第7章	完成一个程序	131
5.6	异常	83	7.1	介绍	131
5.6.1	错误参数	83	7.2	输入和输出	131
5.6.2	范围错误	84	7.3	错误处理	133
5.6.3	输入错误	85	7.4	处理负数	135
5.6.4	截断错误	87	7.5	模运算: %	136
5.7	逻辑错误	88	7.6	清理代码	138
5.8	估计	89	7.6.1	符号常量	138
5.9	调试	90	7.6.2	使用函数	139
5.9.1	实用调试技术	91	7.6.3	代码格式	140
5.10	前置条件和后置条件	94	7.6.4	注释	141
5.10.1	后置条件	95	7.7	错误恢复	143
5.11	测试	96	7.8	变量	145
第6章	编写一个程序	100	7.8.1	变量和定义	145
6.1	一个问题	100	7.8.2	引入单词 name	148
6.2	对问题的思考	100	7.8.3	预定义名字	150
6.2.1	程序设计的几个阶段	101	7.8.4	我们到达目的地了吗	150
6.2.2	策略	101	第8章	函数相关的技术细节	153
6.3	回到计算器问题	102	8.1	技术细节	153
6.3.1	第一步尝试	103	8.2	声明和定义	154
6.3.2	单词	104	8.2.1	声明的类别	156

## 第二部分 输入和输出

8.2.2 变量和常量声明 .....	157	第 10 章 输入/输出流 .....	207
8.2.3 默认初始化 .....	158	10.1 输入和输出 .....	207
8.3 头文件 .....	158	10.2 I/O 流模型 .....	208
8.4 作用域 .....	160	10.3 文件 .....	209
8.5 函数调用和返回 .....	163	10.4 打开文件 .....	210
8.5.1 声明参数和返回类型 .....	163	10.5 读写文件 .....	211
8.5.2 返回一个值 .....	164	10.6 I/O 错误处理 .....	213
8.5.3 传值参数 .....	165	10.7 读取单个值 .....	215
8.5.4 传常量引用参数 .....	166	10.7.1 将程序分解为易管理的 子模块 .....	216
8.5.5 传引用参数 .....	168	10.7.2 将人机对话从函数中分离 .....	218
8.5.6 传值与传引用的对比 .....	169	10.8 用户自定义输出操作符 .....	219
8.5.7 参数检查和转换 .....	171	10.9 用户自定义输入操作符 .....	220
8.5.8 实现函数调用 .....	172	10.10 一个标准的输入循环 .....	220
8.6 求值顺序 .....	175	10.11 读取结构化的文件 .....	222
8.6.1 表达式求值 .....	176	10.11.1 内存表示 .....	222
8.6.2 全局初始化 .....	176	10.11.2 读取结构化的值 .....	224
8.7 名字空间 .....	177	10.11.3 改变表示方法 .....	226
8.7.1 using 声明和 using 指令 .....	178	第 11 章 定制输入/输出 .....	230
第 9 章 类相关的技术细节 .....	183	11.1 有规律的和无规律的输入和输出 .....	230
9.1 用户自定义类型 .....	183	11.2 格式化输出 .....	230
9.2 类和成员 .....	184	11.2.1 输出整数 .....	231
9.3 接口和实现 .....	184	11.2.2 输入整数 .....	232
9.4 演化一个类 .....	185	11.2.3 输出浮点数 .....	232
9.4.1 结构和函数 .....	185	11.2.4 精度 .....	233
9.4.2 成员函数和构造函数 .....	187	11.2.5 域 .....	234
9.4.3 保持细节私有性 .....	188	11.3 文件打开和定位 .....	235
9.4.4 定义成员函数 .....	189	11.3.1 文件打开模式 .....	235
9.4.5 引用当前对象 .....	191	11.3.2 二进制文件 .....	236
9.4.6 报告错误 .....	191	11.3.3 在文件中定位 .....	238
9.5 枚举类型 .....	192	11.4 字符流 .....	238
9.6 运算符重载 .....	193	11.5 面向行的输入 .....	239
9.7 类接口 .....	195	11.6 字符分类 .....	240
9.7.1 参数类型 .....	195	11.7 使用非标准分隔符 .....	241
9.7.2 拷贝 .....	197	11.8 还有很多未讨论的内容 .....	246
9.7.3 默认构造函数 .....	197	第 12 章 一个显示模型 .....	249
9.7.4 const 成员函数 .....	199	12.1 为什么要使用图形用户界面 .....	249
9.7.5 类成员和“辅助函数” .....	200		
9.8 Date 类 .....	201		

12.2 一个显示模型 .....	250	14.1.2 操作 .....	289
12.3 第一个例子 .....	250	14.1.3 命名 .....	290
12.4 使用 GUI 库 .....	252	14.1.4 可变性 .....	291
12.5 坐标系 .....	253	14.2 Shape 类 .....	291
12.6 形状 .....	253	14.2.1 一个抽象类 .....	292
12.7 使用形状类 .....	254	14.2.2 访问控制 .....	293
12.7.1 图形头文件和主函数 .....	254	14.2.3 绘制形状 .....	295
12.7.2 一个几乎空白的窗口 .....	255	14.2.4 拷贝和可变性 .....	297
12.7.3 坐标轴 .....	256	14.3 基类和派生类 .....	298
12.7.4 绘制函数图 .....	257	14.3.1 对象布局 .....	299
12.7.5 Polygon .....	257	14.3.2 类的派生和虚函数定义 .....	300
12.7.6 Rectangle .....	258	14.3.3 覆盖 .....	301
12.7.7 填充 .....	259	14.3.4 访问 .....	302
12.7.8 文本 .....	259	14.3.5 纯虚函数 .....	302
12.7.9 图片 .....	259	14.4 面向对象程序设计的好处 .....	303
12.7.10 还有很多未讨论的内容 .....	260	第 15 章 绘制函数图和数据图 .....	307
12.8 让图形程序运行起来 .....	261	15.1 介绍 .....	307
12.8.1 源文件 .....	261	15.2 绘制简单函数图 .....	307
第 13 章 图形类 .....	264	15.3 Function 类 .....	309
13.1 图形类概览 .....	264	15.3.1 默认参数 .....	310
13.2 Point 和 Line .....	265	15.3.2 更多的例子 .....	311
13.3 Lines .....	267	15.4 Axis 类 .....	311
13.4 Color .....	268	15.5 近似 .....	313
13.5 Line_style .....	270	15.6 绘制数据图 .....	316
13.6 Open_polyline .....	271	15.6.1 读取文件 .....	317
13.7 Closed_polyline .....	271	15.6.2 一般布局 .....	318
13.8 Polygon .....	272	15.6.3 数据比例 .....	319
13.9 Rectangle .....	273	15.6.4 构造数据图 .....	319
13.10 管理未命名对象 .....	276	第 16 章 图形用户界面 .....	324
13.11 Text .....	277	16.1 用户界面的选择 .....	324
13.12 Circle .....	278	16.2 “Next”按钮 .....	325
13.13 Ellipse .....	279	16.3 一个简单的窗口 .....	325
13.14 Marked_polyline .....	280	16.3.1 回调函数 .....	327
13.15 Marks .....	281	16.3.2 等待循环 .....	328
13.16 Mark .....	282	16.4 Button 和其他 Widget .....	329
13.17 Image .....	283	16.4.1 Widget .....	329
第 14 章 设计图形类 .....	288	16.4.2 Button .....	330
14.1 设计原则 .....	288	16.4.3 In_box 和 Out_box .....	330
14.1.1 类型 .....	288	16.4.4 Menu .....	331

16.5 一个实例 .....	332	18.3 必要的操作 .....	374
16.6 控制流的反转 .....	334	18.3.1 显示构造函数 .....	375
16.7 添加菜单 .....	335	18.3.2 调试构造函数与析构函数 .....	376
16.8 调试 GUI 代码 .....	338	18.4 访问向量元素 .....	377
<b>第三部分 数据结构和算法</b>		18.4.1 对 const 对象重载运算符 .....	378
第 17 章 向量和自由空间 .....	343	18.5 数组 .....	379
17.1 介绍 .....	343	18.5.1 指向数组元素的指针 .....	380
17.2 向量的基本知识 .....	344	18.5.2 指针和数组 .....	381
17.3 内存、地址和指针 .....	345	18.5.3 数组初始化 .....	383
17.3.1 运算符 sizeof .....	347	18.5.4 指针问题 .....	383
17.4 自由空间和指针 .....	347	18.6 实例：回文 .....	385
17.4.1 自由空间分配 .....	348	18.6.1 使用 string 实现回文 .....	386
17.4.2 通过指针访问数据 .....	349	18.6.2 使用数组实现回文 .....	386
17.4.3 指针范围 .....	349	18.6.3 使用指针实现回文 .....	387
17.4.4 初始化 .....	350	第 19 章 向量、模板和异常 .....	391
17.4.5 空指针 .....	351	19.1 问题 .....	391
17.4.6 自由空间释放 .....	351	19.2 改变向量大小 .....	393
17.5 析构函数 .....	353	19.2.1 方法描述 .....	393
17.5.1 生成的析构函数 .....	354	19.2.2 reserve 和 capacity .....	394
17.5.2 析构函数和自由空间 .....	355	19.2.3 resize .....	394
17.6 访问向量元素 .....	356	19.2.4 push_back .....	395
17.7 指向类对象的指针 .....	356	19.2.5 赋值 .....	395
17.8 类型混用：无类型指针和指针 类型转换 .....	357	19.2.6 到现在为止我们设计的 vector 类 .....	397
17.9 指针和引用 .....	359	19.3 模板 .....	397
17.9.1 指针参数和引用参数 .....	359	19.3.1 类型作为模板参数 .....	398
17.9.2 指针、引用和继承 .....	360	19.3.2 泛型编程 .....	399
17.9.3 实例：列表 .....	360	19.3.3 容器和继承 .....	401
17.9.4 列表的操作 .....	362	19.3.4 整数作为模板参数 .....	402
17.9.5 列表的使用 .....	363	19.3.5 模板参数推导 .....	403
17.10 this 指针 .....	364	19.3.6 一般化 vector 类 .....	403
17.10.1 关于 Link 使用的更多讨论 .....	365	19.4 范围检查和异常 .....	405
第 18 章 向量和数组 .....	369	19.4.1 附加讨论：设计上的考虑 .....	406
18.1 介绍 .....	369	19.4.2 使用宏 .....	407
18.2 拷贝 .....	369	19.5 资源和异常 .....	408
18.2.1 拷贝构造函数 .....	370	19.5.1 潜在的资源管理问题 .....	409
18.2.2 拷贝赋值 .....	372	19.5.2 资源获取即初始化 .....	410
18.2.3 拷贝术语 .....	373	19.5.3 保证 .....	411
		19.5.4 auto_ptr .....	412

19.5.5	vector 类的 RAII .....	412	21.6.3	另一个 map 实例 .....	458
第 20 章	容器和迭代器 .....	417	21.6.4	unordered_map .....	459
20.1	存储和处理数据 .....	417	21.6.5	集合 .....	461
20.1.1	处理数据 .....	417	21.7	拷贝操作 .....	462
20.1.2	一般化代码 .....	418	21.7.1	拷贝 .....	462
20.2	STL 建议 .....	420	21.7.2	流迭代器 .....	462
20.3	序列和迭代器 .....	423	21.7.3	使用集合保持顺序 .....	464
20.3.1	回到实例 .....	424	21.7.4	copy_if .....	464
20.4	链表 .....	425	21.8	排序和搜索 .....	465
20.4.1	列表操作 .....	426			
20.4.2	迭代 .....	427	<b>第四部分 拓宽视野</b>		
20.5	再次一般化 vector .....	428	第 22 章	理念和历史 .....	471
20.6	实例：一个简单的文本编辑器 .....	429	22.1	历史、理念和专业水平 .....	471
20.6.1	处理行 .....	431	22.1.1	程序设计语言的目标和哲学 .....	471
20.6.2	迭代 .....	431	22.1.2	编程理念 .....	473
20.7	vector、list 和 string .....	434	22.1.3	风格/范型 .....	477
20.7.1	insert 和 erase .....	435	22.2	程序设计语言历史概览 .....	479
20.8	调整 vector 类达到 STL 版本 的功能 .....	436	22.2.1	最早的程序语言 .....	480
20.9	调整内置数组达到 STL 版本 的功能 .....	438	22.2.2	现代程序设计语言的起源 .....	481
20.10	容器概览 .....	439	22.2.3	Algol 家族 .....	485
20.10.1	迭代器类别 .....	440	22.2.4	Simula .....	490
第 21 章	算法和映射 .....	444	22.2.5	C .....	491
21.1	标准库中的算法 .....	444	22.2.6	C++ .....	493
21.2	最简单的算法：find() .....	444	22.2.7	今天的程序设计语言 .....	495
21.2.1	一些一般的应用 .....	446	22.2.8	参考资源 .....	496
21.3	通用搜索算法：find_if() .....	447	第 23 章	文本处理 .....	499
21.4	函数对象 .....	448	23.1	文本 .....	499
21.4.1	函数对象的抽象视图 .....	449	23.2	字符串 .....	499
21.4.2	类成员上的谓词 .....	450	23.3	I/O 流 .....	502
21.5	数值算法 .....	450	23.4	映射 .....	503
21.5.1	累积 .....	451	23.4.1	实现细节 .....	507
21.5.2	一般化 accumulate() .....	452	23.5	一个问题 .....	508
21.5.3	内积 .....	453	23.6	正则表达式的思想 .....	510
21.5.4	一般化 inner_product() .....	453	23.7	用正则表达式进行搜索 .....	511
21.6	关联容器 .....	454	23.8	正则表达式语法 .....	513
21.6.1	映射 .....	454	23.8.1	字符和特殊字符 .....	514
21.6.2	map 概览 .....	456	23.8.2	字符集 .....	514
			23.8.3	重复 .....	515
			23.8.4	子模式 .....	516

23.8.5 可选项 .....	516	25.4.3 解决方案: 接口类 .....	561
23.8.6 字符集和范围 .....	516	25.4.4 继承和容器 .....	564
23.8.7 正则表达式错误 .....	518	25.5 位、字节和字 .....	566
23.9 与正则表达式进行模式匹配 .....	519	25.5.1 位和位运算 .....	566
23.10 参考文献 .....	522	25.5.2 bitset .....	569
第24章 数值计算 .....	525	25.5.3 有符号数和无符号数 .....	570
24.1 介绍 .....	525	25.5.4 位运算 .....	573
24.2 大小、精度和溢出 .....	525	25.5.5 位域 .....	574
24.2.1 数值限制 .....	527	25.5.6 实例: 简单加密 .....	575
24.3 数组 .....	528	25.6 编码规范 .....	579
24.4 C风格的多维数组 .....	528	25.6.1 编码规范应该是怎样的 .....	579
24.5 Matrix库 .....	529	25.6.2 编码原则实例 .....	580
24.5.1 矩阵的维和矩阵访问 .....	530	25.6.3 实际编码规范 .....	584
24.5.2 一维矩阵 .....	532	第26章 测试 .....	589
24.5.3 二维矩阵 .....	534	26.1 我们想要什么 .....	589
24.5.4 矩阵I/O .....	536	26.1.1 说明 .....	590
24.5.5 三维矩阵 .....	536	26.2 程序正确性证明 .....	590
24.6 实例: 求解线性方程组 .....	537	26.3 测试 .....	590
24.6.1 经典的高斯消去法 .....	538	26.3.1 回归测试 .....	591
24.6.2 选取主元 .....	539	26.3.2 单元测试 .....	591
24.6.3 测试 .....	539	26.3.3 算法和非算法 .....	596
24.7 随机数 .....	540	26.3.4 系统测试 .....	601
24.8 标准数学函数 .....	541	26.3.5 测试类 .....	604
24.9 复数 .....	542	26.3.6 寻找不成立的假设 .....	606
24.10 参考文献 .....	543	26.4 测试方案设计 .....	607
第25章 嵌入式系统程序设计 .....	547	26.5 调试 .....	607
25.1 嵌入式系统 .....	547	26.6 性能 .....	607
25.2 基本概念 .....	549	26.6.1 计时 .....	609
25.2.1 可预测性 .....	551	26.7 参考文献 .....	610
25.2.2 理想 .....	551	第27章 C语言 .....	613
25.2.3 生活在故障中 .....	552	27.1 C和C++: 兄弟 .....	613
25.3 内存管理 .....	553	27.1.1 C/C++兼容性 .....	614
25.3.1 动态内存分配存在的问题 .....	554	27.1.2 C不支持的C++特性 .....	615
25.3.2 动态内存分配的替代方法 .....	556	27.1.3 C标准库 .....	616
25.3.3 存储池实例 .....	557	27.2 函数 .....	617
25.3.4 栈实例 .....	557	27.2.1 不支持函数名重载 .....	617
25.4 地址、指针和数组 .....	558	27.2.2 函数参数类型检查 .....	618
25.4.1 未经检查的类型转换 .....	559	27.2.3 函数定义 .....	619
25.4.2 一个问题: 不正常的接口 .....	559	27.2.4 C++调用C和C调用C++ .....	620

27.2.5 函数指针 .....	621	27.6.2 输入 .....	632
27.3 小的语言差异 .....	622	27.6.3 文件 .....	633
27.3.1 结构标签名字空间 .....	622	27.7 常量和宏 .....	633
27.3.2 关键字 .....	623	27.8 宏 .....	634
27.3.3 定义 .....	623	27.8.1 类函数宏 .....	635
27.3.4 C风格类型转换 .....	624	27.8.2 语法宏 .....	636
27.3.5 void* 的转换 .....	625	27.8.3 条件编译 .....	636
27.3.6 枚举 .....	626	27.9 实例: 侵入式容器 .....	637
27.3.7 名字空间 .....	626	术语表 .....	644
27.4 动态内存分配 .....	626	参考书目 .....	648
27.5 C风格字符串 .....	628		
27.5.1 C风格字符串和 const .....	629		
27.5.2 字节操作 .....	630		
27.5.3 实例: strcpy() .....	630		
27.5.4 一个风格问题 .....	630		
27.6 输入/输出: stdio .....	631		
27.6.1 输出 .....	631		

## 第五部分 附 录<sup>⊖</sup>

附录 A C++ 语言概要
附录 B 标准库概要
附录 C Visual Studio 简要入门教程
附录 D 安装 FLTK
附录 E GUI 实现



# 第0章 致读者

“当实际地形与地图不符时，相信实际地形。”

——瑞士军队谚语

本章汇集了多种信息，目的是使你对我书剩余部分的内容有初步了解。你可以略过本章，直接阅读后面你感兴趣的部分。对教师来说，可以立即发现很多有用的内容。如果没有一个好的老师指导你学习本书，请不要试图阅读并理解本章的所有内容，只要阅读“本书结构”一节和“讲授和学习本书的方法”一节的第一部分即可。当你已经能自如编写和执行小程序时，可能需要回过头来重读本章。

## 0.1 本书结构

本书由四个部分和若干个附录组成：

- 第一部分：基本知识，介绍了程序设计的基本概念和技术，以及开始编写代码需要了解的一些 C++ 语言和库的知识。这部分包括类型系统、算术运算、控制结构、错误处理，以及函数和用户自定义类型的设计、实现和使用等内容。
- 第二部分：输入/输出，介绍了如何从键盘和文件获取数值和文本数据，以及如何生成相应的输出到屏幕和文件。然后介绍了如何以图形化方式表示数值数据、文本和几何图形，以及如何从图形用户界面 (graphical user interface, GUI) 获取输入数据。
- 第三部分：数据结构和算法，关注 C++ 标准库中的容器和算法框架 (标准模板库, standard template library, STL)。展示了容器 (如向量、列表和映射) 是如何 (用指针、数组、动态内存、异常和模板) 实现的以及如何使用它们。还展示了标准库算法 (如排序、查找和内积) 如何设计及使用。
- 第四部分：拓宽视野，通过对 C++ 思想和历史的讨论，通过一些实例 (如矩阵运算、文本处理、测试以及嵌入式系统程序设计)，以及通过 C 语言的一个简单描述，为我们呈现了程序设计的一个全景。
- 第五部分：附录，提供了一些不合作为教学但很有用的内容，如 C++ 语言和标准库的概要介绍，以及集成开发环境 (integrated development environment, IDE) 和图形用户界面库 (GUI 库) 的入门简介等。

不幸的是，现实世界中的程序设计并不能真正分为完全独立的四个部分。因此，这种划分仅仅是对本书内容的一种粗略分类。我们认为这是一种有用的分类方法 (这是显然的，否则我们不会采用它)，但现实情况往往与这种简洁的分类法相悖。例如，我们很快就会用到输入操作，但对 C++ 标准 I/O 流 (input/output stream, 输入/输出流) 的完整介绍却出现在本书比较靠后的部分。书中有些地方在提出某个概念时需要先介绍另外一些内容，而这与全书的布局不符，对此，我们会在此处简明介绍这些内容，以便更好地提出概念，而不是仅仅指出这些内容的完整介绍在书中什么地方。刻板的分类法更适合于手册而不是教材。

本书内容的顺序是由程序设计技术决定的，而不是程序设计语言特性，参见 0.2 节。附录 A 是按语言特性组织的。