

THE EXPERT'S VOICE® IN C



Beginning C++

C++入门经典

(第4版)

[美] Ivor Horton 著
石磊 译



apress®

清华大学出版社



C++入门经典

(第4版)

[美] Ivor Horton 著
石磊 译



清华大学出版社
北 京

Ivor Horton

Beginning C++

EISBN: 978-1-4842-0008-7

Original English language edition published by Apress Media. Copyright © 2015 by Apress Media. Simplified Chinese-language edition copyright © 2015 by Tsinghua University Press. All rights reserved.

本书中文简体字版由 Apress 出版公司授权清华大学出版社出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

北京市版权局著作权合同登记号 图字：01-2015-3298

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

C++入门经典(第4版)/(美)霍尔顿(Horton,I.) 著；石磊 译. —北京：清华大学出版社，2015

书名原文：Beginning C++

ISBN 978-7-302-40628-0

I .C… II.①霍…②石… III. ①C 语言—程序设计 IV.TP312

中国版本图书馆 CIP 数据核字(2015)第 150327 号

责任编辑：王 军 于 平

装帧设计：孔祥峰

责任校对：成凤进

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

质 量 反 馈：010-62772015, zhiliang@tup.tsinghua.edu.cn

印 装 者：清华大学印刷厂

经 销：全国新华书店

开 本：185mm×260mm 印 张：38 插 页：1 字 数：973 千字

版 次：2015 年 8 月第 1 版 印 次：2015 年 8 月第 1 次印刷

印 数：1~5800

定 价：69.80 元

产品编号：061852-01

译者序

科学计算、分布式应用、嵌入式行业、智能控制、算法研究，乃至学术讨论和上机考试都会有一种语言的身影，那就是大名鼎鼎的 C++ 语言。原汁原味的 C++ 目前已经执行到 C++14 标准。听到这门编程语言，多数人伴随而来的是晦涩、复杂、强大等关键词！这样的感觉实际上是真实的。

对于这门复杂的编程语言，本书首先从一个最简短的 C++ 程序讲起，通过对这个完整的程序的实际编写引申出一些相关的知识，然后在后面的教程中对该程序不断地扩大和完善，这样，读者不至于一上来就被 C++ 吓坏，同时也能深刻地理解 C++ 的各个特性的设置目的。

有抱负的程序员必将面对三重障碍：

首先，必须掌握遍布程序设计语言中的各类术语。术语是专业人士及优秀业余爱好者之间的交流必不可少的，本书不仅详细讲解了这些术语，还强调如何自如地在各种环境下使用它们。

其次，必须理解如何使用语言元素，而不仅仅只是知道它们的概念。本书采用了代码片段来帮助理解语言元素的语法和作用，还用一些实际应用示例展示语言特性如何应用于特定的问题。

最后，必须领会如何在实际场景中应用该语言。本书针对特定的问题应用所学的知识，给出合理的解决方案，不仅帮助读者获得开发应用程序的能力与信心，了解如何联合以及更大范围地应用语言元素；而且能让读者了解设计实际应用程序与管理实际代码会碰到的问题。

读者使用本书时，记住只有通过动手实践才能学会编程，在学习的过程中，肯定会时不时犯许多错误而感到沮丧。当觉得自己完全停滞时，你要做的就是坚持。最终你一定会体验到成功的喜悦，回头想想，你会觉得它并没有你想象中的那么难。

本版与上一版的出版时间相距 10 年，期间 C++ 已经有了很大的变化，所以本版对前一版进行了彻底的修订，不仅内容经过了重新组织，添加了不少新内容，对每个示例都做了相应的修改，使它们能在新的 C++ 14 标准下编译运行。而且显著改善了可读性，充分体现了 C++ 语言的最新进展和当前的业界最佳实践。

本书是 C++ 初学者的权威指南。无论你是从事软件开发还是其他领域的工作，本书将为你打开程序开发之门。本书还是中高级程序员的必备参考。通过观察程序设计大师如何处理编程中的各种问题，使你获得新的领悟和指引。

在这里要感谢清华大学出版社的编辑们，他们为本书的翻译投入了巨大的热情并付出了很多心血。没有你们的帮助和鼓励，本书不可能顺利付梓。

本书由石磊翻译，参加本次翻译活动的还有孔祥亮、陈跃华、杜思明、熊晓磊、曹汉鸣、陶晓云、王通、方峻、李小凤、曹晓松、蒋晓冬、邱培强、洪妍、李亮辉、高娟妮、曹小震、陈笑。对于这本经典之作，译者本着“诚惶诚恐”的态度，在翻译过程中力求“信、达、雅”，但是鉴于译者水平有限，错误和失误在所难免，如有任何意见和建议，请不吝指正。

最后，希望读者通过阅读本书能早日步入 C++ 语言编程的殿堂，领略 C++ 语言之美！

译 者

作者简介



Ivor Horton 毕业于数学系，被人以获得巨大荣誉且代价极低而诱入信息技术行业。尽管他做了大量的工作，获得的荣誉并不多，但他继续在计算机领域工作。他主要关注编程、系统设计、咨询、非常复杂的项目的管理和实现。

Ivor 在工程设计和生产控制系统的设计、实现方面有多年的经验。他偶尔使用各种编程语言开发有用的应用程序，主要教科学家和工程师开发这些程序。他目前出版的著作包括 C、C++ 和 Java 方面的教材。目前，他不编写编程图书或给他人提供建议时，会钓鱼、旅游、享受生活。



技术编辑简介



Michael Thomas 作为独立撰稿人、团队领袖、项目经理和工程负责人，在软件开发方面工作了 20 余年。**Michael** 在移动设备方面有 10 余年的经验。他当前的重点是医疗领域，使用移动设备加速患者和医护人员之间的信息传递速度。

前 言

欢迎使用《C++入门经典(第4版)》。本书修订并更新了上一版(*Beginning ANSI C++*)。自上一版出版以来, C++语言有了很大的扩展和改进, 但不太可能把 C++的所有内容压缩到一本书中。本书提供的 C++语言基础知识和标准库功能, 足以让读者编写自己的 C++应用程序。掌握了本书介绍的知识, 读者应能毫无困难地扩展 C++专业知识的深度和广度。C++要比许多人想象的更容易理解。本书不需要读者具备任何编程知识。如果你非常渴望学习, 并具备逻辑思考的能力, 掌握 C++就会比想象的更容易。开发 C++技巧, 学习数百万人已在使用的语言, 掌握 C++技能, 它提供了在几乎任何环境下开发应用程序的能力。

本书的 C++语言对应最新的 ISO 标准, 一般称为 C++ 14。C++ 14 对以前的标准 C++ 11 进行了较小的扩展, 所以本书的内容大都不专用于 C++ 14。本书的所有示例都可以使用目前遵循 C++ 11 的编译器来编译和执行。

使用本书

要通过本书学习 C++, 需要一个遵循 C++ 11 标准的编译器和一个适合编写程序代码的文本编辑器。目前, 有几个编译器兼容 C++ 11, 其中一些是免费的。

GNU Project 生产的 GCC 编译器全面支持 C++ 11, 是一个开源产品, 可免费下载。安装 GCC 并将它与合适的编辑器一起使用, 对新手而言略有难度。安装 GCC 和合适编译器的一种简单方法是, 从 <http://www.codeblocks.org> 上下载 Code::Blocks。Code::Blocks 是 Linux、Apple Mac OS X 和 Microsoft Windows 的一个免费 IDE, 它允许使用几个编译器(包括用于 GCC、Clang 和 open Watcom 的编译器)开发程序。这表示, 安装 Code::Blocks 会获得 C、C++和 Fortran 的支持。

另一种方法是使用在 Microsoft Windows 下运行的 Microsoft Visual C++, 它不仅完全兼容 C++ 11, 而且已经安装好了。其免费版本是 Microsoft Visual Studio 2013 Express。编写本书时, 它可以编译本书的大多数示例, 最终应能编译所有示例。Microsoft Visual C++可以从 <http://www.microsoft.com/en-us/download/details.aspx?id=43733> 上下载。与 GCC 相比, Microsoft Visual C++编译器的限制多一些, 但根据它对 C++ 11 的支持力度, 这是一个专业的编译器, 还支持其他语言, 例如 C#和 Basic。当然, 也可以安装这两个编辑器。还有其他支持 C++ 11 的编译器, 在网上搜索会很快找到它们。

本书的内容循序渐进，所以读者应从头开始一直阅读到最后。但是，没有人能仅从一本书中获得所有的编程技巧。本书仅介绍如何使用 C++ 编程，读者应自己输入所有的例子，而不是从下载文件中复制它们，再编译和执行输入的代码，这似乎很麻烦，但输入 C++ 语句可以帮助理解 C++，特别是觉得某些地方很难掌握时，自己输入代码就显得非常有帮助。如果例子不工作，不要直接从书中查找原因，而应在自己输入的例子代码中找原因，这是编写 C++ 代码时必须做的一个工作。

犯错误也是学习过程中不可避免的，练习应提供大量犯错误的机会，最好自己编几个练习题。如果不确定如何编写代码，应翻看前面的内容。犯的错误越多，对 C++ 的功能和错误的原因认识得就越深刻。读者应完成所有的练习，记住不要看答案，直到肯定不能自己解决问题为止。许多练习都涉及某章内容的一个直接应用，换言之，它们仅是一种实践，但也有一些练习需要多动脑子，甚至需要一点灵感。

希望每个人都能成功驾驭 C++。

—Ivor Horton

目 录

第 1 章 基本概念	1
1.1 现代 C++	1
1.2 C++ 程序概念	2
1.2.1 注释和空白	2
1.2.2 预处理指令和头文件	3
1.2.3 函数	3
1.2.4 语句	4
1.2.5 数据输入输出	4
1.2.6 return 语句	5
1.2.7 名称空间	5
1.2.8 名称和关键字	6
1.3 类和对象	6
1.4 模板	7
1.5 程序文件	7
1.6 标准库	7
1.7 代码的表示样式	7
1.8 创建可执行文件	8
1.9 表示数字	9
1.9.1 二进制数	9
1.9.2 十六进制数	11
1.9.3 负的二进制数	12
1.9.4 八进制数	14
1.9.5 Big-Endian 和 Little-Endian 系统	14
1.9.6 浮点数	15
1.10 表示字符	16
1.10.1 ASCII 码	16
1.10.2 UCS 和 Unicode	17
1.11 C++ 源字符	17

1.11.1 三字符序列	18
1.11.2 转义序列	18
1.12 过程化编程方法和面向对象编程方法	20
1.13 本章小结	21
1.14 练习	22
第 2 章 基本数据类型	23
2.1 变量、数据和数据类型	23
2.1.1 定义整型变量	24
2.1.2 定义有固定值的变量	26
2.2 整型字面量	26
2.2.1 十进制整型字面量	27
2.2.2 十六进制的整型字面量	27
2.2.3 八进制的整型字面量	27
2.2.4 二进制的整型字面量	28
2.3 整数的计算	28
2.4 op=赋值运算符	33
2.5 using 声明和指令	34
2.6 sizeof 运算符	34
2.7 整数的递增和递减	35
2.8 定义浮点变量	37
2.8.1 浮点字面量	38
2.8.2 浮点数的计算	38
2.8.3 缺点	38
2.8.4 无效的浮点结果	39
2.9 数值函数	40
2.10 流输出的格式化	43
2.11 混合的表达式和类型转换	45
2.11.1 显式类型转换	46

2.11.2 老式的强制转换	48	4.4.3 逻辑非运算符	91
2.12 确定数值的上下限	49	4.5 条件运算符	92
2.13 使用字符变量	50	4.6 switch 语句	94
2.13.1 使用 Unicode 字符	51	4.7 无条件分支	98
2.13.2 auto 关键字	52	4.8 语句块和变量作用域	99
2.13.3 lvalue 和 rvalue	52	4.9 本章小结	100
2.14 本章小结	53	4.10 练习	100
2.15 练习	54		
第3章 处理基本数据类型	55	第5章 数组和循环	103
3.1 运算符的优先级和相关性	55	5.1 数据数组	103
3.2 按位运算符	57	5.2 理解循环	105
3.2.1 移位运算符	58	5.3 for 循环	106
3.2.2 使用按位与运算符	60	5.3.1 避免幻数	107
3.2.3 使用按位或运算符	61	5.3.2 用初始化列表定义 数组的大小	109
3.2.4 使用按位异或运算符	63	5.3.3 确定数组的大小	109
3.3 枚举数据类型	67	5.3.4 用浮点数值控制 for 循环	110
3.4 数据类型的同义词	70	5.3.5 使用更复杂的循环 控制表达式	112
3.5 变量的生存期	70	5.3.6 逗号运算符	113
3.5.1 定位变量的定义	71	5.3.7 基于区域的 for 循环	114
3.5.2 全局变量	71	5.4 while 循环	115
3.5.3 静态变量	74	5.5 do-while 循环	119
3.5.4 外部变量	75	5.6 嵌套的循环	120
3.6 本章小结	75	5.7 跳过循环迭代	123
3.7 练习	76	5.8 循环的中断	125
第4章 决策	77	5.9 字符数组	128
4.1 比较数据值	77	5.10 多维数组	131
4.1.1 应用比较运算符	78	5.10.1 初始化多维数组	134
4.1.2 比较浮点数值	79	5.10.2 在默认情况下设置 维数	135
4.2 if 语句	80	5.10.3 多维字符数组	136
4.2.1 嵌套的 if 语句	82	5.11 数组的替代品	137
4.2.2 不依赖编码的字符处理	84	5.11.1 使用 array<T,N>容器	138
4.3 if-else 语句	85	5.11.2 使用 std::vector<T> 容器	142
4.3.1 嵌套的 if-else 语句	87	5.11.3 矢量的容量和大小	143
4.3.2 理解嵌套的 if 语句	88		
4.4 逻辑运算符	89		
4.4.1 逻辑与运算符	90		
4.4.2 逻辑或运算符	90		

5.11.4 删除矢量容器中的 元素	145
5.12 本章小结	145
5.13 练习	146
第 6 章 指针和引用	149
6.1 什么是指针	149
6.1.1 地址运算符	151
6.1.2 间接运算符	152
6.1.3 为什么使用指针	153
6.2 char 类型的指针	154
6.3 常量指针和指向常量的 指针	158
6.4 指针和数组	159
6.4.1 指针的算术运算	160
6.4.2 计算两个指针之间的差	162
6.4.3 使用数组名的指针 表示法	162
6.5 动态内存分配	165
6.5.1 栈和堆	165
6.5.2 运算符 new 和 delete	166
6.5.3 数组的动态内存分配	167
6.5.4 通过指针选择成员	169
6.6 动态内存分配的危险	169
6.6.1 内存泄漏	169
6.6.2 自由存储区的碎片	170
6.7 原指针和智能指针	170
6.7.1 使用 unique_ptr<T> 指针	172
6.7.2 使用 shared_ptr<T> 指针	173
6.7.3 比较 shared_ptr<T> 对象	177
6.7.4 weak_ptr<T> 指针	177
6.8 理解引用	178
6.8.1 定义左值引用	179
6.8.2 在基于区域的 for 循环中 使用引用变量	180
6.8.3 定义右值引用	180
6.9 本章小结	181
6.10 练习	181

第 7 章 操作字符串	183
7.1 更好的 string 类型	183
7.1.1 定义 string 对象	184
7.1.2 string 对象的操作	186
7.1.3 访问字符串中的字符	188
7.1.4 访问子字符串	190
7.1.5 比较字符串	191
7.1.6 搜索字符串	196
7.1.7 修改字符串	203
7.2 国际字符串	207
7.3 包含 Unicode 字符串的 对象	208
7.4 原字符串字面量	208
7.5 本章小结	209
7.6 练习	210
第 8 章 定义函数	211
8.1 程序的分解	211
8.1.1 类中的函数	212
8.1.2 函数的特征	212
8.2 定义函数	212
8.2.1 函数体	213
8.2.2 函数声明	215
8.3 给函数传送参数	217
8.3.1 按值传送机制	217
8.3.2 按引用传送	223
8.3.3 main() 的参数	227
8.4 默认的参数值	228
8.5 从函数中返回值	231
8.5.1 返回指针	231
8.5.2 返回引用	235
8.6 内联函数	236
8.7 静态变量	237
8.8 函数的重载	239
8.8.1 重载和指针参数	241
8.8.2 重载和引用参数	241
8.8.3 重载和 const 参数	243
8.8.4 重载和默认参数值	244
8.9 函数模板	245

8.9.1 创建函数模板的实例	246	9.9 练习	283
8.9.2 显式指定模板参数	247	第 10 章 程序文件和预处理指令	285
8.9.3 函数模板的特例	248	10.1 理解转换单元	285
8.9.4 函数模板和重载	249	10.1.1 “一个定义”规则	286
8.9.5 带有多个参数的 函数模板	250	10.1.2 程序文件和链接	286
8.9.6 非类型的模板参数	251	10.1.3 确定名称的链接属性	286
8.10 拖尾返回类型	252	10.1.4 外部名称	287
8.11 函数指针	253	10.1.5 具有外部链接属性的 const 变量	287
8.12 递归	256	10.2 预处理源代码	288
8.12.1 应用递归	259	10.3 定义预处理标识符	289
8.12.2 Quicksort 算法	259	10.4 包含头文件	290
8.12.3 main()函数	260	10.5 名称空间	292
8.12.4 extract_words()函数	261	10.5.1 全局名称空间	293
8.12.5 swap()函数	262	10.5.2 定义名称空间	293
8.12.6 sort()函数	262	10.5.3 应用 using 声明	296
8.12.7 max_word_length() 函数	263	10.5.4 函数和名称空间	296
8.12.8 show_words()函数	264	10.5.5 未命名的名称空间	299
8.13 本章小结	265	10.5.6 名称空间的别名	299
8.14 练习	266	10.5.7 嵌套的名称空间	300
第 9 章 lambda 表达式	269	10.6 逻辑预处理指令	301
9.1 lambda 表达式简介	269	10.6.1 逻辑 #if 指令	301
9.2 定义 lambda 表达式	269	10.6.2 测试指定标识符的值	302
9.3 lambda 表达式的命名	270	10.6.3 多个代码选择	302
9.4 把 lambda 表达式传递给 函数	272	10.6.4 标准的预处理宏	303
9.4.1 接受 lambda 表达式 变元的函数模板	272	10.7 调试方法	304
9.4.2 lambda 变元的函数 参数类型	273	10.7.1 集成调试器	304
9.4.3 使用 std::function 模板类型	274	10.7.2 调试中的预处理指令	305
9.5 捕获子句	277	10.7.3 使用 assert 宏	309
9.6 在模板中使用 lambda 表达式	279	10.7.4 关闭断言机制	310
9.7 lambda 表达式中的递归	281	10.8 静态断言	310
9.8 本章小结	283	10.9 本章小结	312
		10.10 练习	313
		第 11 章 定义自己的数据类型	315
		11.1 类和面向对象编程	315
		11.1.1 封装	316
		11.1.2 继承	318

11.1.3 多态性	318	12.1.3 实现重载运算符	366
11.1.4 术语	319	12.1.4 全局运算符函数	369
11.2 定义类	320	12.1.5 提供对运算符的 全部支持	369
11.3 构造函数	322	12.1.6 在类中实现所有的 比较运算符	371
11.3.1 在类的外部定义 构造函数	324	12.2 运算符函数术语	373
11.3.2 默认构造函数的 参数值	326	12.3 默认类成员	374
11.3.3 在构造函数中使用 初始化列表	326	12.3.1 定义析构函数	375
11.3.4 使用 <code>explicit</code> 关键字	327	12.3.2 何时定义副本 构造函数	377
11.3.5 委托构造函数	329	12.3.3 实现赋值运算符	377
11.3.6 默认的副本构造函数	331	12.3.4 实现移动操作	379
11.4 访问私有类成员	332	12.4 重载算术运算符	380
11.5 友元	333	12.4.1 改进输出操作	384
11.5.1 类的友元函数	334	12.4.2 根据一个运算符实现 另一个运算符	386
11.5.2 友元类	336	12.5 重载下标运算符	387
11.6 <code>this</code> 指针	337	12.6 重载类型转换	394
11.7 <code>const</code> 对象和 <code>const</code> 函数成员	338	12.7 重载递增和递减运算符	395
11.8 类的对象数组	340	12.8 函数对象	396
11.9 类对象的大小	342	12.9 本章小结	397
11.10 类的静态成员	342	12.10 练习	398
11.10.1 静态数据成员	342	第 13 章 继承	399
11.10.2 类的静态函数成员	347	13.1 类和面向对象编程	399
11.11 析构函数	347	13.2 类的继承	401
11.12 类对象的指针和引用	350	13.2.1 继承和聚合	401
11.13 将指针作为类的成员	351	13.2.2 派生类	402
11.13.1 定义 <code>Package</code> 类	353	13.3 把类的成员声明为 <code>protected</code>	405
11.13.2 定义 <code>TruckLoad</code> 类	354	13.4 派生类成员的访问级别	405
11.13.3 实现 <code>TruckLoad</code> 类	355	13.4.1 在类层次结构中 使用访问指定符	406
11.14 嵌套类	360	13.4.2 改变继承成员的 访问指定符	408
11.15 本章小结	363	13.5 派生类中的构造 函数操作	408
11.16 练习	363		
第 12 章 运算符重载	365		
12.1 为类实现运算符	365		
12.1.1 运算符重载	366		
12.1.2 可以重载的运算符	366		

13.5.1 派生类中的副本 构造函数.....	412	14.4 通过指针释放对象.....	457
13.5.2 派生类中的默认 构造函数.....	414	14.5 本章小结.....	458
13.5.3 继承构造函数.....	414	14.6 练习.....	459
13.6 继承中的析构函数.....	415	第 15 章 运行时错误和异常	461
13.7 重复的成员名.....	417	15.1 处理错误.....	461
13.8 重复的函数成员名.....	418	15.2 理解异常.....	462
13.9 多重继承.....	419	15.2.1 抛出异常.....	463
13.9.1 多个基类.....	419	15.2.2 异常处理过程.....	465
13.9.2 继承成员的模糊性.....	420	15.2.3 未处理的异常.....	466
13.9.3 重复的继承.....	424	15.2.4 导致抛出异常的代码.....	467
13.9.4 虚基类.....	425	15.2.5 嵌套的 try 块.....	468
13.10 在相关的类类型之间 转换.....	425	15.3 用类对象作为异常.....	472
13.11 本章小结.....	426	15.3.1 匹配 Catch 处理程序和 异常.....	473
13.12 练习.....	426	15.3.2 用基类处理程序捕获 派生类异常.....	476
第 14 章 多态性	429	15.3.3 重新抛出异常.....	478
14.1 理解多态性.....	429	15.3.4 捕获所有的异常.....	481
14.1.1 使用基类指针.....	429	15.4 抛出异常的函数.....	482
14.1.2 调用继承的函数.....	431	15.4.1 函数 try 块.....	483
14.1.3 虚函数.....	434	15.4.2 不抛出异常的函数.....	483
14.1.4 虚函数中的默认参 数值.....	442	15.4.3 构造函数 try 块.....	484
14.1.5 通过智能指针调用 虚函数.....	443	15.4.4 异常和析构函数.....	484
14.1.6 通过引用调用虚函数.....	444	15.5 标准库异常.....	485
14.1.7 调用虚函数的基类 版本.....	445	15.5.1 异常类的定义.....	486
14.1.8 在指针和类对象之间 转换.....	446	15.5.2 使用标准异常.....	487
14.1.9 动态强制转换.....	447	15.6 本章小结.....	490
14.1.10 转换引用.....	449	15.7 练习.....	491
14.1.11 确定多态类型.....	449	第 16 章 类模板	493
14.2 多态性的成本.....	450	16.1 理解类模板.....	493
14.3 纯虚函数.....	451	16.2 定义类模板.....	494
14.3.1 抽象类.....	452	16.2.1 模板参数.....	495
14.3.2 间接的抽象基类.....	454	16.2.2 简单的类模板.....	495
		16.2.3 定义类模板的函数 成员.....	497
		16.3 创建类模板的实例.....	501
		16.4 类模板的静态成员.....	506

16.5	非类型的类模板参数	507	17.3	文件流	540
16.5.1	带有非类型参数的 函数成员的模板	510	17.3.1	在文本模式下写入 文件	540
16.5.2	非类型参数的变元	514	17.3.2	在文本模式下读取 文件	543
16.5.3	把指针和数组用作 非类型参数	514	17.4	设置流打开模式	546
16.6	模板参数的默认值	515	17.5	未格式化的流操作	554
16.7	模板的显式实例化	516	17.5.1	未格式化的流输入 函数	554
16.8	特殊情形	516	17.5.2	未格式化的流输出 函数	557
16.8.1	在类模板中使用 static_assert()	517	17.6	流输入输出中的错误	557
16.8.2	定义类模板特化	518	17.7	二进制模式中的流操作	559
16.8.3	部分模板特化	519	17.8	文件的读写操作	570
16.8.4	从多个部分特化中 选择	519	17.9	字符串流	577
16.9	类模板的友元	520	17.10	对象和流	578
16.10	带有嵌套类的类模板	521	17.10.1	给对象使用插入 运算符	579
16.11	本章小结	528	17.10.2	给对象使用提取 运算符	579
16.12	练习	528	17.10.3	二进制模式中的 对象 I/O	582
第 17 章	文件输入与输出	531	17.10.4	流中更复杂的对象	585
17.1	C++中的输入输出	531	17.11	本章小结	590
17.1.1	理解流	531	17.12	练习	590
17.1.2	使用流的优点	533			
17.2	流类	534			
17.2.1	标准流对象	535			
17.2.2	流的插入和提取操作	535			
17.2.3	流操纵程序	537			

第 1 章

基本概念

有时必须在详细解释示例之前使用示例中的元素。本章将概述 C++ 的主要元素及其组合方式，以帮助理解这些元素，还要探讨与计算机中表示数字和字符相关的几个概念。

本章主要内容

- 现代 C++ 的含义
- C++ 程序的元素
- 如何注释程序代码
- C++ 代码如何变成可执行程序
- 面向对象的编程方式与过程编程方式的区别
- 二进制、十六进制和八进制数字系统
- Unicode

1.1 现代 C++

现代 C++ 使用最新、最好的 C++ 元素编程。这是 C++ 11 标准定义的 C++ 语言，最新标准 C++ 14 对它进行了谨慎的扩展和改进。本书介绍 C++ 14 定义的 C++。

毫无疑问，C++ 是目前世界上使用最广泛、最强大的编程语言。如果打算学习一门编程语言，C++ 就是一个理想的选择。它能在极大范围内的计算设备和环境中高效地开发应用程序：个人电脑、工作站、大型计算机、平板电脑和移动电话。几乎任何程序都可以用 C++ 编写：设备驱动程序、操作系统、薪水管理程序、游戏等。C++ 编译器也是唾手可得的。最新的编译器运行在 PC、工作站和大型机上，常常具备跨编译功能，即可以在一个环境下开发代码，在另一个环境下编译并执行。

C++ 带有一个非常大的标准库，其中包含大量例程和定义，提供了许多程序需要的功能。例如，数值计算、字符串处理、排序和搜索、数据的组织和管理、输入输出等。标准库非常大，本书仅涉及其皮毛。详细描述标准库提供的所有功能，需要好几本书的篇幅。Beginning STL 是使用标准模板库的一个指南，而标准模板库是 C++ 标准库中以各种方式管理和处理数据的一个子集。

就 C++ 语言的范围和库的广度而言，初学者常常觉得 C++ 令人生畏。将 C++ 的全部内容放在一本书里是不可能的，但其实不需要学会 C++ 的所有内容，就可以编写实用的