

100%

内容丰富、权威

全方位领略C++的强大功能，轻松掌握所有相关知识

通过由浅入深、由易到难的典型实例，快速精通C++编程

运用C++强大灵活的类机制编写一流的面向对象程序

利用标准C++库进行简单、高效的应用程序开发

宝典丛书

100万

标准C++ 编程

[美] Raphael Pender

苏剑 等编著

本书附带的光盘中包含所有例子程序的源代码和一个名为Quincy的集成开发环境

宝典



电子工业出版社
Publishing House of Electronics Industry
<http://www.phei.com.cn>

计算机“宝典”丛书

标准 C++ 编程宝典

[美] Raphael Pender 苏剑 等编著

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

标准 C++ 是 C++ 的国际性标准版本, 本书以此标准为基础, 对 C++ 进行了全面介绍。

本书从最基本的内容讲起, 对 C++ 的强大功能进行了详细介绍; 运用 C++ 的最新功能, 详细论述了面向对象编程的思想; 同时, 还介绍了 C++ 和 C 的不同之处。本书自成体系, 是 C++ 语言的完整教程, 不要要求读者预先掌握 C 语言。本书针对那些比较高深难懂的内容也做了深入浅出的介绍, 比如模板 (包括函数模版、类模版和标准模板库)、名字空间以及运行时类型信息 (RTTI) 等, 这些功能对于开发大型复杂系统的程序设计人员来说是非常重要的。因此, 无论是新手还是有经验的程序员, 都可以从本书获取所需要的知识。

本书内容综合、全面, 叙述流畅、清晰, 实例典型、丰富, 主要针对那些既不懂 C 也不懂 C++、但又希望学习 C++ 的程序员, 是一本从入门到提高的经典书籍。

图书在版编目(CIP)数据

标准 C++ 编程宝典 / (美) 彭德 (Pender,R.), 苏剑等编著. —北京: 电子工业出版社, 2005.1
(计算机“宝典”丛书)
ISBN 7-121-00678-2

I.标... II.①彭...②苏... III.C 语言—程序设计 IV.TP312

中国版本图书馆 CIP 数据核字 (2004) 第 127600 号

责任编辑: 牛晓丽

排版制作: 华信卓越公司制作部

印 刷: 北京东光印刷厂

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编: 100036

经 销: 各地新华书店

开 本: 787 × 1092 1 / 16 印张: 38.5 字数: 1084 千字

印 次: 2005 年 1 月第 1 次印刷

定 价: 69.00 元 (含光盘一张)

凡购买电子工业出版社的图书, 如有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系。联系电话: (010) 68279077。质量投诉请发邮件到 zltts@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

编者序

C++ 语言不仅保持了 C 语言功能强、目标代码效率高、可移植性好的优点，而且还提供了面向对象的程序设计能力。在硬件技术飞速发展的今天，人们对软件功能的要求也越来越高。利用面向对象的程序设计技术实现软件重用，是加速软件开发进程的根本途径。C++ 已经成为当今的主流程序设计语言，在 ANSI/ISO 国际性标准推出之后，更是得到了飞速的发展。本书在此国际标准的基础上介绍 C++ 语言。

本书从最基本的内容讲起，对 C++ 的强大功能进行了详细介绍，并运用 C++ 的最新功能，详细论述了面向对象编程的思想。本书针对那些比较高深难懂的内容也做了深入浅出的介绍，比如模板（包括函数模版、类模版和标准模板库）、名字空间以及运行时类型信息（RTTI）等，这些功能对于开发大型复杂系统的程序设计人员来说是非常重要的。

本书附带一张光盘，其中包含了书中的所有例子程序；同时，还提供了一个基于 Windows 操作系统的 C++ 编译系统，可以用来编译和运行本书中的例子程序。读者可以一边看书，一边运行例子程序，从而加深理解。

本书自成体系，是 C++ 语言的完整教程，不要求读者预先掌握 C 语言。因此，无论对于新手还是有经验的程序员，本书都是其学习和提高 C++ 编程技巧的很好途径。本书内容全面，叙述流畅，实例典型、丰富，一定会使读者获益匪浅。

参加本书编写工作的人员有 Raphael Pender、苏剑、郑香玉、牛晶波、薛引弟、赵洪谊、姜海丽、李晓娟等，全书由 Raphael Pender 和苏剑统稿。

由于编者水平所限，疏漏不当之处在所难免，敬请广大读者批评指正。

编者
2004 年 11 月

前 言

目前，C++已成为全世界专业编程人员的首选语言。它有公认的标准，许多编译器都支持该标准中的绝大多数特性。本书对C++语言的介绍遵循ANSI/ISO国际标准。

读者对象

本书适用于没有C语言基础、但希望学习C++语言的初学者，也适用于具有一定C语言编程经验的用户。

没有C语言基础的用户

过去几年中出版的大多数关于C++的书籍，都假设读者有C语言基础。按照传统的观念，学习C++之前，应该先学习C。现在，这种观念已经过时了。如果先学习了C风格的程序设计，那么可能会先入为主，带有一些不必要、不合理的偏见。如果还没有学习C，那么就应该直接学习C++，这是目前比较流行的观念。

本书自成体系，是C++语言的完整教程，不要求读者预先掌握C语言。到现在为止，大多数C程序员都已经学习过C++了。因此，本书的主要读者对象是那些既不懂C也不懂C++、但又希望学习C++的程序员。当然，C程序员也可以使用本书学习C++。

C程序员

本书第2章到第8章的大部分内容都是关于C语言的。在这几个章节的许多地方，都可以看到C++语言与C语言的细微差别。例如，第2章中介绍了C++控制台输入/输出*iostream*对象，这些对象是C++程序设计中所使用的，而在C语言中，则使用*getchar()*，*putchar()*，*gets()*，*puts()*，*printf()*和*scanf()*等标准函数；第4章中讲述了*goto*语句，它与C中的*goto*语句不同；C++中的*void*指针如果不经强制类型转换，就不能够赋值给一个指针类型变量；C++程序中的*main()*函数不能够被递归调用；对*const*对象必须进行初始化；变量不能被隐含声明为*int*类型；等等。本书是将C++作为一门新的程序设计语言来介绍的，因此，C程序员不应该完全依赖于已经掌握的C语言的相关知识，而忽略从第2章到第8章所讲述的内容。

例子程序

本书在讲述C++的过程中提供了一系列例子程序。每一个程序都包含C++源代码，可以进行编译和运行。如果希望从这个教程中获得最大的收益，那么应该一边看书，一边编译、运行这些例子程序。

本书的例子程序按照内容顺序编排，先从比较简单的概念开始，循序渐进，一直到比较复杂的内

容。后面的程序建立在前面程序所介绍知识的基础之上，因此应该按照例子程序出现的先后顺序阅读。

本书中的每个例子程序都对C++的某个特定功能进行了说明。这些例子是可以进行独立编译和链接的完整程序。

本书附带光盘中提供的编译器

本书附带的光盘中包含了 Quincy 99，它是一个基于 Windows（包括 Windows 9X，Windows 2000，Windows NT 和 Windows XP）的 C++ 编译系统，可以用来编译和运行本书中的例子程序。Quincy 99 可引导读者使用本书中的所有例子程序，其中程序的编号与书中一致，可以根据编号从列表中进行选择。

本书的结构

本书分为 5 大部分，包括 28 章和一个附录，下面分别进行介绍。

第 1 章简单介绍了 C++ 编程概念，指明了读者需要预先掌握的知识以及使用光盘中的例子程序所需要了解的内容。在本章中，对 C++ 的历史做了简要介绍，对 C++ 编程语言进行了概述，还介绍了主函数 `main()`（它是 C++ 程序的入口）。

第 2 章引导读者开始编写简单的程序。本章对 `main()` 函数做了更详尽的介绍，并且介绍了如何为代码添加注释，如何把标准库的头文件包含在程序中，以及如何实现简单的控制台输出以观察程序运行结果。本章还讲述了 C++ 的表达式和赋值语句，以及如何从键盘读入数据并在屏幕上显示。

第 3 章集中讨论函数，讲述了如何在 C++ 程序中声明、定义和调用函数，如何向函数传递参数，如何从函数返回值，以及如何书写 C++ 函数的语句块结构。此外，本章还介绍了如何把 C++ 的程序模块与用其他语言（比如 C 语言）编写的程序模块链接起来。

第 4 章讲述程序流程控制，详细介绍了 `if...else`，`switch...case`，`do...while`，`for`，`break`，`continue` 和 `goto` 语句。

第 5 章详细讨论了 C++ 数据类型，介绍了各种类型的作用域，以及如何用基本数据类型构造数组、结构和联合等聚合数据类型。

第 6 章讲述了如何在程序中使用指针和引用型变量，同时还介绍了有关递归的概念。

第 7 章讨论了标准 C 库函数中的一部分内容，介绍了如何使用头文件实现在程序中调用标准函数，还介绍了字符串函数、内存分配函数等。

第 8 章介绍了 C++ 的预处理功能，包括定义宏、编写条件编译表达式来控制程序编译过程。

第 9 章对结构进行扩展并引入了类机制，介绍了内部数据类型和用户自定义数据类型的数据抽象特性，还介绍了数据成员、成员函数和访问控制符。

第 10 章介绍了如何使用函数模板创建通用函数，用于处理不同类型的数据；还介绍了关于模板的高级应用，例如模板重载。

第 11 章深入讨论了类的相关内容，包括构造函数、析构函数、转换函数、赋值函数、类对象数组、类对象的内存分配，等等。

第 12 章讨论了重载运算符，介绍了如何为类重载各种 C++ 运算符，以便使类对象模仿内部数据类型行为。

第13章介绍类继承。使用类的继承，可以建立面向对象的类层次结构，类层次结构由基类和派生类构成。本章还介绍了多态的概念。

第14章讲述了多重继承，它是一种语言特性，允许派生类从多个基类中继承属性。

第15章讨论了类模板，它是C++语言的特性，可以根据参数生成类。

第16章描述了面向对象编程的技术，包括数据抽象、封装、继承和多态性等。

第17章讨论了标准C++的库。对iostream控制台输入/输出类进行了扩充介绍，还介绍了标准字符串类和复数类，以及标准模板库(STL)的容器类集合。

第18章介绍了iostream和stringstream类的管理和格式化。

第19章详细介绍了C++流的概念和磁盘文件操作。在读写文件时可以使用不同的流类。本章还介绍了如何处理文本文件和二进制文件。

第20章介绍了标准模板库(STL)中定义的各种类型的类，包括序列、关联式容器、算法和迭代器。

第21章讲述了如何使用序列进行编程，序列可以实现链表、队列、堆栈、向量等。

第22章介绍了如何使用关联式容器，关联式容器可以实现映射、集合和位集合。

第23章讨论了函数库中的通用算法，这些算法可以使程序处理定义在STL中的对象数据。处理过程包括计数、排序和分组。

第24章介绍了STL迭代器——一种特殊类型的指针，以及如何用迭代器访问存储在容器中的元素的信息。

第25章详细介绍了异常处理，包括如何使程序以有序的方式处理异常。

第26章介绍了名字空间，包括如何定义名字空间、名字空间的范围、无名名字空间、名字空间的别名。

第27章介绍了C++的运行时类型信息(RTTI)和新风格类型转换，包括动态类型转换、静态类型转换和常类型转换。

第28章讲述了C++中的locale类，包括如何使用locale类实现应用程序的国际化，如何对locale进行设置，以及如何根据locale要求的形式显示信息。

附录介绍了本书附带的光盘中的内容。

本书中的约定

为了清晰起见，本书中的程序代码及输出结果均使用Courier New字体。

图标

在本书中，读者会看到各种专门的图形符号(即图标)，这些图标表示该部分内容特别重要或特别值得注意。本书主要使用如下几个图标：



该图标提供相关内容的补充信息和值得注意的事项，这对内容的理解是至关重要的。



该图标指示在其他章节有与本主题相关的信息。



该图标告诉读者例子程序在本书附带光盘上的位置。

目 录

前言

第 1 部分 C++ 语言	1
第 1 章 C++ 语言概述	2
1.1 C 语言和 C++ 语言的历史	2
1.2 C 与 C++ 的比较	2
1.3 C++ 简述	3
1.4 main() 函数	5
1.5 小结	6
第 2 章 编写简单的 C++ 程序	8
2.1 学习编写第一个程序	7
2.1.1 #include 指令	8
2.1.2 空行	8
2.1.3 main() 函数声明	8
2.1.4 main() 函数的语句块	8
2.1.5 源代码注释	9
2.1.6 控制台输出	9
2.1.7 return 语句	9
2.1.8 语句块的结束	9
2.2 标识符	9
2.3 关键字	10
2.4 标准输出流	10
2.5 变量	11
2.5.1 布尔型变量	11
2.5.2 字符型变量	12
2.5.3 wchar_t 型变量	13
2.5.4 整型变量	14
2.5.5 浮点型变量	15
2.6 常量	15
2.6.1 字符型常量	16
2.6.2 转义字符	16
2.6.3 整型常量	17
2.6.4 浮点型常量	17
2.6.5 地址常量	17
2.6.6 字符串常量	18
2.7 表达式	18
2.8 赋值语句	19
2.9 用逗号分隔的声明语句	20
2.10 表达式中的运算符	21
2.10.1 算术运算符	21
2.10.2 逻辑运算符	22

2.10.3	位逻辑运算符	23
2.10.4	位移运算符	24
2.10.5	关系运算符	24
2.10.6	自增运算符和自减运算符	25
2.10.7	赋值运算符	27
2.10.8	复合赋值运算符	28
2.10.9	条件运算符	29
2.10.10	逗号运算符	31
2.11	运算符的优先级与结合性	32
2.11.1	结合性	32
2.11.2	优先级	33
2.12	表达式不参加运算的情况	33
2.13	初始化	34
2.14	类型转换	35
2.15	控制台输入与输出	36
2.15.1	标准输出流	36
2.15.2	格式化输出	37
2.15.3	标准错误流	38
2.15.4	标准输入流	39
2.16	小结	40
第3章	函数	42
3.1	函数	42
3.2	实际参数和形式参数	43
3.3	函数原型	43
3.3.1	无标识符的参数类型	43
3.3.2	一个典型的函数原型	44
3.3.3	返回类型为 void 的函数	44
3.3.4	参数列表为空的函数	44
3.3.5	参数列表可变的函数	44
3.3.6	既不带参数也没有返回值的函数	45
3.3.7	标准库函数的原型	45
3.3.8	不需要原型的函数	45
3.4	定义和调用函数	45
3.5	从函数返回	46
3.6	函数的返回值	48
3.7	传递和使用参数	49
3.7.1	标识符的作用域	50
3.7.2	通过函数调用来进行初始化	50
3.7.3	初始化的顺序	50
3.7.4	具有多个参数的函数	51
3.7.5	用做实际参数的函数调用	52
3.7.6	值传递	52
3.7.7	实际参数和返回值的类型转换	52
3.8	无名形式参数	52
3.9	默认的函数参数	53
3.10	内联函数	54
3.11	递归	55



3.12	函数的重载	55
3.12.1	为不同的操作而重载	55
3.12.2	为不同的格式而重载	57
3.13	安全链接	58
3.14	链接说明	58
3.15	小结	62
第 4 章	C++ 的程序流程控制	63
4.1	语句块	63
4.1.1	语句块的嵌套深度	63
4.1.2	缩进风格	63
4.2	分支语句	64
4.2.1	if 语句	64
4.2.2	if...else 语句	67
4.2.3	else if 语句	67
4.2.4	switch...case 语句	69
4.2.5	在 if 语句的条件表达式中进行变量声明	70
4.3	循环语句	71
4.3.1	while 语句	71
4.3.2	do...while 语句	72
4.3.3	for 语句	74
4.3.4	在 for 语句的条件表达式中声明变量	76
4.4	循环控制语句	76
4.4.1	break 语句	76
4.4.2	continue 语句	77
4.5	跳转语句: goto	78
4.5.1	goto 语句的错误用法	79
4.5.2	C++ 语言的 goto 语句和 C 语言的 goto 语句的比较	80
4.5.3	改正错误的 goto 语句	80
4.5.4	是否应该使用 goto 语句	82
4.6	小结	82
第 5 章	数据类型	83
5.1	标识符的作用域	83
5.1.1	全局作用域	83
5.1.2	局部作用域	84
5.1.3	全局作用域解析符	86
5.1.4	文件作用域	87
5.1.5	变量的作用域和生存周期	88
5.2	存储类	89
5.2.1	自动存储类	89
5.2.2	静态存储类	90
5.2.3	外部存储类	91
5.2.4	寄存器存储类	92
5.3	初始化的默认值	93
5.4	类型修饰符	94
5.4.1	const 类型修饰符	94
5.4.2	volatile 类型修饰符	95

5.5	用户定义的数据类型	95
5.5.1	声明结构	96
5.5.2	定义结构变量	96
5.5.3	引用结构成员	96
5.5.4	初始化结构变量	97
5.5.5	结构的嵌套	98
5.5.6	向函数传递结构和从函数返回结构	99
5.6	联合数据类型	100
5.6.1	初始化联合	102
5.6.2	匿名联合	103
5.6.3	枚举常量	104
5.7	数组	105
5.7.1	声明数组	105
5.7.2	利用下标访问数组元素	105
5.7.3	初始化数组	106
5.7.4	结构数组	107
5.7.5	多维数组	108
5.7.6	字符型数组：一个特例	109
5.8	小结	110
第 6 章	地址、指针和引用型变量	111
6.1	地址和指针	111
6.1.1	指向内部数据类型的指针	111
6.1.2	指针的算术运算	113
6.1.3	指针和数组的关系	115
6.1.4	指向结构的指针	118
6.1.5	作为函数参数的指针	119
6.1.6	以地址作为函数的返回值	122
6.1.7	指向函数的指针	124
6.1.8	指向指针的指针	127
6.1.9	指向指针数组的指针	128
6.1.10	指向 const 型变量的指针	129
6.1.11	const 型指针变量	131
6.1.12	void 型指针	132
6.2	sizeof 运算符	134
6.3	强制类型转换	136
6.3.1	C 风格的强制类型转换	136
6.3.2	typedef	136
6.4	命令行参数：argc 和 argv	138
6.5	程序的内存结构	139
6.5.1	堆	139
6.5.2	new 运算符和 delete 运算符	139
6.5.3	为大小固定的数组分配内存	140
6.5.4	为动态数组分配内存	141
6.5.5	堆耗尽时的处理	142
6.5.6	栈	142



6.6	递归	143
6.6.1	一个简单的递归函数	143
6.6.2	一个利用递归实现的计算器程序	145
6.7	引用型变量	148
6.7.1	引用是一个别名	149
6.7.2	初始化引用	150
6.7.3	用引用来简化复杂的表示法	151
6.7.4	用做函数形参的引用	152
6.7.5	以引用方式调用函数	154
6.7.6	const 型引用形参	155
6.7.7	作为函数返回值的引用	155
6.7.8	以 const 型引用作为返回值	157
6.7.9	返回指向自变量的引用	157
6.7.10	引用和指针	157
6.8	小结	158
第 7 章	库函数	159
7.1	<cassert>头文件	160
7.2	<cctype>头文件	161
7.3	<cerrno>头文件	162
7.4	<cmath>头文件	163
7.5	<csetjmp>头文件	164
7.6	<cstdarg>头文件	166
7.7	<cstdio>头文件	167
7.8	<cstdlib>头文件	168
7.8.1	数字函数	168
7.8.2	内存管理函数	168
7.8.3	系统函数	168
7.8.4	随机数发生器函数	169
7.9	<cstring>头文件	170
7.10	<ctime>头文件	172
7.11	小结	173
第 8 章	预处理程序	174
8.1	预处理指令	174
8.2	包含文件	174
8.3	宏	175
8.3.1	#define 预处理指令	175
8.3.2	带参数的 #define 预处理指令	176
8.3.3	格式和对齐	181
8.3.4	# 字符串化运算符	182
8.3.5	## 符号连接运算符	182
8.3.6	#undef 预处理指令	184
8.4	条件编译指令	184
8.4.1	#if 预处理指令	184
8.4.2	#endif 预处理指令	184
8.4.3	#if defined 预处理指令	185
8.4.4	#ifdef 和 #ifndef 预处理指令	186

8.4.5	#else 预处理指令	186
8.4.6	#elif 预处理指令	186
8.4.7	#error 预处理指令	187
8.5	其他预处理指令	188
8.5.1	#line 预处理指令	188
8.5.2	#pragma 预处理指令	188
8.6	小结	188
第 9 章	类与结构	189
9.1	C++ 的类	189
9.2	数据类型的属性	189
9.2.1	数据表示	189
9.2.2	实现	190
9.2.3	行为	190
9.2.4	接口	190
9.3	用户定义的数据类型	190
9.3.1	抽象	190
9.3.2	对象是什么	190
9.3.3	数据抽象和面向过程编程	190
9.3.4	数据抽象和封装	192
9.4	带有函数的结构	192
9.4.1	向结构添加函数	192
9.4.2	同一结构的多个实例	194
9.4.3	具有相同名称函数的不同结构	195
9.5	访问控制符	197
9.6	是滞应该使用面向过程编程	197
9.7	结构和类的比较	198
9.8	联合	198
9.9	小结	198
第 10 章	函数模板	199
10.1	函数模板基础知识	199
10.2	函数的定制	201
10.3	利用模板进行排序	204
10.4	小结	206
第 2 部分	使用类	207
第 11 章	C++ 类	208
11.1	设计类	208
11.1.1	对类进行声明	208
11.1.2	类成员	210
11.1.3	类成员的可见性	210
11.1.4	类的数据成员	210
11.1.5	初始化类对象	210
11.1.6	类的成员函数	211
11.1.7	面向对象的类的设计	211
11.1.8	类对象的作用域	212
11.1.9	内联函数	212

11.2	构造函数	214
11.2.1	具有默认参数的构造函数	214
11.2.2	默认构造函数	215
11.2.3	重载构造函数	215
11.3	析构函数	217
11.4	类的转换	217
11.4.1	转换函数	217
11.4.2	对类进行转换	221
11.4.3	调用转换函数	223
11.4.4	转换发生的情形	224
11.4.5	显式构造函数	227
11.4.6	表达式中的转换	227
11.5	私有数据成员	228
11.5.1	取值成员函数和赋值成员函数	228
11.5.2	常量成员函数	230
11.5.3	经过改进的成员转换函数	230
11.6	友元	231
11.6.1	友元类	231
11.6.2	隐式构造函数	233
11.6.3	预引用	233
11.6.4	显式友元预引用	234
11.6.5	友元函数	235
11.6.6	匿名对象	236
11.6.7	非类成员的友元函数	237
11.7	析构函数	238
11.8	重载赋值运算符函数	240
11.9	this 指针	243
11.9.1	用 this 指针作为返回值	243
11.9.2	在链表中使用 this 指针	246
11.10	类对象数组	248
11.10.1	类对象数组和默认构造函数	249
11.10.2	类对象数组和析构函数	251
11.11	静态成员	252
11.11.1	静态数据成员	252
11.11.2	静态成员函数	254
11.11.3	公有静态成员	256
11.12	类和堆	256
11.12.1	构造函数和 new 运算符、析构函数和 delete 运算符	256
11.12.2	堆和类数组	257
11.12.3	在类内部重载 new 和 delete 运算符	260
11.12.4	异常检测	262
11.12.5	重载 new 和 delete 运算符函数中的异常处理	262
11.12.6	重载 new[] 和 delete[] 运算符	262
11.13	复制构造函数	264
11.14	类中的引用	267
11.15	构造函数的参数初始化表	269
11.16	关于 const 修饰符的简单说明	270

11.17	可变数据成员	272
11.18	对类的源文件和目标文件的管理	273
11.18.1	头文件中的类声明	273
11.18.2	库中的类成员函数	273
11.19	小结	273
第 12 章	运算符重载	274
12.1	重载运算符的时机	274
12.2	重载运算符举例	275
12.3	重载运算符的规则	275
12.4	双目算术运算符	276
12.5	类成员运算符函数	276
12.6	非类成员运算符函数	278
12.7	关系运算符	280
12.8	其他赋值运算符	282
12.9	自增运算符和自减运算符	284
12.10	单目正和单目负运算符	286
12.11	下标运算符	287
12.12	成员指针运算符	288
12.13	小结	290
第 13 章	类继承	292
13.1	继承	292
13.2	进行继承的原因	294
13.3	设计定制的数据抽象类	294
13.3.1	基类	295
13.3.2	旨在提高效率的设计	297
13.3.3	单一继承	300
13.3.4	派生类	300
13.3.5	受保护成员	300
13.3.6	派生成员和定制成员	301
13.3.7	公有基类和私有基类	301
13.3.8	基类和派生类的构造函数	301
13.3.9	定制新的成员函数	302
13.3.10	通过重定义基类成员函数进行定制	302
13.3.11	编写程序	303
13.3.12	用于基类和派生类的作用域运算符	304
13.3.13	多个派生类	305
13.3.14	第二个派生类	305
13.3.15	使用基类和两个派生类	307
13.3.16	派生类之间的关系	308
13.3.17	从派生类派生出来的类	308
13.4	问题域类层次结构的设计	311
13.4.1	C++ 与纯面向对象的设计	312
13.4.2	更多的数据抽象	313
13.4.3	重载<<和>>运算符	314
13.4.4	SSN 和 Money 类	316
13.4.5	包含头文件	318
13.4.6	C++ 的标准字符串类	319

13.5	Person: 基类	319
13.5.1	枚举数据类型	322
13.5.2	虚函数	322
13.5.3	抽象基类: 纯虚函数	322
13.5.4	两种数据显示方式	322
13.6	派生类	322
13.6.1	Employee 类	322
13.6.2	函数重定义	324
13.6.3	WagedEmployee 类	324
13.6.4	SalariedEmployee 类	326
13.6.5	Contractor 类	327
13.6.6	构造目标代码库	329
13.6.7	问题域类层次结构	329
13.6.8	通过引用调用虚函数	332
13.6.9	通过引用调用非虚函数	333
13.6.10	调用重定义的虚函数	333
13.6.11	没有被派生类重定义的虚函数	333
13.6.12	虚析构造函数	333
13.6.13	应该将哪些成员函数定义为虚函数	336
13.7	多态性	336
13.8	小结	337
第 14 章	多重继承	338
14.1	多个基类	338
14.1.1	多重继承中的构造函数的执行	339
14.1.2	多重继承中的析构函数的执行	339
14.2	细化资产系统设计	339
14.3	对多重继承中的成员进行重定义	341
14.4	多重继承中的二义性问题	341
14.4.1	具有二义性的成员函数	342
14.4.2	具有二义性的数据成员	342
14.4.3	在设计时解决二义性问题	342
14.4.4	不可避免的二义性问题	343
14.5	虚基类	344
14.6	实现所设计的类层次结构	345
14.6.1	DisplayObject 类	345
14.6.2	Vehicle 类	345
14.6.3	Property 类	346
14.6.4	Asset 类和 Expense 类	347
14.6.5	Computer 类和 Pencils 类	349
14.6.6	CompanyCar 类和 LeaseCar 类	349
14.6.7	应用程序	349
14.7	回顾	350
14.8	小结	351
第 15 章	类模板	352
15.1	基础知识	352
15.2	声明类模板的多个对象	353