TURING 图灵程序设计丛书

Addison Wesley

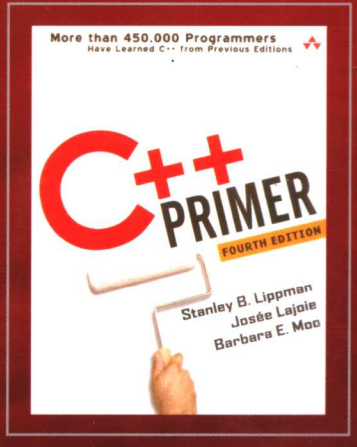
第4版

C++ Primer 中文版

Stanley B. Lippman
Josée Lajoie
Barbara E. Moo
著

李师贤 蒋爱军
梅晓勇 林 瑛
译

- 久负盛名、无可替代的C++经典著作
- 原版销量超过**450 000**册
- 教授最新的编程理念与实践
- 所有C++程序员不可或缺的参考书



人民邮电出版社
POSTS & TELECOM PRESS

TURING 图灵程序设计丛书

C++ Primer 中文版

（第 4 版）

Stanley B. Lippman
Josée Lajoie
Barbara E. Moo
著

李师贤　蒋爱军
梅晓勇　林　瑛
译

人民邮电出版社
POSTS & TELECOM PRESS

图书在版编目（CIP）数据

C++ Primer 中文版：第 4 版 /（美）李普曼，（美）拉茹瓦，（美）穆著；李师贤等译.
—北京：人民邮电出版社，2006.3
（图灵程序设计丛书）
ISBN 7-115-14554-7

Ⅰ. C... Ⅱ. ①李...②拉...③穆...④李... Ⅲ. C 语言—程序设计 Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2006）第 014313 号

内 容 提 要

本书是久负盛名的 C++经典教程，完美结合了 C++大师 Stanley B. Lippman 丰富的实践经验和 C++标准委员会原负责人 Josée Lajoie 对 C++标准的深入理解，已经帮助全球无数程序员学会了 C++。本版对前一版进行了彻底的修订，内容经过了重新组织，更加入了 C++先驱 Barbara E. Moo 在 C++教学方面的真知灼见，既显著改善了可读性，又充分体现了C++语言的最新进展和当前的业界最佳实践。书中不但新增大量教学辅助内容，用于强调重要的知识点，提醒常见的错误，推荐优秀的编程实践，给出使用提示，还包含大量来自实战的示例和习题。

对 C++基本概念和技术全面而且权威的阐述，对现代 C++编程风格的强调，使本书成为 C++初学者的最佳指南；对于中高级程序员，本书也是不可或缺的参考书。

图灵程序设计丛书

C++ Primer 中文版（第 4 版）

◆ 著 Stanley B. Lippman Josée Lajoie Barbara E. Moo
译 李师贤 蒋爱军 梅晓勇 林 瑛
责任编辑 杨海玲

◆ 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
北京顺义振华印刷厂印刷
新华书店总店北京发行所经销

◆ 开本：800×1000 1/16
印张：47.75
字数：1 149 千字 2006 年 3 月第 1 版
印数：8 001 – 11 000 册 2006 年 6 月北京第 2 次印刷

著作权合同登记号 图字：01-2005-3578 号
ISBN 7-115-14554-7/TP • 5273

定价：99.00 元

读者服务热线：(010)88593802 印装质量热线：(010)67129223

译 者 序

作为目前业界广泛使用的编程语言，C++可谓包罗万象、博大精深。20 年来，讲述 C++的图书早已经汗牛充栋、层出不穷，但其中业界公认的完整涵盖 C++标准的权威著作只有两部，曾经有评论将之喻为“倚天屠龙”。其中一部当然是 C++之父 Bjarne Stroustrup 所著的《C++程序设计语言》，内容精辟深刻，但是要求较高，只适合有一定经验的程序员提升功力之用。而另一部就是本书，自 1989 年初版以来，历经多次修订，始终保持了内容全面准确、循序渐进、明快易读的特色，早已奠定了无可替代的经典地位。原版到第 3 版就累积销售了 45 万册以上，第 3 版的中文版引入国内时，也产生了极大影响，甚至曾经出现过洛阳纸贵的局面。

本书的成功当然离不开强大的作者阵容。本书第一作者 Stanley Lippman 早在 C++还处于萌芽时期就是 Stroustrup 所在的 C++编译器项目团队的成员，目前又在微软领导 Visual C++和 CLI 的开发，对 C++可以说是了如指掌，实践经验极为丰富，加之多年来著书撰文不辍，在开发社区深孚众望，已成为公认的大师级人物。第二作者 Josée Lajoie 从第 3 版开始加入，她曾经在 IBM 从事 C++编译器开发，并担任过 ISO C++标准委员会核心语言工作组主席多年，她的加盟，充分保证了本书与 C++标准的兼容。

应该说，在很大程度上本书的第 3 版已臻完美。但是拿到第 4 版样书之后，我们发现新版完全不是对前版的简单扩充，不仅在布局结构上进行了彻底更新和重新规划，对具体文字和实例也进行了大幅改动，两个版本甚至很难找到相同的段落。在并无新版本的 C++标准定案发布的情况下，作者撰写新版而且做出这么大修订的原因何在？而新版又有何重要改进呢？

众所周知，C++从 C 语言继承而来的历史包袱，C++对多种编程风格的支持，以及各种误解和不良习惯，都大大增加了 C++教学和使用的复杂性，而传统教材和教学方法的各种弊端更加剧了这一情况，使 C++成为不少人望而生畏的难学难用的“专家语言”。

阅读本书后，我们不得不承认，几位大师级作者们很好地回应了上述挑战。这里我们应该特别提到本书新版增加的第三作者 Barbara Moo。她作为项目经理，曾经领导了包括 Stroustrup 和 Lippman 在内的贝尔实验室 C++编译器团队。她在斯坦福大学教授 C++课程的丰富经验和教学改革成果，对本书新版产生了重大影响。

相比之下，本书第 4 版主要有如下特点：

- 反映了现代理念。新版紧扣 C++语言当前的应用趋势——更加关注程序员的开发效率而不是系统的运行效率，摒弃了传统的阐述方式，不再注重低层编程技术，而是从一开始就强调标准库的使用，让人耳目一新。
- 突出了实践性。新版在继承了本书原有特色——全面、详细、准确地介绍 C++知识点的基础上，特别注重介绍那些实际开发中通用的、行之有效的编程技术。在特定场合，C++提供的丰富“武器库”中应该选择哪些设施，应该注意哪些问题，业界已经总结了哪些优秀的编程实践和易犯的错误等，而这些正是目前其他 C++图书所缺乏的。

- 增加了教学环节，改善了可读性。新版版式设计非常适合阅读，而且每一章都精心组织了重要术语、小结、大量示例和习题，文中另有丰富的额外提示和交叉引用，便于读者查找复习，消化核心概念，巩固所学知识。

我们在翻译过程中深深感到，本书新版在经典前版的基础上又有了质的飞跃，体现了世界C++教学方面的最新进展和最高水平。我们衷心希望本书中文版的出版，能够推动国内C++教学和使用的发展。

参加本书翻译工作的有李师贤、蒋爱军、梅晓勇、林瑛，全书由李师贤审校。参与部分校对或录入工作的还有古思山、黎永基、陈晓君、刘海伟等，在此对他们的工作表示衷心的感谢！

感谢人民邮电出版社图灵公司的编辑们，他们为保证本书的质量做了大量的工作。

由于书中概念和术语数目繁多，且有许多概念和术语目前尚无公认的中文译法，加之译者水平所限，译文中不当之处，恳请读者批评指正。

前　言

本书全面介绍了 C++语言。作为一本入门书（Primer），它以教程的形式对 C++语言进行清晰的讲解，并辅以丰富的示例和各种学习辅助手段。与大多数入门教程不同，本书对 C++语言本身进行了详尽的描述，并特别着重介绍了目前通行的、行之有效的程序设计技巧。

无数程序员曾使用本书的前几个版本学习 C++，在此期间 C++也逐渐发展成熟。这些年来，C++语言的发展方向以及 C++程序员的关注点，已经从以往注重运行时的效率，转到千方百计地提高程序员的编程效率上。随着标准库的广泛可用，我们现在能够比以往任何时候更高效地学习和使用 C++。本书这一版本充分体现了这一点。

第 4 版的改动

为了体现现代 C++编程风格，我们重新组织并重写了本书。书中不再强调低层编程技术，而把中心转向标准库的使用。书中很早就开始介绍标准库，示例也已经重新改写，充分利用了标准库设施。我们也对语言主题叙述的先后次序进行了重新编排，使讲解更加流畅。

除重新组织内容外，为了便于读者理解，我们还增加了几个新的环节。每一章都新增了“小结”和“术语”，概括本章要点。读者可以利用这些部分进行自我检查；如果发现还有不理解的概念，可以重新学习该章中的相关部分。

书中还加入了下述几种学习辅助手段：

- 重要术语用**黑体**表示，我们认为读者已经熟悉的重要术语则用楷体表示。这些术语都会出现在章后的“术语”部分。
- 书中用特殊版式突出标注的文字，是为了向读者提醒语言的重要特征，警示常见的错误，标明良好的编程实践，列出通用的使用技巧。希望这些标注可以帮助读者更快地消化重要概念，避免犯常见错误。
- 为了更易于理解各种特征或概念间的关系，书中大量使用了前后交叉引用。
- 对于某些重要概念和 C++新手最头疼的问题，我们进行了额外的讨论和解释。这部分也以特殊版式标出。
- 学习任何程序设计语言都需要编写程序。因此，本书提供了大量的示例。所有示例的源代码可从下列网址获得：

  `http://www.awprofessional.com/cpp_primer`

万变不离其宗，本书保持了前几版的特色，仍然是一部全面介绍 C++的教程。我们的目标是提供一本清晰、全面、准确的指南性读物。我们通过讲解一系列示例来教授 C++语言，示例除了解释语言特征外，还展示了如何善用这门语言。虽然读者不需要事先学过 C 语言（C++最初的基

础）的知识，但我们假定读者已经掌握了一种现代结构化语言。

本书结构

本书介绍了C++国际标准，既涵盖语言的特征，又讲述了也是标准组成部分的丰富标准库。C++的强大很大程度上来自它支持抽象程序设计。要学会用C++高效地编程，只是掌握句法和语义是远远不够的。我们的重点在于，教会读者怎样利用C++的特性，快速地写出安全的而且性能可与C语言低层程序相媲美的程序。

C++是一种大型的编程语言，这可能会吓倒一些新手。现代C++可以看成由以下三部分组成：

- 低级语言，多半继承自C。
- 更高级的语言特征，用户可以借此定义自己的数据类型，组织大规模的程序和系统。
- 标准库，使用上述高级特征提供一整套有用的数据结构和算法。

多数C++教材按照下面的顺序展开：先讲低级细节，再介绍更高级的语言特征；在讲完整个语言后才开始解释标准库。结果往往使读者纠缠于低级的程序设计问题和复杂类型定义的编写等细节，而不能真正领会抽象编程的强大，更不用说学到足够的知识去创建自己的抽象了。

本版中我们独辟蹊径。一开始就讲述语言的基础知识和标准库，这样读者就可以写出比较大的有实际意义的程序来。透彻阐释了使用标准库（并且用标准库编写了各种抽象程序）的基础知识之后，我们才进入下一步，学习用C++的其他高级特征来编写自己的抽象。

第一和第二部分讨论语言的基础知识和标准库设施。其重点在于学会如何编写C++程序，如何使用标准库提供的抽象设施。大部分C++程序员需要了解本书这两部分的内容。

除了讲解基础知识以外，这两部分还有另外一个重要的意图。标准库设施本身是用C++编写的抽象数据类型，定义标准库所使用的是任何C++程序员都能使用的构造类的语言特征。我们教授C++的经验说明，一开始就使用设计良好的抽象类型，读者会更容易理解如何建立自己的类型。

第三到第五部分着重讨论如何编写自己的类型。第三部分介绍C++的核心，即对类的支持。类机制提供了编写自定义抽象的基础。类也是第四部分中讨论的面向对象编程和泛型编程的基础。全书正文的最后是第五部分，这一部分讨论了一些高级特征，它们在构建大型复杂系统时最为常用。

致谢

与前几版一样，我们要感谢Bjarne Stroustrup，他不知疲倦地从事着C++方面的工作，他与我们的深厚友情由来已久。我们还要感谢Alex Stepanov，正是他最初凭借敏锐的洞察力创造了容器和算法的概念，这些概念最终形成了标准库的核心。此外，我们要感谢C++标准委员会的所有成员，他们多年来为C++澄清概念、细化标准和改进功能付出了艰苦的努力。

我们要衷心地感谢本书的审稿人，他们审阅了我们的多份书稿，帮助我们对本书进行了无数大大小小的修改。他们是Paul Abrahams、Michael Ball、Mary Dageforde、Paul DuBois、Matt Greenwood、Matthew P. Johnson、Andrew Koenig、Nevin Liber、Bill Locke、Robert Murray、Phil

Romanik、Justin Shaw、Victor Shtern、Clovis Tondo、Daveed Vandevoorde 和 Steve Vinoski。

书中所有示例都已通过 GNU 和微软编译器的编译。感谢他们的开发者和所有开发其他 C++编译器的人，是他们使 C++变成现实。

最后，感谢 Addison-Wesley 的工作人员，他们引领了这一版的整个出版过程：Debbie Lafferty——我们最初的编辑，是他提出出版本书的新版，他从本书最初版本起就一直致力于本书；Peter Gordon——我们的新编辑，他坚持更新和精简本书内容，极大地改进了这一版本；Kim Boedigheimer——他保证了我们所有人能按进度工作；还有 Tyrrell Albaugh、Jim Markham、Elizabeth Ryan 和 John Fuller，他们和我们一起经历了整个设计和制作过程。

目　录

第五部分 高级主题

第1章

快速入门

目录

本章介绍C++的大部分基本要素：内置类型、库类型、类类型、变量、表达式、语句和函数。在这一过程中还会简要说明如何编译和运行程序。

读者读完本章内容并做完练习，就应该可以编写、编译和执行简单的程序。后面的章节会进一步阐明本章所介绍的主题。

要学会一门新的程序设计语言，必须实际动手编写程序。在这一章，我们将编写程序解决一个简单的数据处理问题：某书店以文件形式保存其每一笔交易。每一笔交易记录某本书的销售情况，含有 ISBN（国际标准书号，世界上每种图书的唯一标识符）、销售册数和销售单价。每一笔交易形如：

```
0-201-70353-X 4 24.99
```

第一个元素是 ISBN，第二个元素是销售的册数，最后是销售单价。店主定期地查看这个文件，统计每本书的销售册数、总销售收入以及平均售价。我们要编写程序来进行这些计算。

在编写这个程序之前，必须知道 C++的一些基本特征。至少我们要知道怎么样去编写、编译和执行简单的程序。这个程序要做什么呢？虽然还没有设计出解决方案，但是我们知道程序必须：

- 定义变量。
- 实现输入和输出。
- 定义数据结构来保存要处理的数据。
- 测试是否两条记录具有相同的 ISBN。
- 编写循环，处理交易文件中的每一条记录。

我们将首先考察 C++的这些部分，然后编写书店问题的解决方案。

1.1　编写简单的 C++程序

每个 C++程序都包含一个或多个函数，而且必须有一个命名为 **main**。函数由执行函数功能的语句序列组成。操作系统通过调用 main 函数来执行程序，main 函数则执行组成自己的语句并返回一个值给操作系统。

下面是一个简单的 main 函数，它不执行任何功能，只是返回一个值：

```
int main()
{
    return 0;
}
```

操作系统通过 main 函数返回的值来确定程序是否成功执行完毕。返回 0 值表明程序成功执行完毕。

main 函数在很多方面都比较特别，其中最重要的是每个 C++程序必须含有 main 函数，且 main 函数是（唯一）被操作系统显式调用的函数。

定义 main 函数和定义其他函数一样。定义函数必须指定 4 个元素：返回类型、函数名、圆括号内的形参表（可能为空）和函数体。main 函数的形参个数是有限的。本例中定义的 main 函数形参表为空。7.2.6 节将介绍 main 函数中可以定义的其他形参。

main 函数的返回值必须是 int 型，该类型表示整数。int 类型是**内置类型**，即该类型是由 C++语言定义的。

函数体是函数定义的最后部分，是以**花括号**开始并以花括号结束的语句块：

```
{
    return 0;
}
```

例中唯一的语句就是 return，该语句终止函数。

注意 return 语句后面的分号。在 C++中多数语句以分号作为结束标记。分号很容易被忽略，而漏写分号将会导致莫名其妙的编译错误信息。

当 return 带上一个值（如 0）时，这个值就是函数的返回值。返回值类型必须和函数的返回类型相同，或者可以转换成函数的返回类型。对于 main 函数，返回类型必须是 int 型，0 是 int 型的。

在大多数系统中，main 函数的返回值是一个状态指示器。返回值 0 往往表示 main 函数成功执行完毕。任何其他非零的返回值都有操作系统定义的含义。通常非零返回值表明有错误出现。每一种操作系统都有自己的方式告诉用户 main 函数返回什么内容。

编译与执行程序

程序编写完后需要进行编译。如何进行编译，与具体操作系统和编译器有关。你需要查看有关参考手册或者询问有经验的同事，以了解所用的编译器的工作细节。

许多基于 PC 的编译器都在集成开发环境（IDE）中运行，IDE 将编译器与相关的构建和分析工具绑定在一起。这些环境在开发复杂程序时非常有用，但掌握起来需要花费一点时间。通常这些环境包含点击式界面，程序员在此界面下可以编写程序，并使用各种菜单来编译与执行程序。本书不介绍怎样使用这些环境。

大多数编译器，包括那些来自 IDE 的，都提供了命令行界面。除非你已经很熟悉你的 IDE，否则从使用简单的命令行界面开始可能更容易些。这样可以避免在学习语言之前得先去学习 IDE。

调用 GNU 或微软编译器

调用 C++编译器的命令因编译器和操作系统的不同而不同，常用的编译器是 GNU 编译器和微软 Visual Studio 编译器。调用 GNU 编译器的默认命令是 g++：

```
$ g++ prog1.cc -o prog1
```

这里的$是系统提示符。这个命令产生一个名为 prog1 或 prog1.exe 的可执行文件。在 UNIX 系统下，可执行文件没有后缀；而在 Windows 下，后缀为 .exe。-o prog1 是编译器参数以及用来存放可执行文件的文件名。如果省略-o prog1，那么编译器在 UNIX 系统下产生名为 a.out 而在 Windows 下产生名为 a.exe 的可执行文件。

微软编译器采用命令 cl 来调用：

```
C:\directory> cl -GX prog1.cpp
```

这里的 C:\directory>是系统提示符，directory 是当前目录名。cl 是调用编译器的命令，-GX 是一个选项，该选项在使用命令行界面编译程序时是必需的。微软编译器自动产生与源文件同名的可执行文件，这个可执行文件具有 .exe 后缀且与源文件同名。本例中，可执行文件命名为 prog1.exe。

更多的信息请参考你的编译器用户指南。