前　言

C++ Primer 的第二版和第三版之间的变化非常大。最引人注意的是，C++已经被国际标准化，这不但为语言增加了新的特性，比如异常处理、运行时刻类型识别(RTTI)、名字空间、内置布尔数据类型、新的强制转换方式，而且还大量修改并扩展了现有的特性，比如模板、支持面向对象（object-oriented）和基于对象（object-based）程序设计所需要的类（class）机制、嵌套类型、以及重载函数的解析机制。也许更重要的是，一个覆盖面非常广阔的库现在成了标准 C++的一部分，其中包括以前的标准模板库或 STL。新的 string 类型、一组顺序和关联容器类型——比如 vector、list、map 和 set——以及在这些类型上进行操作的一组可扩展的泛型算法（generic algorithms），都是这个新标准库的特性。本书不但包括了许多新的资料，而且还涵盖了怎样在 C++中进行程序设计的新的思考方法。简而言之，实际上，不但 C++已经被重新创造，它的 C++ Primer，第三版，也有了很大的变化。

在此第三版中，不但对语言的处理方式发生了根本的变化，而且作者本身也发生了变化：首先，我们的人数已经加倍。而且，我们的写作过程也已经被国际化了（尽管我们还牢牢扎根于北美大陆）：Stan 是美国人，Josée 是加拿大人。最后，这个双作者关系也反映了 C++团体的两类主要活动：Stan 现在正在华特迪思尼动画公司（Walt Disney Feature Animation）致力于以 C++为基础的 3D 计算机图形和动画应用，而 Josée 正专心于 C++的定义与实现，同时她也是 C++标准的核心语言小组的主席，以及 IBM 加拿大实验室的 C++编译器组的成员。

Stan 是 Bell 实验室中与 Bjarne Stroustrup（C++的发明人）一起工作的早期成员之一，从 1984 年开始一直从事 C++方面的工作。Stan 曾经致力于 cfront 的各种实现，从 1986 年的版本 1.1 到版本 3.0，并领导了 2.1 和 3.0 版本的开发组。之后，他参与了 Stroustrup 领导的、Foundation Research Project 项目中关于程序设计环境的对象模型部分。

Josée 作为 IBM 加拿大实验室 C++编译器组的成员已经有八年时间了。从 1990 年开始她成为 C++标准委员会的一个成员。她曾经担任委员会的副主席三年，目前担任核心语言小组委员会的主席已经达四年之久。

C++ Primer 的第三版展示了一个大幅修订的版本，不仅反映了语言的变化和扩展，也反映了作者洞察力和经验的变化。

本书的结构

C++ Primer 为 C++国际标准提供了一个全面的介绍。在这种意义上，它是一个初级读本（primer），它提供了一种指导性质的方法来描述 C++语言。（但是，它也为 C++语言提供了一种简单而温和的描述，从这个角度来看，它不是一本初级读物。）C++语言的程序设计要素，比如异常处理、容器类型、面向对象的程序设计等等，都在解决特定问题或程序设计任务的上下文环境中展示出来。C++语言的规则，比如重载函数调用的解析过程以及在面向对象程序设计下支持的类型转换，也都有广泛的论述，这看起来就超出了一本初级读本的范畴。我们相信，为了加强读者对于 C++语言的理解，覆盖这些内容是必要的。对于这些材料，读者应该不时地回头翻阅，而不是一次消化了事。如果开始的时候你发现这些内容比较难以接受或者过于枯燥，请把它们放到一边，以后再回头来看——我们把这样的章节加上特殊的记号：※。

我们没有假设读者具有 C 语言的知识，但是，熟悉某些现代的块结构语言会使得学习进展更快一些。本书的意图是作为学习 C++的第一本书；而不是学习程序设计的第一本书！为了确保这一点，我们都以一个公共的词汇表作为开始；然而，开始的章节涵盖了一些基本的概念，比如循环语句和变量等，有些读者可能会觉得这些概念太浅显了。不必担心：深层的内容很快就会看到。

C++的许多威力来自于它对程序设计新方法的支持，以及对程序设计问题的思考方式。因此，要想有效地学习使用 C++，不只要求简单地学会一组新的语法和语义。为了使这种学习更加容易，本书将围绕一系列可扩展的例子来组织内容。这些例子被用来介绍各种语言特性的细节，同时也说明了这些语言特性的动机所在。当我们在一个完整例子的上下文环境中学习语言特性时，这样的特性为什么会有用处的原因也就变得很清楚了，它使我们对于“何时以及怎样在实际的问题解决过程中使用这些特性”有了一点感觉。另外，把焦点放在例子上，可使读者能够尽早地使用一些概念，随着读者的知识基础被建立起来之后，这些概念会被进一步完整地解释清楚。本书初期的例子含有 C++基本概念的简单用法，读者可以先领略一下 C++中程序设计的概貌，而不要求完全理解 C++程序设计和实现的细节。

第 1 章和第 2 章形成了一个独立完整的 C++介绍和概述。第 1 篇的目的是使我们快速地理解 C++支持的概念和语言设施，以及编写和执行一个程序所需要的基础知识。读完这部分内容之后，你应该对 C++语言有了一些认识，但是还谈不上真正理解 C++。这就够了：那是本书余下部分的目的。

第 1 章向我们介绍了语言的基本元素：内置数据类型、变量、表达式、语句以及函数。它将介绍一个最小的、合法的 C++程序，简要讨论编译程序的过程，介绍所谓的预处理器（preprocessor），以及对输入和输出的支持。它给出了多个简单但却完整的 C++程序，鼓励读者亲自编译并执行这些程序。第 2 章介绍了 C++通过类机制，为基于对象和面向对象的程序设计提供了支持，同时通过数组抽象的演化过程来说明这些设计思想。另外，它简要介绍

了模板、名字空间、异常处理，以及标准库为一般容器类型和泛型程序设计提供的支持。这一章的进度比较快，有些读者可能会觉得难以接受。如果是这样，我们建议你跳过这一章，以后再回头来看。

C++的基础是各种设施，它们使用户能够通过定义新的数据类型来扩展语言本身，这些新类型可以具有与内置类型一样的灵活性和简单性。掌握这些设施的第一步是理解基本语言本身。第 3 章到第 6 章（第二篇）在这个层次上介绍了 C++语言。

第 3 章介绍了 C++语言预定义的内置和复合数据类型，以及 C++标准库提供的 string、complex、vector 类数据类型。这些类型构成了所有程序的基石。第 4 章详细讨论了 C++语言支持的表达式，比如算术、关系、赋值表达式。语句是 C++程序中最小的独立单元，它是第 5 章的主题。C++标准库提供的容器类型是第 6 章的焦点。我们不是简单地列出所有可用的操作，而是通过一个文本查询系统的实现，来说明这些容器类型的设计和用法。

第 7 章到第 12 章（第 3 篇）集中在 C++为基于过程化的程序设计提供的支持上。第 7 章介绍 C++函数机制。函数封装了一组操作，它们通常形成一项单一的任务，如 print()。（名字后面的括号表明它代表了一个函数。）关于程序域和变量生命期的概念、以及名字空间设施的讨论是第 8 章的主题。第 9 章扩展了第 7 章中引入的关于函数的讨论，介绍了函数的重载。函数重载允许多个函数实例（它们提供一个公共的操作）共享一个公共的名字（但是，要求不同的实现代码）。例如，我们可以定义一组 print()函数来输出不同类型的数据。第 10 章介绍和说明函数模板的用法。函数模板为一组可能无限多个函数实例的自动生成提供一个规范描述（“处方”），这些函数实例的类型不同，但实现保持不变。

C++支持异常处理设施。一个异常表示一个没有预料的程序行为，比如所有可用的程序内存耗尽。异常出现的程序部分抛出一个异常——即，使程序的其他部分可以访问到这个异常。程序中的某个函数必须捕获这个异常并做一些必要的动作。对于异常处理的讨论跨越了两章。第 11 章用一个简单的例子介绍了异常处理的基本语法和用法，该例子捕获和抛出一个类类型的异常。因为在我们的程序中，实际被处理的异常通常是一个面向对象类层次结构的类对象，所以，关于怎样抛出和处理异常的讨论一直继续到第 19 章，也就是在介绍面向对象程序设计之后。

第 12 章介绍标准库提供的泛型算法集合，看一看它们怎样和第 6 章的容器类型以及内置数组类型互相作用。这一章以一个使用泛型算法的程序设计作为开始。第 6 章引入的 iterator（迭代器）在第 12 章被进一步讨论，因为它们为泛型算法与实际容器的绑定提供了粘合剂。这一章也引入并解释了函数对象的概念。函数对象使我们能够为泛型算法中用到的操作符（比如等于或小于操作符）提供替换语义。关于泛型算法在附录中有详细说明，并带有用法的示例。

第 13 章到第 16 章（第 4 篇）焦点集中在基于对象的程序设计上——即，创建独立的抽象数据类型的类设施定义和用法。通过创建新的类型来描述问题域，C++允许程序员在写应用程序时可以不用关心各种乏味的簿记工作。应用程序的基本类型可以只被实现一次，而多次被重用，这使程序员能够将注意力集中在问题本身上，而不是实现细节上。封装数据的设施可以极大地简化应用程序的后续维护和改进工作。

第 13 章集中在一般的类机制上：怎样定义一个类、信息隐藏的概念（即，把类的公有接口同私有实现分离），以及怎样定义并封装一个类的对象实例。这一章还有关于类域、嵌套类、类作为名字空间成员的讨论。

第 14 章详细讨论 C++为类对象的初始化、析构以及赋值而提供的特殊支持。为了支持这些特殊的行为，需要使用一些特殊的成员函数，分别是构造函数、析构函数和拷贝赋值操作符。这一章我们还将看一看按成员初始化和拷贝的主题（即指一个类对象被初始化为或者赋值为该类的另一个对象），以及为了有效地支持按成员初始化和拷贝而提出的*命名返回值*（*named return value*）扩展。

第 15 章将介绍类特有的操作符重载，首先给出一般的概念和设计考虑，然后介绍一些特殊的操作符，如赋值、下标、调用以及类特有的 new 和 delete 操作符。这一章还介绍了一个类的友元的概念（对一个类具有特殊的访问特权）以及友元的必要性。然后讨论用户定义的转换，包括底层的概念和用法的扩展实例。这一章还详细讨论了函数重载解析的规则，并带有代码示例说明。

类模板是第 16 章的主题。类模板是一个被用于创建一族类的规范描述（“处方”），这样的类包含一个或多个参数化的类型或值。例如，一个 vector 类可以对内含的元素类型进行参数化。一个 buffer 类，可以对内含的元素类型以及缓冲区的大小进行参数化。更复杂的用法，比如在分布式计算中，IPC 接口、寻址接口、同步接口等，都可以被参数化。这一章讨论了怎样定义一个类模板、怎样创建一个类模板的特定类型的实例、怎样定义类模板的成员（成员函数、静态成员和嵌套类型），以及怎样用类模板来组织我们的程序。最后以一个扩展的类模板的例子作为结束。

面向对象的程序设计和 C++的支持机制是第 17、18、19 和 20 章（第 5 篇）的主题。第 17 章介绍了 C++对于面向对象程序设计主要要素的支持：继承和动态绑定。在面向对象的程序设计中，父/子关系（也被称为类型/子类型关系）被用来定义“共享共同行为的各个类”。一个类不是重新实现那些共享特性，而是继承了父类的数据和操作。子类或者子类型只针对它与父类不同的地方进行设计。 例如，我们可以定义一个父类 Employee，以及两个子类型：TemporaryEmpl 和 Manager。这些子类型继承了 Employee 的全部行为。它们只实现自己特有的行为。

继承的第二个方面，被称为多态性，是指父类型具有“引用它的任何子类型”的能力。例如，一个 Employee 可以指向自己的类型，也可以指向 TemporaryEmpl 或者 Manager。动态绑定是指“在运行时刻根据多态对象的实际类型来确定应该执行哪个操作”的解析能力。在 C++中，这是通过虚拟函数机制来处理的。

第 17 章介绍了面向对象程序设计的基本特性。这一章说明了如何设计和实现一个 Query 类层次结构，用来支持第 6 章实现的文本查询系统。

第 18 章介绍更为复杂的继承层次结构，多继承和虚拟继承机制使得这样的层次结构成为可能。这一章利用多继承和虚拟继承，把第 16 章的模板类例子扩展成一个三层的类模板层次结构。

第 19 章介绍运行时刻类型识别（RTTI）设施。RTTI 使得我们的程序在执行过程中可

以查询一个多态类对象的类型。例如，我们可以询问一个 Employee 对象，它是否实际指向一个 Manager 类型。另外，第 19 章回顾了异常处理机制，讨论了标准库的异常类层次机构，并说明了如何定义和处理我们自己的异常类层次结构。这一章也深入讨论了在继承机制下的重载函数解析过程。

第 20 章详细说明了如何使用 C++的 iostream 输入/输出库。它通过例子说明了一般的数据输入和输出，说明了如何定义类特有的输入输出操作符实例、如何辨别和设置条件状态、如何对数据进行格式化。iostream 库是一个用虚拟继承和多继承实现的类层次结构。

C++ Primer 以一个附录作为结束，附录给出了每个泛型算法的简短讨论和程序例子。这些算法按字母排序，以便于参考。

最后，我们要说的是，无论谁写了一本书，他所省略掉的，往往与他所讲述的内容一样重要。C++语言的某些方面，比如构造函数的工作细节、在什么条件下编译器会创建内部临时对象、或者对于效率的一般性考虑，虽然这些方面对于编写实际的应用程序非常重要，但是不适合于一本入门级的语言书籍。在开始写作 C++ Primer 的第三版之前，Stan 写的 *Inside the C++ Object Model* （参见本前言最后所附的参考文献中的 [LIPPMAN96a] ）包含了许多这方面的内容。当读者希望获得更详细的说明（特别是讨论基于对象和面向对象的程序设计）时，本书常常会引用“ Inside the C++ Object Model ”中的讨论。

本书故意省略了 C++标准库中的某些部分，比如对本地化和算术运算库的支持。C++标准库非常广泛，要想介绍它的所有方面，则远远超出了本书的范围。在后面所附的参考文献中，某些书更详细地讨论了该库（见[MUSSER96] 和 [STROUSTRUP97]）。我们相信，在这本书出版之后，一定会有更多的关于 C++标准库各个方面的书被随之出版。

第三版的变化

第三版的变化分为四类：

1．涵盖了语言所增加的新特性：异常处理、运行时刻类型识别、名字空间、内置 bool 类型、新风格的类型强制转换。

2．涵盖了新的 C++标准库，包括 complex 和 string 类型、auto_ptr 和 pair 类型、顺序容器和关联容器类型（主要是 list、vector、map、set 容器），以及泛型算法。

3．对原来的文字作了调整，以反映出标准 C++对原有语言特性的精炼、变化以及扩展。语言精炼的一个例子是，现在能够前向声明一个嵌套类型，这在以前是不允许的。语言变化的一个例子是，一个虚拟函数的派生类实例能够返回一个“ 基类实例的返回类型 ”的派生类。这种变化支持一个被称为 clone 或 factory 的方法（关于 clone()虚拟函数，见 17.5.7 节说明）。对原有语言特性进行扩展的一个例子是，现在可以显式地指定一个函数模板的一个或多个模板实参。（实际上，模板已经被大大地扩展了，差不多已经成为一个新特性！）

4．加强了对 C++高级特性的对待和组织方式，尤其是对于模板、类以及面向对象程序设计。Stan 从一个相对较小的 C++提供者团体到一般的 C++用户团体的转移，带来的一个影响是，他相信，越是深入地了解问题，则程序员越是能够高明地使用 C++语言。因此，在第

三版中，许多情况下，我们已经把焦点转移到如何更好地说明底层特性的概念，以及怎样最好地使用它们，并指出在适当的时候应该避免潜在的缺陷。

C++的未来

在出版这本书的时候，ISO/ANSI C++标准委员会已经完成了 C++第一个国际标准的技术工作。该标准将在 1998 年的夏天由 ISO 公布。

C++标准公布之后，支持标准 C++的 C++编译器实现也将很快会推出。随着标准的公布，C++语言的进化将会稳定下来。这种稳定性使得以标准 C++编写的复杂的程序库，可以被用来解决工业界特有的问题。因此，在 C++世界中，我们将会看到越来越多的 C++程序库。

一旦标准被公布，标准委员会仍然会继续工作（当然步伐会慢下来），以解决 C++标准的用户所提出的解释请求。这会导致对 C++标准作一些细微的澄清和修正。如果需要，国际标准将会每五年修订一次，以考虑技术的变化和工业界的需要。

C++标准公布五年之后将会发生什么事情我们还不知道。有可能是，一个被工业界广泛使用的新库组件将被加入到 C++标准库的组件集中。但是，到现在为止，面对 C++标准委员会已经完成的工作，C++的命运就全掌握在用户的手中了。

致谢

特别的感谢总是给予 Bjarne Stroustrup，感谢他给予我们如此美妙的编程语言，以及这些年他对我们的关心。特别感谢 C++标准委员会成员的奉献和艰苦工作（常常是自愿的），以及他们对 C++所作的贡献。

下面这些人为本书稿的各个草稿提供了许多有益的建议：Paul Abrahams、Michael Ball、Stephen Edwards、 Cay Horstmann、 Brian Kernighan、Tom Lyons、 Robert Murray、Ed Scheibel、Roy Turner 和 Jon Wada。尤其要感谢 Michael Ball 的建议和鼓励。特别感谢 Clovis Tondo 和 Bruce Leung 为本书作的深度评论。

Stan想把特别的感谢给予Shyh-Chyuan Huang 和 Jinko Gotoh，感谢他们对Firebird给与的帮助和支持，当然还有Jon Wada和Josée。

Josée要感谢Gabby Silberman、Karen Bennet以及在高级研究中心（the Centre for Advanced Studies）的小组，感谢他们为写作这本书所提供的支持。最大的感谢要给予Stan，感谢他带着她经历了这项伟大的冒险活动。

最后，我们两个都要感谢编辑们的辛苦工作以及巨大的耐心：Debbie Lafferty，他从刚刚开始的时候，就已经为这本书工作了；Mike Hendrickson以及 John Fuller。Big Purple 公司在排版上做了精彩的工作。6.1节的插图是Elena Driskill做的。非常感谢Elena使我们能够再版它。

第二版的致谢

这本书是无数看不见的手在帮助作者的结果。最由衷地感谢 Barbara Moo。她的鼓励、

建议，以及对原手稿无数遍地仔细阅读已经无法评价。特别感谢 Bjarne Stroustrup 持续不断的帮助和鼓励，以及他给予我们如此美妙的编程语言；还有 Stephen Dewhurst，他在我第一次学习 C++时给了许多支持；以及 Nancy Wilkinson，另一位 cfront 编写者和 Gummi Bears 的提供者。

Dag Brück、Martin Carroll、William Hopkins、Brian Kernighan、Andrew Koenig、Alexis Layton 以及 Barbara Moo 提供了特别详细和敏锐的建议。他们的评论大大完善了这本书。Andy Baily、Phil Brown、James Coplien、Elizabeth Flanagan、David Jordan、Don Kretsch、Craig Rubin、Jonathan Shopiro、Judy Ward、Nancy Wilkinson 以及 Clay Wilson 检查了书稿的各种草稿，提出了许多有益的建议。David Prosser 澄清了 ANSI C 的许多问题。Jerry Schwarz，他实现了 iostream 包，提供了附录 A 所依据的原始文档（在第三版中是第 20 章）。非常感谢他对附录的细致检查。感谢版本 3.0 开发小组的其他成员：Laura Eaves、George Logothetis、Judy Ward、and Nancy Wilkinson。

以下的人对 Addison-Wesley 的手稿作了评论：James Adcock、Steven Bellovin、Jon Forrest、Maurice Herlihy、Norman Kerth、Darrell Long、Victor Milenkovic 以及 Justin Smith。

以下的人指出了第一版的各种印刷错误：David Beckedorff、Dag Brück、John Eldridge、Jim Humelsine、Dave Jordan、Ami Kleinman、Andrew Koenig、Tim O'Konski、Clovis Tondo 和 Steve Vinoski。

我非常感谢 Brian Kernighan 和 Andrew Koenig 提供了大量可用的排版工具。

参考文献

下面的文献，有的影响了本书的撰写，有的则是关于 C++的重要推荐资料。

[BOOCH94] Booch, Grady, *Object-Oriented Analysis and Design*, Benjamin/Cummings, Redwood City, CA (1994) ISBN 0-8053-5340-2.

[GAMMA95] Gamma, Erich, Richard Helm, Ralph Johnson, and John Vlissides, *Design Patterns*, Addison Wesley Longman, Inc., Reading, MA (1995) ISBN 0- 201-63361-2.

[GHEZZI97] Ghezzi, Carlo, and Mehdi Jazayeri, *Programming Language Concepts, 3rd Edition*, John Wiley and Sons, New York, NY (1997) ISBN 0-471-10426-4.

[HARBISON88] Samuel P. Harbison and Guy L. Steele, Jr, *C: A Reference Manual, 3rd Edition*, Prentice-Hall, Englewood Cliffs, NJ (1988) ISBN 0-13-110933-2.

[ISO-C++97] Draft Proposed International Standard for Information Systems — Programming Language C++ - Final Draft (FDIS) 14882.

[KERNIGHAN88] Kernighan, Brian W., and Dennis M. Ritchie, *The C Programming Language*, Prentice-Hall, Englewood Cliffs, NJ (1988) ISBN 0-13-110362-8.

[KOENIG97] Koenig, Andrew, and Barbara Moo, *Ruminations on C++*, Addison Wesley Longman, Inc., Reading, MA (1997) ISBN 0-201-42339-1.

[LIPPMAN91] Lippman, Stanley, *C++ Primer, 2nd Edition*, Addison Wesley Longman, Inc., Reading, MA (1991) ISBN 0-201-54848-8.

[LIPPMAN96a] Lippman, Stanley, *Inside the C++ Object Model*, Addison Wesley Longman, Inc., Reading, MA (1996) ISBN 0-201-83454-5.

[LIPPMAN96b] Lippman, Stanley, Editor, *C++ Gems*, a SIGS Books imprint, Cambridge University Press, Cambridge, England (1996) ISBN 0-13570581-9.

[MEYERS98] Meyers, Scott, *Effective C++, 2nd Edition*, Addison Wesley Longman, Inc., Reading, MA (1998) ISBN 0-201-92488-9.

[MEYERS96] Meyers, Scott, *More Effective C++*, Addison Wesley Longman, Inc., Reading, MA (1996) ISBN 0-201-63371-X.

[MURRAY93] Murray, Robert B., *C++ Strategies and Tactics*, Addison Wesley Longman, Inc., Reading, MA (1993) ISBN 0-201-56382-7.

[MUSSER96] Musser, David R., and Atul Saini, *STL Tutorial and Reference Guide*, Addison Wesley Longman, Inc., Reading, MA (1996) ISBN 0-201-63398-1.

[NACKMAN94] Barton, John J., and Lee R. Nackman, *Scientific and Engineering C++, An Introduction with Advanced Techniques and Examples*, Addison Wesley Longman, Inc., Reading, MA (1994) ISBN 0-201-53393-6.

[NEIDER93] Neider, Jackie, Tom Davis, and Mason Woo, *OpenGL Programming Guide*, Addison Wesley, Inc., Reading, MA (1993) ISBN 0-201-63274-8.

[PERSON68] Person, Russell V., *Essentials of Mathematics, 2nd Edition*, John Wiley & Sons, Inc., New York, NY (1968) ISBN 0-132-84191-6.

[PLAUGER92] Plauger, P.J., *The Standard C Library*, Prentice-Hall, Englewood Cliffs, NJ (1992) ISBN 0-13-131509-9.

[SEDGEWICK88] Sedgewick, Robert, *Algorithms, 2nd Edition*, Addison Wesley Longman, Inc., Reading, MA (1988) ISBN 0-201-06673-4.

[SHAMPINE97] Shampine, L.F., R.C. Allen, Jr., and S. Pruess, *Fundamentals of Numerical Computing*, John Wiley & Sons, Inc., New York, NY (1997) ISBN 0- 471-16363-5.

[STROUSTRUP94] Stroustrup, Bjarne, *The Design and Evolution of C++*, Addison Wesley Longman, Inc., Reading, MA (1994) ISBN 0-201-54330-3.

[STROUSTRUP97] Stroustrup, Bjarne, *The C++ Programming Language, 3rd Edition*, Addison Wesley Longman, Inc., Reading, MA (1997) ISBN 0-201-88954-4.

[UPSTILL90] Upstill, Steve, *The RenderMan Companion*, Addison Wesley Longman, Inc., Reading, MA (1990) ISBN 0-201-50868-0.

[WERNECKE94] Wernecke, Josie, *The Inventor Mentor*, Addison Wesley Longman, Inc., Reading, MA (1994) ISBN 0-201-62495-8.

[YOUNG95] Young, Douglas A., *Object-Oriented Programming with C++ and OSF/ Motif, 2nd Edition*, Prentice-Hall, Englewood Cliffs, NJ (1995) ISBN 0-132- 09255-7.