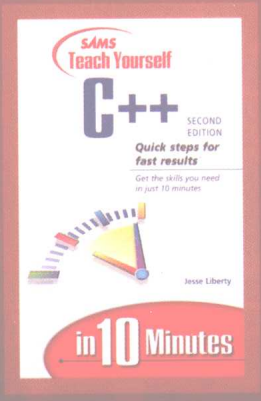
TURING 图灵程序设计丛书 C/C++系列

SAMS

Sams Teach Yourself C++ in 10 Minutes Second Edition

每天10分钟轻松掌握C++（第2版）

[美] Jesse Liberty 著
马礼伟 等译



- 无需编程经验，抓住地铁、公车上的零碎时间，你也能学会C++
- 微软专家带给你前所未有的轻松学习之旅
- 内容全面，言简意赅

人民邮电出版社
POSTS & TELECOM PRESS

TURING 图灵程序设计丛书 C/C++系列

Sams Teach Yourself C++ in 10 Minutes Second Edition

每天10分钟轻松掌握C++（第2版）

[美] Jesse Liberty 著
马礼伟 等译

人民邮电出版社
北 京

图书在版编目（CIP）数据

每天10分钟轻松掌握C++：第2版/（美）利伯蒂（Liberty, J.）著；马礼伟等译. —北京：人民邮电出版社，2009.8
（图灵程序设计丛书）
书名原文：Sams Teach Yourself C++ in 10 Minutes, Second Edition
ISBN 978-7-115-21105-7

Ⅰ.每… Ⅱ.①利… ②马… Ⅲ.C语言－程序设计
Ⅳ.TP312

中国版本图书馆CIP数据核字（2009）第114111号

内 容 提 要

本书围绕一个实用程序的开发展开，讲解了C++各方面的特性，包括函数、异常处理、堆与栈的区别、结构体、类、继承、多态等基本概念，还包括模板、性能优化等比较深入的知识。与其他只讲C++语言本身的图书不同，本书更注重语言特性的应用，并在应用的基础上讲解了迭代的软件开发过程，涉及软件的设计、开发、调试、测试以及重构等若干方面。

本书适合各层次C++程序员使用。

图灵程序设计丛书

每天10分钟轻松掌握C++（第2版）

◆ 著 [美] Jesse Liberty
译 马礼伟 等
责任编辑 傅志红
执行编辑 王军花

◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街14号
邮编 100061 电子函件 315@ptpress.com.cn
网址 http://www.ptpress.com.cn
北京鑫正大印刷有限公司印刷

◆ 开本：850×1168 1/32
印张：8
字数：250千字 2009年8月第1版
印数：1－3 000册 2009年8月北京第1次印刷

著作权合同登记号 图字：01-2009-3801号

ISBN 978-7-115-21105-7/TP

定价：29.00元

读者服务热线：(010)51095186 印装质量热线：(010)67129223
反盗版热线：(010)67171154

版 权 声 明

前　言

当你对C++产生了兴趣，却没有时间深入探究它的特性和功能的时候……

当你的C++技能有些生疏，却身陷繁事无暇温故的时候……

当你想学习C++编程，却不想买一本比你的计算机都沉的书的时候……

当你想去了解C++的一些特性，却不想艰苦地查找参考手册的时候……

本书正是为你准备的！

10分钟阅读，快速学习

假如你没有很长的时间用来阅读，假如你想获取一些语言知识和实用技巧以便更好地使用C++，假如你需要理解真实的C++程序是如何创建和演化成能产生有效结果的程序……

本书不会把C++参考手册改写成大篇幅的章节向你讲述C++的方方面面。相反，本书注重语言本身最重要的基本特性和高级特性，书中每章包含的特性大概只需10分钟即可读完。

不同的方法

如果你想学习如何编写程序却一直没学，则你在程序设计各个方面都需要一些帮助。如果你是一位专业的程序员，你可能就想了解C++在软件生命周期的各个阶段是如何工作的。

只要其中一种情况适合你，那本书就是你梦寐以求的。

本书以一个具体程序的演化和改进为线索组织内容，这让你更专注

于语言本身，而不必去为每一个例子创建一个新的应用程序。本书将向你展示如何创建、增强、修改、重构、测试以及优化C++程序。通过实践你会获得实用的技巧，以便可以用C++完成自己的工作。[1]

如何学习C++

本书将向你介绍学习C++语言必需的各方面内容，包括C++的基本特性和高级特性。本书将通过清晰的讲解和图表，帮助你最有效地学习每章只需10分钟的课程，主要内容有：

- 基本运算；
- 变量和常量；
- 判断语句（if和switch）和逻辑表达式；
- 循环语句（do、while和for）；
- 函数；
- 输入和输出；
- 错误和异常处理；
- 分离编译；
- 数组、指针和引用；
- 函数指针；
- 从堆中获取存储空间；
- 数据结构和用户自定义类型；
- 类和类成员；
- 函数和操作符重载；
- 继承与多重继承；
- 类的多态性；
- 模板。

无论你以前是初级程序员还是经验丰富的程序员，读完本书后都将获得开发和维护专业级C++程序的能力。

① 本书源代码可从图灵网站（http://www.turingbook.com）上免费注册下载。——编者注

本书约定

本书各章讲解了C++程序设计不同方面的特性。以下图标会帮助你理解书中一些特殊信息片段的用途。

注意 此图标用来标明C++新手经常会遇到麻烦的地方，并且提供解决这些问题的实用方案。

说明 此处的信息用简单明了的方式澄清概念和过程。

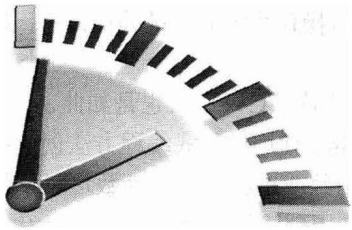
提示 在此，你可以找到一些解决疑难问题的思路。

简明注释 这里用通俗易懂的语言定义新出现的或生僻的术语。

致谢

首先，我要感谢我的家人，是他们不断地支持我写作，并容忍我几近疯狂的时间安排；其次，我得感谢在Sams工作的朋友，尤其是Carol Ackerman、Matt Purcell和Matt Wynalda；另外，我非常感谢Mark Cashman和Christopher McGee，他们的帮助使得本书更加出色。

目　录



第1章

准备工作

本章中，你将学习如何准备、设计、创建和修改C++程序。

1.1 目标

本书以一个具体的程序的开发为线索，记录了它从创建到成熟的整个过程。与很多程序一样，该程序将从一个非常简单的概念开始，然后在每一章中不断地改进，增加更多的功能。

使用这种方法的目的在于让你集中精力学习和使用C++语言。整本书基本上只围绕着一个示例程序，这样你就可以更专注示例程序的新功能和C++如何实现这些新增的功能。示例中只有少部分代码是专门创建用以演示语言特性的，大部分的增加和修改都是真实的，由程序的需求驱动，这与你将来开发程序的过程是一致的。

你将从本书学到如下内容。

- C++语言。
- 软件开发的生命周期。
- 所谓的渐进开发或自适应开发过程，即从简单的原型开始逐步演化为更复杂的程序。这种开发方式在专业程序设计过程中经常使用。

1.2 C++语言

C++语言是在开发下一代C语言的过程中产生的。1969年至1973年，Brian Kernighan和Dennis Ritchie在贝尔实验室发明了C语言。最初，C语言被设计成为计算机底层服务的编程语言，例如，用于编写操作系统（Kernighan和Ritchie就用它来写Unix），人们想用它替代汇编语言编程。

用汇编语言编写的程序很难读懂，并且极难创建为分离单元（separate unit）。C语言获得了广泛的接受，成为Unix操作系统的关键编程语言，最终也成为Windows平台的主要编程语言。

C语言从一开始就注重生成高性能的程序，C++语言也是如此。

C语言代表创建程序的过程式编程风格。使用过程式编程方法创建的程序，由一系列处理数据以产生结果的函数或过程组成。函数可以调用其他函数获取服务和协助，这样就可以使用分治策略简化问题求解。

C语言也是一种强类型语言。也就是说，C语言中的每一个数据项都有对应的类型，并且只能按照其类型定义使用。弱类型语言，例如BASIC，要么忽略要么隐藏这项重要的原则。强类型在程序第一次运行之前就可以保证程序的正确性。

Bjarne Stroustrup于1983年发明了C++语言，用作对C语言的扩展。C++具有C语言大部分的特性。实际上，用C++语言可以编写看起来与C程序一样的程序——本书的第一部分便是如此。你可以访问Stroustrup的网站http://www.research.att.com/~bs/C++.html，那里有很多极好的补充资料。

C++代表面向对象编程风格。当你开始编写面向对象的程序时，就会看到过程式编程风格与面向对象编程风格的异同。

面向对象编程把程序看做是用于生成对象的类的集合。类既包含数据，又包含函数。对象可以通过调用其他类的对象获取服务和协助。数据总是被隐藏在类中，因而面向对象的程序比过程式程序更安全，更容易被修改（修改过程式程序的数据结构有可能会影响程序中的所有函数）。

类由数据成员和成员函数组成。因为数据成员和成员函数与过程式的结构体和函数几乎完全一样，所以先从那里开始。

1.3 准备编程

在准备设计任何程序之前都要先问："我要解决的问题是什么？"每个程序都应该有一个清楚的、明确的目标，你会发现即使本书中最简单的程序也有这样的目标。

1.4 C++、ANSI C++、Windows等问题

本书不会对你使用的计算机有任何要求。本书讲解的是ISO/ANSI标

准C++（从现在起我姑且称之为标准C++）。美国国家标准委员会（ANSI）是国际标准化组织（ISO）的一个成员。国际标准化组织是一个制定标准的组织，它发布了定义C++语言程序的文档。你可以在任何系统上创建标准C++程序。

你不会在本书中看到与窗口、列表框和图形相关的内容。与操作系统（如Windows、Unix和Mac）直接打交道的类和函数的集合（通常称为库）提供这类特殊的功能，但这些并不是ISO/ANSI标准的一部分。因此，本书的程序只用控制台输入/输出数据，这样程序不但更简单，而且可用于各种操作系统。

本书中创建的程序稍做修改就可以使用图形用户界面（GUI）特性，因此你可以在学习本书的基础上学习使用那些库。

1.5 编译器和编辑器

对不熟悉这些术语的人，这里要解释一下。*编译器*（compiler）是把以便于人阅读的高级计算机语言编写的程序转换成利用操作系统（如Windows、Unix或Mac）能在计算机上运行的文件（即把*源代码转换为可执行程序*）的程序。*编辑器*（editor）是能录入源代码并将其保存为文件的程序（如Windows的写字板程序）。使用本书，你至少需要一个编译器和一个编辑器。

本书假定你知道如何使用编辑器创建、保存和修改文本文件，知道如何使用编译器和其他必需的工具，比如链接器。关于这些主题，请查阅操作系统和编译器的文档获取更多的内容。

书中的代码用Borland C++ Builder 5在严格的ANSI模式下编译通过，并用Microsoft Visual Studio 6.0检查过。你可以使用若干免费和共享的编译器，例如Borland的编译器（http://www.borland.com）、著名的gcc编译器（http://gcc.gnu.org）等。你可以通过本书网页（http://www.samspublishing.com）获得更多关于免费和共享的C++编译器的信息。

新建项目

你需要为你写的每段程序创建一个目录。如果还不清楚如何操作，请参考操作系统自带的文档。推荐你为本书创建一个根目录，在其下面

为每章的示例创建一个子目录。

你可以用编辑器创建C++源代码文件，然后保存到对应章的目录下。
8 源代码文件通常以.cpp（文件扩展名）结尾，用以标识是C++代码。

如果使用集成开发环境（IDE），例如Borland公司的C++ Builder或者微软公司的Visual C++，那么你需要通过“文件”→“新建”菜单新建项目。在这种环境下创建的程序需要新建控制台类型的项目。你可以把项目和组成它的源文件一起保存在对应章的目录下。关于这个过程，请参考IDE文档获取更详细的信息。

1.6 开发周期

如果程序一写成就能用，那么完整的开发周期应该包括编写代码、编译源代码和运行程序。但是，几乎每个程序，无论多么简单，都可能会出现错误。有些错误会使编译失败，而有些错误只有在程序运行时才会显现出来。

实际上，每个程序的开发都经过如下几个阶段。

- **分析**：决定程序需要做什么。
- **设计**：确定程序如何完成需要做的事情。
- **编辑**：根据设计编写源代码。
- **编译**：用编译器把源程序变成可执行文件。如果你写的C++语句不正确，编译器就会产生错误提示消息。通过理解这些含义模糊的错误提示消息，你就可以发现错误，然后修改代码直到编译成功。
- **链接**：编译器通常会自动链接编译成功的程序和其依赖的库。
- **测试**：编译器无法捕捉所有的错误，所以必须运行程序，有时必须以特别设计的输入数据运行，才能保证程序在运行时不会出什么差错。有的运行时错误会导致操作系统终止程序，有的则会产
9 生错误的结果。
- **调试**：你需要跟踪程序运行过程才能发现运行时错误是如何产生的。有时候是设计的错误，有时候是语言特性的误用，有时候则是操作系统使用不当。调试器是能帮忙找出这些问题的一种特殊的程序。如果没有调试器，你就需要包含能够告诉你程序每一步做什么的源代码。

无论你发现哪种类型的bug，都必须修复。修复过程涉及编辑源代码，重新编译，重新链接，重新运行程序，直到程序能正确运行。你将在本书中经历所有这些活动。

1.7 改进程序

程序完成之后，你可能需要它再做点别的事情。用户可能会要求增加新的功能，或者要求修改测试过程中没有发现的bug。你对程序内部的结构也可能会感觉不满意，想重构一下，让它变得更容易理解和维护。

1.8 一个简单的程序

这个简单的程序实际上什么都不做，但编译器并不关心。你可以成功地编译和运行这个程序。

代码中的行号 下面的代码清单中包含了行号。这些数字在书中起引用作用，不要在编辑器中输入。例如，代码清单1-1的第1行，你应该这样输入：

```
int main(int argc, char* argv[])
```

代码清单1-1 main.cpp —— 一个空程序

```
1: int main(int argc, char* argv[])
2: {
3:     return 0;
4: }
```

10

请确保你输入的代码与代码清单1-1中的内容相同（不计行号），特别需要注意标点符号。第3行以分号结尾，这个分号不能省略。

C++语言中所有的字符（包括标点符号）都很关键，必须正确输入。另外，C++是区分大小写的，例如`return`和`Return`表示不同的含义。

1.9 程序的组成

本程序只有一个函数，称为`main`函数。该函数出现在第1行，具有两个参数（在圆括号内），并返回一个数值（即位于行起始部分的`int`）。

函数（function）是完成某一特定任务的一组代码行。函数由位于顶部的函数头（用第二个单词作为函数名）和以左大括号（{）作为开始右大括号（}）作为结束的函数体组成，其中右括号后面的分号是可选的。关于函数的更多细节将在第7章讨论。

main函数是所有C++程序都必须具有的函数。系统传递给main函数的参数（称为实参）如下所示。

- int argc。运行程序时在命令行中输入的单词个数。
- char* argv[]。运行程序时在命令行中输入的内容，被分解成单词组。

本例中的main函数由函数头（第1行）和函数体（第2行～第4行）组成。第2行和第4行的大括号用来标明函数体的开始和结束。左大括号和右大括号中间所有的代码称为代码块（block）或复合语句。第3行是一条简单语句，当程序结束时返回数字0给系统。

这个程序只是一个.cpp文件，这种文件也称为模块（module）。有时一个模块由两个文件组成：一个头文件（以.h结尾）和一个.cpp文件。main.cpp不需要头文件。

返回值　main函数都有返回值，但经常不使用（在Unix和DOS系统中，返回值有时用于批处理文件，用来标识程序是否运行成功）。

1.10　编译期错误

程序的编译期错误可能是由输入错误或不正确地使用语言引起的。好的编译器不但能告诉你错误的内容，而且能正确指出错误发生的位置，有时候甚至还能提供修复错误的建议。

标点符号错误　当编译器尝试查找错误出现的位置时，遗漏分号和大括号的错误会导致错误提示消息不准确，你会发现提示消息指向的行实际上是正确的代码。要注意标点符号错误，这种错误有时解决起来十分棘手。

你可以在程序中故意设置一个错误，测试一下编译器的反应。如果main.cpp运行良好，就编辑一下代码，去掉右大括号（第4行）。程序现在看起来如代码清单1-2所示。

代码清单1-2 示范一个编译期错误

```
1: int main(int argc, char* argv[])
2: {
3:     return 0;
```

重新编译一下程序，你会发现编译器提示类似下面的错误：

12

```
[C++ Error] Main.cpp(3): E2134 Compound statement missing }
```

这个错误告诉你问题所在的文件名和行号，以及问题的类型。

有时错误提示消息只能提示问题大概出现的位置。如果去掉左大括号（第2行）而不是右大括号（第4行），你就会看到类似下面的错误：

```
[C++ Error] Main.cpp(3): E2141 Declaration syntax error
[C++ Error] Main.cpp(4): E2190 Unexpected }
```

一个错误有时会引发另外一个错误，例如本例中的情况。修改最前面的错误，重新编译，再修改下一个错误，这种改正错误的方法通常很有效。

如果能“像编译器一样思考”，你就会发现错误提示消息更容易理解。编译器先一次一个单词，然后一次一个句子浏览源文件，所以无法理解程序的意图，也无法理解你对程序的意图。

1.11 小结

本章中，你先学习了C++的历史和程序的生命周期，然后创建了一个简单的C++程序并对其进行编译，最后学会了如何理解编译器的错误提示消息。

13