

<<C++ Primer中文版>>

图书基本信息

书名：<<C++ Primer中文版>>

13位ISBN编号：9787115220172

10位ISBN编号：7115220174

出版时间：2010

出版单位：人民邮电出版社

作者：Stanley B.Lippman,Josee Lajoie,Barbara E.Moo

页数：974

译者：李师贤,蒋爱军,梅晓勇

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

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，并辅之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。

<<C++ Primer中文版>>

内容概要

C++诞生20年后，因其强大的功能、广泛的适用性和极高的效率，已经成为毋庸置疑的主流编程语言。

但是C++语言也不得不面对这样的挑战：其博大精深不仅令初学者望而生畏，而且即使是许多富于经验的老手也很难全面掌握，更有不少C++程序员一直背负着C语言的历史包袱，常常落入各种微妙难解的安全和性能陷阱。

如何使现代C++理念深入人心，使C++更加容易学习和使用，已经成为众所瞩目的关键问题。

C++ Primer是久负盛名、无可替代的C++经典著作，已经帮助全球无数程序员学会了C++。第4版完美结合了C++大师Stan B. Lippman丰富的实践经验和C++标准委员会原负责人Jos é e Lajoie对C++标准的深入理解，更加入了C++先驱Barbara E. Moo在C++教学方面的真知灼见，充分体现了C++语言的最新进展和当前的业界最佳实践。

对C++基本概念和技术全面而且权威的阐述，以及对现代C++编程风格的强调，使本书不仅是初学者的最佳C++指南，而且是中高级程序员不可或缺的参考书。

作者简介

《C++ Primer (第4版)》的三位作者都是C++领域的权威人物。

Stanley B. Lippman 微软公司 Visual C++ 团队的架构师。
他从1984年开始在贝尔实验室与C++的设计者Bjarne Stroustrup一起从事C++的设计与开发。
他还著有Inside the C++ Object Model。

Jos é e Lajoie 曾经是IBM加拿大研究中心C/C++编译器开发团队的成员，在ISO C++标准委员会工作了7年，担任过ISO核心语言工作组的主席和C++ Report杂志的专栏作家。

Barbara E. Moo 拥有25年软件经验的独立咨询顾问。
在AT&T，她与Stroustrup、Lippman一起管理过复杂的C++开发项目。
她和Andrew Koenig合著了Accelerated C++和Ruminations on C++。

书籍目录

第1章 快速入门	1.1 编写简单的C++程序	1.2 初窥输入/输出	1.2.1 标准输入与输出对象
	1.2.2 一个使用IO库的程序	1.3 关于注释	1.4 控制结构
	1.4.1 while语句	1.4.2 for语句	1.4.3 if语句
	1.4.4 for语句	1.4.5 if语句	1.4.4 读入未知数目的输入
	1.5 类的简介	1.5.1 Sales_item类	1.5.2 初窥成员函数
C++程序	小结	术语	1.6
2.1 基本内置类型	2.1.1 整型	2.1.2 浮点型	2.2 字面值常量
2.3 变量	2.3.1 什么是变量	2.3.2 变量名	2.3.3 定义对象
2.3.4 变量初始化规则	2.3.5 声明和定义	2.3.6 名字的作用域	
2.3.7 在变量使用处定义变量	2.4 const限定符	2.5 引用	2.6
typedef名字	2.7 枚举	2.8 类类型	2.9 编写自己的头文件
2.9.1 设计自己的头文件	2.9.2 预处理器的简单介绍	小结	术语
第3章 标准库类型	3.1 命名空间的using声明	3.2 标准库string类型	
3.2.1 string对象的定义和初始化	3.2.2 String对象的读写	3.2.3 string对象的操作	
3.2.4 string对象中字符的处理	3.3 标准库vector类型	3.3.1 vector对象的定义和初始化	
3.3.2 vector对象的操作	3.4 迭代器简介	3.5.1 bitset对象的定义和初始化	3.5.2 bitset对象上的操作
3.5 标准库bitset类型	3.5.1 bitset对象的定义和初始化	3.5.2 bitset对象上的操作	
小结	术语	第4章 数组和指针	4.1 数组
4.1.1 数组的定义和初始化	4.1.2 数组操作	4.2 指针的引入	4.2.1 什么是指针
4.2.2 指针的定义和初始化	4.2.3 指针操作	4.2.4 使用指针访问数组元素	
4.2.5 指针和const限定符	4.3 C风格字符串	4.3.1 创建动态数组	
4.3.2 新旧代码的兼容	4.4 多维数组	小结	术语
第5章 表达式	5.1 算术操作符	5.2 关系操作符和逻辑操作符	
5.3 位操作符	5.3.1 bitset对象或整型值的使用	5.3.2 将移位操作符用于IO	
5.4 赋值操作符	5.4.1 赋值操作的右结合性	5.4.2 赋值操作具有低优先级	
5.4.3 复合赋值操作符	5.5 自增和自减操作符	5.6 箭头操作符	
5.7 条件操作符	5.8 sizeof操作符	5.9 逗号操作符	
5.10 复合表达式的求值顺序	5.10.1 优先级	5.10.2 结合性	5.10.3 求值顺序
5.11 new和delete表达式	5.12 类型转换	5.12.1 何时发生隐式类型转换	
5.12.2 算术转换	5.12.3 其他隐式转换	5.12.4 显式转换	
5.12.5 何时需要强制类型转换	5.12.6 命名的强制类型转换		
5.12.7 旧式强制类型转换	小结	术语	第6章 语句
6.1 简单语句	6.2 声明语句	6.3 复合语句(块)	6.4 语句作用域
6.5 if语句	6.6 switch语句	6.6.1 使用switch	6.6.2 switch中的控制流
6.6.3 default标号	6.6.4 switch表达式与case标号	6.6.5 switch内部的变量定义	
6.7 while语句	6.8 for循环语句	6.8.1 省略for语句头的某些部分	
6.8.2 for语句头中的多个定义	6.9 do while语句	6.10 break语句	
6.11 continue语句	6.12 goto语句	6.13 try块和异常处理	
6.13.1 throw表达式	6.13.2 try块	6.13.3 标准异常	6.14 使用预处理器进行调试
小结	术语	第7章 函数	7.1 函数的定义
7.1.1 函数返回类型	7.1.2 函数形参表	7.2 参数传递	7.2.1 非引用形参
7.2.2 引用形参	7.2.3 vector和其他容器类型的形参	7.2.4 数组形参	7.2.5 传递给函数的数组的处理
7.2.6 main: 处理命令行选项	7.2.7 含有可变形参的函数	7.3 return语句	7.3.1 没有返回值的函数
7.3.2 具有返回值的函数	7.3.2 具有返回值的函数	7.3.3 递归	7.4 函数声明
7.5 局部对象	7.5.1 自动对象	7.5.2 静态局部对象	7.6 内联函数

7.7 类的成员函数	7.7.1 定义成员函数的函数体	7.7.2 在类外定义成员函数
7.8 重载函数	7.8.1 重载与作用域	7.8.2 函数匹配与实参转换
7.8.3 重载确定的三个步骤	7.8.4 实参类型转换	7.9 指向函数的指针
小结	第8章 标准IO库	8.1 面向对象的标准库
术语	8.2 条件状态	8.2.1 文件流
8.3 输出缓冲区的管理	8.4 文件的输入和输出	8.4.1 文件流
8.4.2 文件模式	8.4.3 一个打开并检查输入文件的程序	
对象的	第二部分 容器和算法	第9章 顺序容器
8.5 字符串流	小结	9.1 顺序容器的定义
9.1 顺序容器的定义	术语	9.1.1 容器元素的初始化
9.1.2 容器内元素的类型	9.2 迭代器和迭代器范围	9.2.1 迭代器范围
9.2.2 使迭代器失效的容器操作	9.3 顺序容器的操作	9.3.1 容器定义的类型别名
9.3.2 begin和end成员	9.3.3 在顺序容器中添加元素	9.3.4 关系操作符
9.3.5 容器大小的操作	9.3.6 访问元素	9.3.7 删除元素
9.3.8 赋值与swap	9.4 vector容器的自增长	9.5 容器的选用
9.6 再谈string类型	9.6.1 构造string对象的其他方法	9.6.2 修改string对象的其他方法
9.6.3 只适用于string类型的操作	9.6.4 string类型的查找操作	
9.6.5 string对象的比较	9.7 容器适配器	9.7.1 栈适配器
9.7.2 队列和优先级队列	小结	第10章 关联容器
术语	10.1 引言	10.1.1 map对象的定义
10.2 关联容器	10.3 map类型	10.3.1 map对象的定义
10.3.2 map定义的类型	10.3.3 给map添加元素	10.3.4 使用下标访问map对象
10.3.5 map::insert的使用	10.3.6 查找并读取map中的元素	10.3.9
10.3.7 从map对象中删除元素	10.3.8 map对象的迭代遍历	
10.3.9 “单词转换” map对象	10.4 set类型	10.4.1 set容器的定义和使用
10.4.2 创建“单词排除”集	10.5 multimap和multiset类型	10.5.1 元素的添加和删除
10.5.2 在multimap和multiset中查找元素	10.6 容器的综合应用：文本查询程序	10.6.1 查询程序的设计
10.6.2 TextQuery类	10.6.3 TextQuery类的使用	10.6.4 编写成员函数
10.6.4 编写成员函数	小结	术语
11.1 概述	11.2 初窥算法	11.2.1 只读算法
11.2.2 写容器元素的算法	11.2.3 对容器元素重新排序的算法	11.3 再谈迭代器
11.3.1 插入迭代器	11.3.2 istream迭代器	11.3.3 反向迭代器
11.3.4 const迭代器	11.3.5 五种迭代器	11.4 泛型算法的结构
11.4.1 算法的形参模式	11.4.2 算法的命名规范	11.5 容器特有的算法
11.4.2 算法的命名规范	第三部分 类和数据抽象	第12章 类
11.5 容器特有的算法	12.1 类的定义	12.1.1 类定义：扼要重述
12.1 类的定义	12.1.2 数据抽象和封装	12.1.3 关于类定义的更多内容
12.1.1 类定义：扼要重述	12.1.4 类声明与类定义	12.1.5 类对象
12.1.2 数据抽象和封装	12.2 隐含的this指针	12.3 类作用域
12.1.3 关于类定义的更多内容	12.4 构造函数	12.4.1 构造函数初始化式
12.1.4 类声明与类定义	12.4.2 默认实参与构造函数	12.4.3 默认构造函数
12.1.5 类对象	12.4.4 隐式类类型转换	12.4.5 类成员的显式初始化
12.2 隐含的this指针	12.5 友元	12.6 static类成员
12.3 类作用域	12.6.1 static成员函数	12.6.2 static数据成员
12.4 构造函数	12.6.2 static数据成员	小结
12.4.1 构造函数初始化式	第13章 复制控制	术语
12.4.2 默认实参与构造函数	13.1 复制构造函数	13.1.1 合成的复制构造函数
12.4.3 默认构造函数	13.1.2 定义自己的复制构造函数	13.1.3 禁止复制
12.4.4 隐式类类型转换	13.2 赋值操作符	13.2 赋值操作符
12.4.5 类成员的显式初始化	13.3 析构函数	13.3 析构函数
12.5 友元	13.4 消息处理示例	13.4 消息处理示例
12.6 static类成员	13.5 管理指针成员	13.5 管理指针成员
12.6.1 static成员函数	13.5.1 定义智能指针类	13.5.2 定义值型类
12.6.2 static数据成员	13.5.2 定义值型类	小结
小结	第14章 重载操作符与转换	术语
术语	14.1 重载操作符的定义	14.2 输入和输出操作符
第13章 复制控制	14.2 输入和输出操作符	14.2.1 输出操作符的重载
13.1 复制构造函数	14.3 算术操作符和关系操作符	14.3.1 相等操作符
13.1.1 合成的复制构造函数	14.4 赋值操作符	14.4.1 相等操作符
13.1.2 定义自己的复制构造函数	14.5 下标操作符	14.5 下标操作符
13.1.3 禁止复制	14.6 成员访问操作符	14.6 成员访问操作符
13.2 赋值操作符	14.7 自增操作符和自减操作符	14.7 自增操作符和自减操作符
13.3 析构函数	14.8 调用操作符和函数对	14.8 调用操作符和函数对
13.4 消息处理示例		
13.5 管理指针成员		
13.5.1 定义智能指针类		
13.5.2 定义值型类		
小结		
术语		
第14章 重载操作符与转换		
14.1 重载操作符的定义		
14.2 输入和输出操作符		
14.2.1 输出操作符的重载		
14.3 算术操作符和关系操作符		
14.3.1 相等操作符		
14.4 赋值操作符		
14.5 下标操作符		
14.6 成员访问操作符		
14.7 自增操作符和自减操作符		
14.8 调用操作符和函数对		

<<C++ Primer中文版>>

象	14.8.1 将函数对象用于标准库算法	14.8.2 标准库定义的函数对象
用	14.8.3 函数对象的函数适配器	14.9 转换与类类型
类的实参	14.9.2 转换操作符	14.9.1 转换为什么有用
面向对象编程与泛型编程	14.9.5 重载、转换和操作符	14.9.3 实参匹配和转换
定义基类和派生类	14.9.4 重载确定和	14.9.4 重载确定和
派生类	14.9.5 重载、转换和操作符	小结
数	第15章 面向对象编程	术语
数中的虚函数	15.1 面向对象编程：概述	第四部分 面
用域	15.2.1 定义基类	15.2
再谈文本查询示例	15.2.2 protected成员	15.2.3
eval函数	15.2.4 virtual与其他成员函数	15.2.5 公用、私有和受保护的继承
	15.2.6 友元关系与继承	15.3 转换与继承
	15.3.1 派生类到基类的转换	15.4 构造函数
	15.4.1 基类构造函数和复制控制	15.4.2 派生类构造函数
	15.4.3 复制控制和继承	15.4.4 虚析构函数
	15.4.4 虚析构函数	15.4.5 构造函数和析构函
	15.5 继承情况下的类作用域	15.5.1 名字查找在编译时发生
	15.5.2 名字冲突与继承	15.5.2 名字查找在编译时发生
	15.5.3 作用域与成员函数	15.5.3 作用域与成员函数
	15.5.4 虚函数与作用域	15.5.4 虚函数与作用域
	15.6 纯虚函数	15.7 容器与继承
	15.8.1 指针型句柄	15.8 句柄类与继承
	15.8.2 复制未知类型	15.8.1 指针型句柄
	15.9.1 面向对象的解决方案	15.8.2 复制未知类型
	15.9.2 值型句柄	15.8.3 句柄的使用
	15.9.3 Query_base类	15.9.1 面向对象的解决方案
	15.9.4 Query句柄类	15.9.2 值型句柄
	15.9.5 派生类	15.9.3 Query_base类
	小结	15.9.4 Query句柄类
	术语	15.9.5 派生类
	第16章 模板与泛型编程	15.9.6
	16.1 模板定义	16.1 模板定义
	16.1.1 定义函数模板	16.1.1 模板形参
	16.1.2 定义类模板	16.1.2 模板形参
	16.1.3 模板形参	16.1.3 模板形参
	16.1.4 模板类型形参	16.1.4 模板类型形参
	16.1.5 非类型模板形参	16.1.5 非类型模板形参
	16.1.6 编写泛型程序	16.1.6 编写泛型程序
	16.2 实例化	16.2.1 模板实参推断
	16.2.1 模板实参推断	16.2.2 函数模板的显式实参
	16.2.2 函数模板的显式实参	16.2.2 函数模板的显式实参
	16.3 模板编译模型	16.4.1 类模板成员函数
	16.4 类模板成员	16.4.1 类模板成员函数
	16.4.1 类模板成员函数	16.4.2 非类型形参的模板实参
	16.4.2 非类型形参的模板实参	16.4.3 类模板中的友元声明
	16.4.3 类模板中的友元声明	16.4.4 Queue
	16.4.4 Queue	16.4.5 成员模板
	16.4.5 成员模板	16.4.6 完整的Queue类
	16.4.6 完整的Queue类	16.4.6 完整的Queue类
	16.5 一个泛型句柄类	16.5.1 定义句柄类
	16.5.1 定义句柄类	16.5.1 定义句柄类
	16.6 模板特化	16.6.1 函数模板的特化
	16.6.1 函数模板的特化	16.6.2
	16.6.2	16.6.2
	16.6.3 特化成员而不特化类	16.6.3 特化成员而不特化类
	16.6.4 类模板的部分特化	16.6.4 类模板的部分特化
	16.7 重载与函数模板	小结
	小结	术语
	术语	第五部分 高级主题
	第17章 用于大型程序的工	第17章 用于大型程序的工
	17.1 异常处理	17.1.1 抛出类类型的异常
	17.1.1 抛出类类型的异常	17.1.2 栈
	17.1.2 栈	17.1.2 栈
	17.1.3 捕获异常	17.1.3 捕获异常
	17.1.4 重新抛出	17.1.4 重新抛出
	17.1.5 捕获所有异常的处理	17.1.5 捕获所有异常的处理
	17.1.6 函数测试块与构造函数	17.1.6 函数测试块与构造函数
	17.1.7 异常类层次	17.1.7 异常类层次
	17.1.8	17.1.8
	17.1.9 auto_ptr类	17.1.9 auto_ptr类
	17.1.10 异常说明	17.1.10 异常说明
	17.1.11 函数	17.1.11 函数
	17.2 命名空间	17.2.1 命名空间的定义
	17.2.1 命名空间的定义	17.2.1 命名空间的定义
	17.2.2 嵌套命名空间	17.2.2 嵌套命名空间
	17.2.3 未命名的命名空间	17.2.3 未命名的命名空间
	17.2.4 命名空间成员的使用	17.2.4 命名空间成员的使用
	17.2.5 类、命名空间和作用域	17.2.5 类、命名空间和作用域
	17.2.6 重载与命名空间	17.2.6 重载与命名空间
	17.2.7 命名空间与	17.2.7 命名空间与
	17.3 多重继承与虚继承	17.3.1 多重继承
	17.3.1 多重继承	17.3.1 多重继承
	17.3.2 转换与多个	17.3.2 转换与多个
	17.3.3 多重继承派生类的复制控制	17.3.3 多重继承派生类的复制控制
	17.3.4 多重继承下的类作用域	17.3.4 多重继承下的类作用域
	17.3.5 虚继承	17.3.5 虚继承
	17.3.6 虚基类的声明	17.3.6 虚基类的声明
	17.3.7 特殊的初始化语义	17.3.7 特殊的初始化语义
	小结	术语
	术语	第18章 特殊工具与技术
	第18章 特殊工具与技术	18.1 优化内存分配
	18.1.1 C++中的内存分配	18.1.1 优化内存分配
	18.1.2 allocator类	18.1.2 allocator类
	18.1.3 operator new函数	18.1.3 operator new函数
	18.1.4 定位new表达式	18.1.4 定位new表达式
	18.1.5 显式析构函数的调用	18.1.5 显式析构函数的调用
	18.1.6 类特定的new和delete	18.1.6 类特定的new和delete
	18.1.7 一个内存分配器基类	18.1.7 一个内存分配器基类
	18.2 运行时类	18.2 运行时类
	18.2.1 dynamic_cast操作符	18.2.1 dynamic_cast操作符
	18.2.2 typeid操作符	18.2.2 typeid操作符
	18.2.3	18.2.3
	18.2.4 type_info类	18.2.4 type_info类
	18.3 类成员的指针	18.3.1 声明成员
	18.3.1 声明成员	18.3.1 声明成员
	18.3.2 使用类成员的指针	18.3.2 使用类成员的指针
	18.4 嵌套类	18.4.1 嵌套类的实现
	18.4.1 嵌套类的实现	18.4.1 嵌套类的实现
	18.4.2 嵌套类作用域中的名字查找	18.4.2 嵌套类作用域中的名字查找
	18.5 联合：节省空间的类	18.5 联合：节省空间的类
	18.6	18.6
	18.7 固有的不可移植的特征	18.7.1 位域
	18.7.1 位域	18.7.2 volatile限定
	18.7.2 volatile限定	18.7.2 volatile限定

符
索引

18.7.3 链接指示 : extern "C"
C++编程规范

小结

术语

附录 标准库

章节摘录

第一部分 基本语言 第3章 标准库类型 第2章所涉及的类型都是低级数据类型：这些类型表示数值或字符的抽象，并根据其具体机器表示来定义。

除了这些在语言中定义的类型外，C++标准库还定义了许多更高级的抽象数据类型(abstract data type)。

之所以说这些标准库类型是更高级的，是因为其中反映了更复杂的概念；之所以说它们是抽象的，是因为我们在使用时不需要关心它们是如何表示的，只需知道这些抽象数据类型支持哪些操作就可以了。

。

两种最重要的标准库类型是string和vector。

string类型支持长度可变的字符串，vector可用于保存一组指定类型的对象。

说它们重要，是因为它们在C++定义的基本类型基础上作了一些改进。

第4章还将学习类似于标准库中string和vector类型的语言级构造，但标准库的string和vector类型可能更灵活，且不易出错。

媒体关注与评论

“在遇到无法解决的问题时，我总会求助于C++ Primer一书。

”——Bruce Eckel，“编程思想”系列图书作者 “众所周知，C++ Primer是学习C++最理想的参考书之一，适用于各种水平的C++程序员。

第4版不但保持了这种传统，而且有了很大改善。

”——Steve Vinoski，IONA科技公司首席工程师，CORBA与C++权威 “（本书）不但能让初学者迅速入门，而且是用优秀的编程实践引导他们入门。

”——Nevin Liber，资深C++开发人员 “如果你想仅通过一本书就彻底学会C++并能很好地运用，这本书值得购买。

”——Paul M. Dubuc，软件开发人员 “无论如何，这是我读过的最好的程序设计书……如果你是编程新手，这本书可以让你以最好的方式开始编程，并为你介绍了最佳的编程实践。

”——Alberto Moriconi “……这是学习C++语言极好的一本书。

在介绍面向对象编程以及C++类的设计和实现之前，先介绍了C++标准库，这样很容易很快就编写出有用的程序。

这本书的组织方式、写作思路和风格都很出色。

”——James M. Scott

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>