

高等教育质量工程信息技术系列示范教材

# 新概念

# C++程序设计 大学教程

(第3版)

张基温 编著



清华大学出版社

内容简介

高等教育质量工程信息技术系列示范教材

# 新概念 C++程序设计大学教程 (第3版)

张基温 编著

清华大学出版社  
北京

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社

清华大学出版社  
北京

## 内 容 简 介

本书是一本面向大学计算机专业的 C++ 程序设计教材, 以面向对象程序设计为主线, 突出 C++ 的基本特点, 介绍 C++1y 的重要新特性。全书共分为 4 篇 11 个单元。

第 1 篇: C++ 面向对象起步。用 4 个单元帮助初学者建立面向对象的问题分析思维, 掌握相关方法和语法知识, 树立面向对象程序中“一切皆对象, 一切来自类”的意识, 初步领略面向对象程序设计的奥妙。

第 2 篇: C++ 面向抽象程序设计。用两个单元介绍 C++ 的继承机制, 并帮助读者理解如何在一个程序中组织类, 以及什么样的类结构才是好的程序结构。

第 3 篇: C++ 泛型程序设计。用两个单元介绍多态性和 STL。C++ 的泛型的通用、灵活的特点将给读者的学习带来一定乐趣, 也为读者将来从事程序开发工作提供了更多便捷方法。

第 4 篇: C++ 深入编程。用 3 个单元介绍 C++ 实体访问、函数和 I/O 流等方面的细节和内容, 进一步丰富程序设计语言机制, 使读者在程序开发上能够锦上添花。

本书理念先进、概念清晰、讲解透彻、便于理解。书中例题经典、习题丰富、覆盖面广, 适合作为高等学校各专业的面向对象程序设计教材。本书还可供培训机构使用, 也可供相关领域人员自学。

本书封面贴有清华大学出版社防伪标签, 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

新概念 C++ 程序设计大学教程 / 张基温编著. —3 版. —北京: 清华大学出版社, 2018

(高等教育质量工程信息技术系列示范教材)

ISBN 978-7-302-48154-6

I. ①新… II. ①张… III. ①C 语言-程序设计-高等学校-教材 IV. ①TP312.8

中国版本图书馆 CIP 数据核字 (2017) 第 207474 号

责任编辑: 白立军 王冰飞

封面设计: 常雪影

责任校对: 梁毅

责任印制: 李红英

出版发行: 清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址: 北京清华大学学研大厦 A 座 邮 编: 100084

社 总 机: 010-62770175 邮 购: 010-62786544

投稿与读者服务: 010-62776969, [c-service@tup.tsinghua.edu.cn](mailto:c-service@tup.tsinghua.edu.cn)

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

课 件 下 载: <http://www.tup.com.cn>, 010-62795954

印 装 者: 清华大学印刷厂

经 销: 全国新华书店

开 本: 185mm×260mm

印 张: 26.5

字 数: 629 千字

版 次: 2013 年 3 月第 1 版

2018 年 1 月第 3 版

印 次: 2018 年 1 月第 1 次印刷

印 数: 1~1500

定 价: 49.80 元

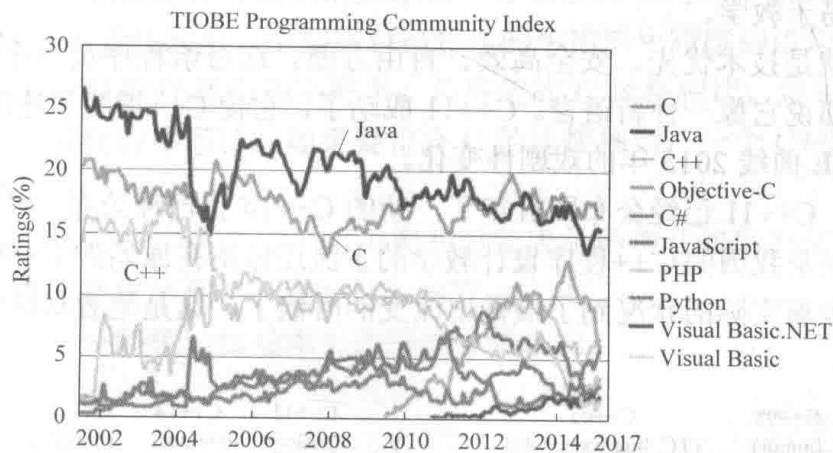
产品编号: 075145-01

# 前言

## (一)

1979年，Bjarne Stroustrup（C++之父）正在准备他的博士毕业论文，他有机会使用一种称为 Simula 的语言。顾名思义，Simula 语言主要用于仿真。其 Simula 67 版被公认是首款支持面向对象的语言。Stroustrup 发现面向对象的思想对于软件开发非常有用，但是因 Simula 语言执行效率低，其实用性不强。于是他决定自行开发一种面向对象的语言，这就是今日的 C++。

笔者一直关心 TIOBE 社区的程序设计语言排行榜，因为它能为开发和教学人员提供一份程序设计语言的行情变化资料。图 1 为 2002 年到 2017 年 2 月之间主要 TIOBE 程序设计语言排行的变化情况。在这个排行榜上发生了戏剧性变化的程序设计语言就是 C++。其第一次戏剧性的变化发生在 2004 年，在这一年中它的市场份额急剧下滑。但在之后的 10 年间基本稳定，一直保持在第三位。本书的第 1 版就是在这样的情况下编写的。其第二次戏剧性变化是在本书的第 1 版出版之后，它先在 2014 年间急剧下跌，又在 2015 年奇迹般地回归。



Source: www.nobe.com

图 1 2002 年到 2017 年 2 月 TIOBE 程序设计语言排行变化情况

C++的这些变化似乎有些莫名其妙，但认真地分析一下，这些变化还是可以解释的：其一是其他新兴语言（主要是 C#和 Object-C）对于市场份额的分割；其二则是其自身标准变化的影响。下面主要分析第二方面的因素。

C++是 Bjarne Stroustrup 于 1979 年准备一个项目时着手开发的一种程序设计语言，1985 年被市场化。C++标准委员会于 1998 年 11 月推出了其第一个 ISO 标准（俗称 C++98），2003 年推出其 ISO 标准第 2 版（俗称 C++03），C++11 则是从 2005 年就开始提交，到 2011 年 8

月才发布的 C++ 标准（俗称 C++11，提交时称为 C++0x）。从图 1 可以看出，每个标准出台到影响其市场份额有一个窗口期，这是该标准投石问路的过程。

C++03 是 C++98 的修正版，其初衷是修正 C++98 的一些不足。但是由于 C++ 脱胎于 C，遵循着 C 是 C++ 子集的原则，同时 Bjarne Stroustrup 坚持要保持其“适合教学”及既支持面向过程又支持面向对象的多泛型特色，成就了其概念清晰、设计严密、功能强大、效率较高的优点，但也带来过于复杂（如指针）、标准库功能不足，被人称为有精英化倾向的语言。因此它比较受教育界欢迎，而程序员觉得难用。不过，在通过 C++03 标准之前，人们还没有认识到这些问题，反而降低了效率，加剧了其缺陷的影响，使其在 2004 年遭受到第一次强力冲击。

2004—2005 年间的滑铁卢之惨使 C++ 的设计者和标准制定者开始清醒起来，将指导思想修订如下。

- (1) 维持与 C++98，可能的话还有与 C 语言之间的兼容性与稳定性。
- (2) 尽可能通过标准程序库来引进新的特性，而不是扩展核心语言。
- (3) 能够促进编程技术的变更优先。
- (4) 改进 C++ 以帮助系统和程序库的设计，而不是引进只对特定应用有用的新特性。
- (5) 增强类型安全，给现行不安全的技术提供更安全的替代方案。
- (6) 增强直接与硬件协同工作的性能和能力。
- (7) 为现实世界中的问题提供适当的解决方案。
- (8) 实行零负担原则(如果某些功能要求额外支持，那么只有在该功能被用到时这些额外的支持才被用到)。
- (9) 使 C++ 易于教学。

简单地说，就是技术优先、安全高效、自由方便。这带给程序员一个全新的面貌，以至于连 C++ 之父都说它像一种新语言。C++11 成功了，它使 C++ 摆脱了连续 10 年的步步下降，造就了 TIOBE 曲线 2015 年的戏剧性变化。

如图 2 所示，C++11 已经公布四五年了，新的 C++14 也已经公布，C++17 也在紧锣密鼓地部署之中。但是我国的 C++ 程序设计教学的主流还停留在原始的 C++98 甚至更旧的版本上。如此严重脱离实际的状况到了该做出改变的时候了。这是笔者对这本 C++ 教材进行改编的动因之一。

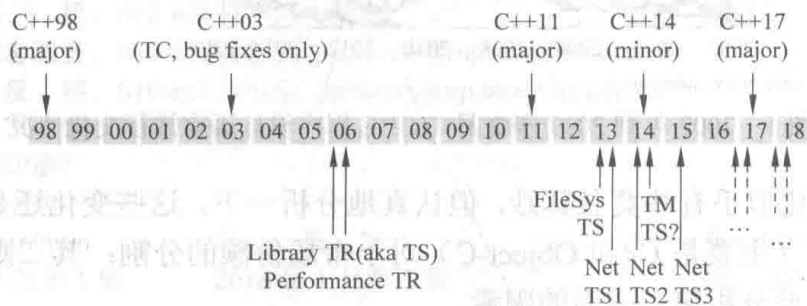


图 2 近期 C++ 标准修订步伐

不过，世界万物都有惯性。要一下子全部改到 C++11 上，很多人还是难以接受的，并且笔者自己也还在学习消化之中。在本书中，仅仅给出了 C++11 的部分新特点的接口，先

让大家了解一下这些性能。在适当的时候，再做较全面和深入的介绍。

## (二)

撇开 C++11 和 C++14 不谈，仅从一般性来讲，目前的 C++ 教学也是不尽如人意的，把 C++ 当作 C 的教学模式还广泛存在。在本书的第 1 版中虽然做了不少努力，但还不够。

Bjarne Stroustrup 曾经感慨地说：“我不是使用支持工具进行巧妙设计的信徒，但是我强烈支持系统地使用数据抽象、面向对象编程和类属编程。不拥有支持库和模板，不进行事先的总体设计，而是埋头写下一页页的代码，这是在浪费时间，这是在给维护增加困难。”他还认为，“一个人对 C 了解得越深，在写 C++ 程序时就越难避免 C 的风格，并会因此丢掉 C++ 的某些潜在优势。”为此，他提出了以下几个相关的要点，在这些情况下做同样的事情时，在 C++ 中存在比 C 中更好的处理方式。

(1) 在 C++ 中几乎不需要用宏。用 `const` 或 `enum` 定义显式常量，用 `inline` 避免函数调用的额外开销，用 `template` 去刻画一簇函数或者类型，用 `namespace` 去避免名字冲突。

(2) 不用在需要变量之前去声明它，以保证立即对其进行初始化。声明可以出现在能出现语句的所有位置上，可以出现在 `for` 语句的初始化部分，也可以出现在条件中。

(3) 不要用 `malloc()`，`new` 运算符能将同样的事情做得更好。对于 `realloc()`，试一试 `vector()`。

(4) 试着去避免 `void*`、指针算术、联合和强制，除了在某些函数或类实现的深层之外。在大部分情况下，强制都是设计错误的指示器。如果必须使用某个显式的类型转换，设法用一个“新的强制”，设法写出一个描述自己想做的事情的更精确的语句。

(5) 尽量少用数组和 C 风格的字符串。与传统的 C 风格相比，使用 C++ 标准库 `string` 和 `vector` 常常可以简化程序设计。如果要符合 C 的连接规则，一个 C++ 函数就必须被声明为具有 C 连接的。

最重要的是，要将程序考虑为一组由类和对象表示的相互作用的概念，而不是一堆数据结构和一些去拨弄数据结构中二进制位的函数。

探索如何彰显 C++ 特色，也是本书改编的重要动因。

## (三)

本次修改，将全书划分为 4 篇。

第 1 篇：C++ 面向对象起步。用 4 个单元帮助初学者建立面向对象的分析问题的思维，掌握相关方法和相关知识，树立面向对象程序设计中“一切皆对象，一切来自类”的意识。

第 2 篇：C++ 面向抽象程序设计。用两个单元帮助读者理解如何在一个程序中组织类，以及什么样的类结构才是好的程序结构。

第 3 篇：C++ 泛型程序设计。用两个单元介绍多态性和 STL。C++ 的泛型、通用、灵活的特点给读者的学习带来了一定乐趣，也为读者将来从事程序开发工作提供了更多便捷

方法。

第4篇：C++深入编程。用3个单元介绍C++在名字和实体、常量、函数、I/O流等几个方面的细节，让读者在程序开发上能够做到锦上添花。进一步提升读者“程序设计=计算思维+语言艺术”的观念。

此外，本书每个单元都围绕一个主题展开，部分单元还增添了“知识链接”部分，其目的是引申基本内容，或为以后的学习作一些铺垫。

本书的结构体系安排是考虑了以下几个因素和写作思想的结果。

- (1) B.S 的建议。
- (2) 重要先学，特色优先。
- (3) 思维开路，语法补充。
- (4) 多层次教学需要。

需要说明一点：本书给出的许多示例，虽然有用，但主要用于说明一种语法概念或给出一种编程思路，还不是精益求精的实用程序。

## (四)

由 J.Piaget、O.Kernberg、R.J.Sternberg、D.Katz、Vogotsky 等人创建的建构主义(constructivism)学习理论认为，知识不是通过教师传授得到的，而是学习者在一定的情境即社会文化背景下借助其他人(包括教师和学习伙伴)的帮助，利用必要的学习资料，通过意义建构的方式而获得的。在信息时代，人们获得知识的途径发生了根本性的变化，教师不再是单一的“传道、授业、解惑”者，帮助学习者构建一个良好的学习环境也成为其一个重要职责。当然，这也是现代教材的责任。本书充分考虑了这些问题。

为了给读者创造一个良好的学习环境，本书的部分单元设置了“知识链接”栏目。这个栏目的设置可以帮助学习者对C++的机制作更深入的了解，也为精力充沛、不够消化者提供一点“加餐”。

除了正文外，在每个单元后面都安排了“概念辨析”“代码分析”“开发实践”和“探索验证”4种自测和训练实践环节，从而建立起一个全面的学习环境。

(1) 概念辨析主要提供选择和判断两类自测题目，帮助学习者理解本单元学习过的有关概念，把当前学习内容所反映的事物尽量和自己已经知道的事物相联系，并认真思考这种联系，通过“自我协商”与“相互协商”，形成新知识的同化与顺应。

(2) 代码分析。代码阅读是程序设计者所应掌握的基本能力之一。代码分析部分的主要题型是通过阅读程序找出错误或给出程序执行结果。

(3) 开发实践。提高程序开发能力是本书的主要目标。本书在绝大多数单元后面都给出了相应的作业题目。但是，完成这些题目并非只是简单地写出其代码，而要将其看作一个“思维+语法+方法”的工程训练。因此，要求每道题的作业都要以文档的形式完成。文档中应包括以下内容。

- ① 问题分析与建模。

② 源代码设计。

③ 测试用例设计。

④ 程序运行结果分析。

⑤ 编程心得（包括运行中出现的问题与解决方法、对于测试用例的分析、对于运行结果的分析等）。

⑥ 文档的排版也要遵循统一的格式。

（4）探索验证。建构主义提倡，学习者要用探索法和发现法去建构知识的意义。学习者要在意义建构的过程中主动地收集和分析有关的信息资料，对碰到的问题提出各种假设，并努力加以验证。

## （五）

笔者从事程序设计教学近 30 多年。这 30 多年是在不断探索中走过来的。从 20 世纪 80 年代末，笔者就开始探索程序设计课程从语法体系到问题驱动的改革；到了 20 世纪 90 年代中期又在此基础上考虑让学生在学习程序设计的同时掌握程序测试技能；2003 年开始考虑如何改变学习了 C++ 而设计出的程序却是面向过程的状况。每个阶段的探索都反映在自己不同时期的相关作品中。本书则是自我认识又一次深化的表达。

在这 30 多年的探索中，笔者越来越感觉到编写教材的责任很大、困难很多。要编写一本好的教材，不仅需要对本课程涉及内容有深刻的了解，还要熟悉相关领域的知识，特别是要不断探讨贯穿其中的教学理念和教育思想。所以，越到后来，就越感到自己知识和能力的不足。可是，作为一项历史性任务的研究，笔者又不愿意将之半途而废，只能硬着头皮写下去。每一次任务的完成，都得益于一些热心者的支持和帮助。在本书的写作过程中，赵忠孝教授、姚威博士、张展为博士以及张秋菊、史林娟、张有明、戴璐、张展赫、董兆军、吴灼伟（插图）等参加了有关部分的写作。在此谨表谢意。同时，一如既往地希望得到读者广泛的批评和建议，以便将这本书改得更好。

本书就要出版了。它的出版，是笔者在这项教学改革工作中跨上的一个新的台阶。笔者衷心希望得到有关专家和读者的批评和建议，也希望能多结交一些志同道合者，把这项教学改革推向更新的境界。

张基温

丁酉秋于穗小海之畔

# 目 录

## 第 1 篇 C++面向对象起步

第 1 单元 职员类.....	3
1.1 从具体对象到职员类.....	3
1.1.1 具体职员对象的分析与描述.....	3
1.1.2 Employee 类的声明.....	4
1.1.3 C++保留字、标识符与名字空间.....	5
1.2 C++基本数据类型.....	7
1.2.1 C++算术数据类型的表示格式.....	7
1.2.2 C++算术数据类型的取值范围.....	9
1.2.3 C++运算符与算术数据类型的操作集合.....	10
1.3 表达式.....	11
1.3.1 字面值.....	11
1.3.2 数据实体.....	11
1.3.3 含有操作符的表达式.....	13
1.3.4 表达式中的隐式数据类型转换.....	14
1.4 函数.....	15
1.4.1 函数的关键环节.....	15
1.4.2 标准输出流 cout 与 printEmployee() 函数.....	17
1.4.3 构造函数与析构函数.....	18
1.4.4 构造函数重载.....	21
1.4.5 复制构造函数.....	22
1.4.6 主函数.....	24
1.5 程序编译.....	26
1.5.1 编译预处理.....	26
1.5.2 编译和连接.....	28
1.5.3 多文件程序的编译.....	28
1.6 知识链接.....	30
1.6.1 C++字面值.....	30
1.6.2 const 符号常量.....	33
1.6.3 指针=基类型+地址.....	34
1.6.4 指向对象的指针与 this.....	36

1.6.5  引用.....	38
习题 1.....	39
<b>第 2 单元 简单计算器.....</b>	<b>44</b>
2.1 简单计算器建模.....	44
2.1.1 简单计算器分析.....	44
2.1.2 Calculator 类的声明.....	44
2.2 calculate() 函数的实现.....	45
2.2.1 布尔类型与关系运算符.....	45
2.2.2 用 if-else 结构实现成员函数 calculate().....	46
2.2.3 用 switch 结构实现 calculate().....	48
2.2.4 if-else 判断结构与 switch 判断结构比较.....	49
2.2.5 Calculator 类测试.....	49
2.3 C++异常处理.....	52
2.3.1 程序错误.....	52
2.3.2 C++异常处理机制.....	54
2.3.3 C++异常处理技术.....	55
2.3.4 用类作为异常类型.....	58
2.3.5 捕获任何异常.....	62
2.4 简单桌面计算器的改进.....	63
2.4.1 使用浮点数计算的 Calculator 类.....	63
2.4.2 标准输入流与键盘输入.....	64
2.4.3 简单多项式计算的实现.....	67
2.4.4 用重复结构实现任意多项式计算.....	72
2.5 知识链接.....	74
2.5.1 条件表达式.....	74
2.5.2 局部变量.....	74
2.5.3 类属变量、实例变量与局部变量的比较.....	76
习题 2.....	76
<b>第 3 单元 素数产生器.....</b>	<b>81</b>
3.1 问题描述与对象建模.....	81
3.1.1 对象建模.....	81
3.1.2 getPrimeSequence() 函数的基本思路.....	82
3.2 使用 isPrime() 的 PrimeGenerator 类实现.....	82
3.2.1 用 for 结构实现的 getPrimeSequence() 函数.....	82
3.2.2 用 for 结构实现的 isPrime() 函数.....	84
3.2.3 完整的 PrimeGenerator 类及其测试.....	84
3.3 不使用 isPrime() 的 PrimeGenerator 类实现.....	85
3.3.1 采用嵌套重复结构的 getPrimeSequence() 函数.....	85

3.3.2	重复结构中的 continue 语句和 break 语句.....	86
3.4	数组.....	87
3.4.1	数组及其定义.....	87
3.4.2	数组的初始化规则.....	89
3.4.3	用数组存储素数序列.....	90
3.4.4	sizeof 操作符.....	91
3.4.5	C++11 中基于容器的 for 循环.....	91
3.4.6	数组 prime 的声明.....	92
3.5	string 类型.....	94
3.6	知识链接.....	94
3.6.1	C++操作符.....	94
3.6.2	左值表达式与右值表达式.....	96
3.6.3	具有副作用的表达式与序列点.....	97
3.6.4	表达式类型的推断与获取: auto 与 decltype.....	98
3.6.5	类型转换构造函数与 explicit 关键字.....	101
3.6.6	C++语句.....	105
	习题 3.....	105
<b>第 4 单元</b>	<b>Time 类</b> .....	<b>110</b>
4.1	Time 类需求分析与操作符重载.....	110
4.1.1	Time 类需求分析.....	110
4.1.2	关键字 operator 与操作符重载.....	111
4.1.3	操作符+的重载.....	112
4.1.4	增量操作符++的重载.....	113
4.1.5	用友元函数实现<<重载.....	116
4.1.6	赋值操作符=的重载.....	117
4.1.7	操作符重载的基本规则.....	118
4.1.8	Time 类的类型转换构造函数.....	120
4.2	浅复制与深复制.....	122
4.2.1	数据复制及其问题.....	122
4.2.2	复制构造函数再讨论.....	124
4.2.3	深复制的赋值操作符重载.....	126
4.3	动态内存分配.....	127
4.3.1	用 new 进行动态内存分配.....	127
4.3.2	用 delete 释放动态存储空间.....	128
4.3.3	对象的动态存储分配.....	129
4.3.4	动态内存分配时的异常处理.....	131
4.4	对象数组.....	132
4.4.1	对象数组的定义.....	132

88	4.4.2 对象数组元素的访问 .....	133
78	4.4.3 数组存储空间的动态分配 .....	134
78	4.5 知识链接 .....	134
88	4.5.1 友元 .....	134
09	4.5.2 const 修饰类成员与对象 .....	138
10	4.5.3 enum 类型 .....	140
10	习题 4 .....	144

## 第 2 篇 C++面向抽象程序设计

第 5 单元 继承 .....	153
5.1 单基继承 .....	153
5.1.1 公司人员的类层次结构模型 .....	153
5.1.2 C++继承关系的建立 .....	153
5.1.3 在派生类中重定义基类成员函数 .....	157
5.1.4 基于血缘关系的访问控制——protected .....	159
5.1.5 类层次结构中构造函数和析构函数的执行顺序 .....	160
5.2 类层次中的赋值兼容规则与里氏代换原则 .....	163
5.2.1 类层次中的类型赋值兼容规则 .....	163
5.2.2 里氏代换原则 .....	164
5.2.3 对象的向上转换和向下转换 .....	164
5.3 多基继承 .....	165
5.3.1 C++多基继承格式 .....	165
5.3.2 计算机系统=软件+硬件问题的类结构 .....	165
5.3.3 多基继承的歧义性问题 .....	167
5.3.4 虚拟基类 .....	169
5.4 用虚函数实现动态绑定 .....	170
5.4.1 画圆、三角形和矩形问题的类结构 .....	170
5.4.2 虚函数与动态绑定 .....	171
5.4.3 虚函数表与虚函数规则 .....	173
5.4.4 override 和 final .....	175
5.4.5 纯虚函数与抽象类 .....	177
5.5 运行时类型鉴别 .....	179
5.5.1 RTTI 概述 .....	179
5.5.2 dynamic_cast .....	179
5.5.3 typeid 类与 typeid 操作符 .....	185
习题 5 .....	188

第6单元 C++程序结构优化	195
6.1 面向对象程序设计的几个原则	195
6.1.1 引言	195
6.1.2 从可重用说起：合成/聚合优先原则	197
6.1.3 从可维护性说起：开闭原则	199
6.1.4 面向抽象原则	201
6.1.5 单一职责原则	207
6.1.6 接口分离原则	208
6.1.7 不要和陌生人说话	212
6.2 GoF 设计模式举例：工厂模式	214
6.2.1 概述	214
6.2.2 简单工厂模式	215
6.2.3 工厂方法模式	217
习题 6	219

### 第3篇 C++泛型程序设计

第7单元 模板	223
7.1 算法抽象模板——函数模板	223
7.1.1 从函数重载到函数模板	223
7.1.2 函数模板的实例化与具体化	224
7.2 数据抽象模板——类模板	227
7.2.1 类模板的定义	227
7.2.2 类模板的实例化与具体化	228
7.2.3 类模板的使用	230
7.2.4 类模板实例化时的异常处理	231
7.2.5 实例：MyVector 模板类的设计	232
习题 7	236
第8单元 STL 编程	242
8.1 STL 概述	242
8.1.1 容器	242
8.1.2 迭代器	244
8.1.3 容器的成员函数	247
8.1.4 STL 算法	250
8.1.5 函数对象	253
8.1.6 STL 标准头文件	255
8.2 扑克游戏——vector 容器应用实例	256
8.2.1 vector 容器的特点	256
8.2.2 扑克游戏对象模型	256

201	8.2.3	用 vector 容器对象 poker 存储 54 张扑克牌 .....	257
201	8.2.4	洗牌函数设计 .....	260
201	8.2.5	整牌函数设计 .....	263
201	8.2.6	发牌函数设计 .....	264
201	8.2.7	vector 操作小结 .....	267
100	8.3	list 容器及其应用实例 .....	268
100	8.3.1	构建 list 对象及其迭代器 .....	268
800	8.3.2	操作 list 对象 .....	269
210	8.3.3	基于 list 容器的约瑟夫斯问题求解 .....	273
110	8.4	string .....	276
110	8.4.1	字符串对象的创建与特性描述 .....	277
210	8.4.2	字符串对象的输入/输出 .....	277
710	8.4.3	字符串的迭代器与字符操作 .....	278
010	8.4.4	两个字符串间的操作 .....	282
	8.5	stack 容器 .....	284
	8.5.1	stack 及其特点 .....	284
100	8.5.2	stack 的操作 .....	284
100	8.5.3	应用举例: 将一个十进制整数转换为 $K$ 进制数 .....	285
200	8.6	关联容器 .....	287
100	8.6.1	用结构体定义的 pair 类模板 .....	287
100	8.6.2	set 和 multiset 容器 .....	289
100	8.6.3	map 和 multimap 容器 .....	293
800	8.7	知识链接 .....	297
000	8.7.1	const_iterator .....	297
100	8.7.2	分配器 .....	298
200	习题 8	.....	299

## 第 4 篇 C++深入编程

第 9 单元	C++实体访问探幽 .....	305
9.1	C++实体的基本访问属性 .....	305
	9.1.1 变量的生命期与 C++存储分配 .....	305
	9.1.2 标识符的作用域与链接性 .....	307
9.2	C++变量的存储属性 .....	309
	9.2.1 C++的 extern 关键字 .....	309
	9.2.2 C++的 static 关键字 .....	313
9.3	C++名字空间域 .....	320
	9.3.1 名字冲突与名字空间 .....	320
	9.3.2 名字空间的使用 .....	322

9.3.3	无名名字空间和全局名字空间 .....	324
9.4	const 指针与 const 引用 .....	325
9.4.1	const 修饰指针 .....	325
9.4.2	const 修饰引用 .....	326
9.4.3	顶层 const 与底层 const .....	327
9.5	C++11 的左值引用与右值引用 .....	329
9.5.1	C++11 的左值引用与右值引用的基本概念 .....	329
9.5.2	基于左值和右值概念的表达式分类 .....	330
9.5.3	C++引用的扩展及绑定规则 .....	330
9.5.4	C++ 11 的引用折叠与模板参数推导 .....	333
9.6	智能指针 .....	334
9.6.1	智能指针及其基本原理 .....	334
9.6.2	auto_ptr 智能指针 .....	335
9.6.3	Boost 库的智能指针 .....	337
	习题 9 .....	339
<b>第 10 单元 C++函数探幽 .....</b>		<b>345</b>
10.1	C++函数调用时的参数匹配规则 .....	345
10.1.1	C++函数调用时的参数匹配规则 .....	345
10.1.2	关于函数实参的计算顺序 .....	346
10.1.3	函数名重载 .....	346
10.1.4	形参带有默认值的函数 .....	349
10.1.5	参数数目可变的函数 .....	350
10.2	参数类型 .....	351
10.2.1	对象参数 .....	351
10.2.2	指针参数 .....	353
10.2.3	数组参数 .....	354
10.2.4	左值引用参数 .....	356
10.2.5	const 保护函数参数 .....	359
10.2.6	完美转发 .....	360
10.3	函数返回 .....	363
10.3.1	函数返回的基本规则 .....	363
10.3.2	函数返回指针 .....	363
10.3.3	函数返回引用 .....	364
10.3.4	const 保护函数返回值 .....	367
10.4	移动语义 .....	369
10.4.1	移动语义的提出 .....	369
10.4.2	移动构造函数与移动赋值操作符 .....	369
10.4.3	强制移动与 std::move() .....	371

10.5	Lambda 表达式	372
10.5.1	简单的 Lambda 表达式	372
10.5.2	在方括号中加入函数对象参数	373
习题 10		375
<b>第 11 单元 C++I/O 流</b>		379
11.1	流与 C++流类	379
11.1.1	流与缓冲区	379
11.1.2	C++流类库	380
11.1.3	ios 类声明	381
11.2	标准流对象与标准 I/O 流操作	383
11.2.1	C++标准流对象	383
11.2.2	标准输入/输出流操作	383
11.3	流的格式化	383
11.3.1	ios 类的格式化成员函数和格式化标志	384
11.3.2	格式化操作符	384
11.4	文件流	385
11.4.1	文件流的概念及其分类	385
11.4.2	文件操作过程	386
11.5	流的错误状态及其处理	389
11.5.1	流的出错状态	389
11.5.2	测试与设置出错状态位的 ios 类成员函数	390
习题 11		390
<b>附录 A C++保留字</b>		392
A.1	C++关键字	392
A.2	C++替代标记	392
A.3	C++库保留名称	393
A.4	C++特定字	393
<b>附录 B C++运算符的优先级别和结合方向</b>		394
<b>附录 C C++标准库与准标准库</b>		396
C.1	C++标准库头文件	396
C.1.1	标准库中与语言支持功能相关的头文件	396
C.1.2	支持流输入/输出的头文件	396
C.1.3	与诊断功能相关的头文件	397
C.1.4	定义工具函数的头文件	397
C.1.5	支持字符串处理的头文件	397
C.1.6	定义容器类的模板的头文件	397
C.1.7	支持迭代器的头文件	398
C.1.8	有关算法的头文件	398

C.1.9	有关数值操作的头文件.....	398
C.1.10	有关本地化的头文件.....	398
C.2	C++ Boost 库内容.....	398
C.2.1	字符串和文本处理库.....	399
C.2.2	容器库.....	399
C.2.3	迭代器库.....	399
C.2.4	算法库.....	400
C.2.5	函数对象和高阶编程库.....	400
C.2.6	泛型编程库.....	400
C.2.7	模板元编程.....	400
C.2.8	预处理元编程库.....	401
C.2.9	并发编程库.....	401
C.2.10	数学和数字库.....	401
C.2.11	排错和测试库.....	401
C.2.12	数据结构库.....	402
C.2.13	图像处理库.....	402
C.2.14	输入/输出库.....	402
C.2.15	跨语言混合编程库.....	402
C.2.16	内存管理库.....	402
C.2.17	解析库.....	402
C.2.18	编程接口库.....	402
C.2.19	综合类库.....	403
C.2.20	编译器问题的变通方案库.....	403
参考文献	.....	404