

计算机 C/C++ 语言系列丛书



PROGRAMMING WITH MICROSOFT VISUAL C++ 1.5

SCOTT PALMER



希望

学苑出版社

Programming With Microsoft Visual C++ 1.5

Visual C++ 1.5 程序设计

SCOTT PALMER 著

黄光 程思乾 译

李津 万博 审校

学苑出版社

(京)新登字 151 号

内 容 简 介

本书是针对利用 Visual C++ 进行 C 和 C++ 程序设计的人编写的,全面系统地介绍了 Visual C++。首先介绍 Visual C++ 集成环境(Visual Workbench),说明集成编辑器和调试器的用法;然后介绍 C 和 C++ 语言的基础知识,包括操作符和表达式、说明、控制语句、函数、指针、其他高级数据结构和文件输入/输出;接着介绍了类型兼容、宏、条件编译指令、预编译处理,讨论了面向对象的基本编程方法。

欲购本书的用户,可与北京海淀 8721 信箱书刊部联系,电话 2562329,邮编 100080。

版 权 声 明

Copyright © 1994 by SCOTT PALMER.

This translation published by arrangement with Random House, Inc.

本书英文版名为《Programming With Microsoft Visual C++ 1.5》,由 Random House 公司出版,本书中文版由 Random House 授权出版。未经出版者书面许可,本书的任何部分不得以任何形式或任何手段复制或传播。

计算机 C/C++ 语言系列丛书

Visual C++ 1.5 程序设计

著 者: SCOTT PALMER

译 者: 黄 光 程思乾

审 校: 李 津 万 博

责任编辑: 甄国宪

排 版: 万博图书创作社

出版发行: 学苑出版社 邮政编码: 100036

社 址: 北京市海淀区万寿路西街 11 号

印 刷: 施园印刷厂

开 本: 787×1092 1/16

印 张: 21.875 字数: 495 千字

印 数: 1~5000 册

版 次: 1994 年 9 月北京第 1 版第 1 次

ISBN7-5077-0875-6/TP·24

本册定价: 44 元

学苑版图书印、装错误可随时退换

前 言

您买这本书,也许是由于本书论述用最新的 Visual C++ 1.5 进行程序设计。您是对的。但只猜对一半:本书还是一个冒险家的乐园,充满了惊险和乐趣。我们大多数程序员不是为了金钱而编程,而是为了乐趣。本书介绍如何用 C++ 进行程序设计,如何编写 Windows 程序,如何使用 Visual C++ 的工具,如何发现乐趣。

C++ 和 Microsoft Windows 就像是一个程序的游乐场:充满新的思想、功能强大的工具和技巧。有了这本书,将学习到面向对象程序设计的最新概念和技术,如如何创建数据类型,懂得如何管理类和数据,如何从预定义数据类型中继承函数和数据,如何使用 C++ 流 I/O 性能,以及如何管理更复杂的工程。最后,还介绍如何使用 Windows Programming Interface (API),既可通过直接使用,也可通过功能强大的 Microsoft Foundation Class 库 (MFC) 来提高开发效率。

本书的内容

本书主要讨论三个主要的中心:首先,介绍 C++ 和面向对象程序设计的基本技术;其次介绍 Microsoft Windows 程序设计的基础思想和技术;最后,讨论如何 Microsoft Visual C++ 的主要功能。由于 C++ 是 C 程序设计语言的超集,有必要让读者了解 C 从而更有效地理解 C++。因此,C 并没有深入讨论,只在第一章中介绍了 C 和 C++ 之间的差别。另外,本书中说明的 Microsoft Windows 程序设计技术适用于 C、C++ 和其他能编写 Windows 程序的程序设计语言。

本书的目标

本书适合任何一个有一些编程经验的人想学习 C++ 和面向对象程序设计,特别是学习用标准和专业版的 Microsoft Visual C++ 编写 Microsoft Windows 3.X 程序的人。也可以被用其他能编写 Windows 程序的程序设计语言人作为一般 C++ 和 Windows 程序设计的指导性资料。

本书的组织

本书分成三个部分,第一部分由第一章和第二章组成,介绍 C++ 的概况。第二部分由

第三至十五章组成,循序渐进说明了习面向对象程序设计和 C++ 程序设计语言主要特性。这部分的内容适用于其他目标环境下任何标准 C++ 编译器的程序设计,目标环境既可以是基于文本的,也可以是基于图形界面的 PC、Macintosh 和主机。

第三部分由第十六至第十八章组成,阐述用 Visual C++ 循序渐进进行 Windows 程序设计技术。内容包括 Windows 程序设计的基本特点、Windows API 和 Visual C++ 中的 Microsoft Foundation Class 库。

必备条件

由于本书涉及一般的 C++ 程序设计以及 Windows 程序设计,可以作为任何符合 C++ 标准的语言的参考书,包括 Visual C++ 1.0 和 1.5, Microsoft C/C++ 7, Borland C++, Watcom C++ 和 Zortech C++。由于 C 程序设计没有深入讨论,因此,需要一个 C++ 编译器,但一些概念和技术用 Quick C for Windows 和其他 C 工具进行 Windows 程序设计相同。

为了更好地使用本书,需要至少有一个 80386 以上的 PC, 安装有 Visual C++、Microsoft Windows 3.1 和 MS-DOS 5.0 或更高版本。其中的许多程序设计技术和示例还可适用于 Microsoft Windows NT、Windows for Workgroups 和老的 Windows 3.0。

除此之外,还需要一个脑袋,一点空余时间和冒险精神。

致 谢

虽然我的名字出现在封面上,但这本书是集体努力的结果。对 Random House 的编辑 Kim Fryer 表示诚挚的感谢,没有他的信赖、鼓励和少有的耐心,这本书不可能与读者见面。Random House 的主编 Mia McCroskey 也给予了很大的帮助。

Electric Ink 的 Karl Barndt 和 Lydia Levins 以极度认真的态度处理了文书编辑和生产。感谢我的家人。

最后要感谢的人是帮助我出版这本书的您,读者。如果本书对您有所帮助,是对我的最大慰藉。

目 录

第一章 序 论	1
1.1 C++历史的简单回顾	1
1.2 过程式程序设计	2
1.2.1 结构化程序设计	2
1.2.2 被动式的数据	2
1.3 面向对象的程序设计(OOP)	3
1.3.1 封装:把对象当作智能的数据.....	3
1.3.2 智能数据的自我保护	4
1.3.3 智能数据对编程的简化	5
1.3.4 OOP 对结构化程序设计思想的扩展	5
1.3.5 继承和类层次	6
1.3.6 多态和对象	7
1.4 从C到C++	7
1.4.1 C++对C的继承	7
1.4.2 I/O特性的改进	7
1.4.3 源代码可读性的提高	9
1.4.4 函数的重载.....	10
1.4.5 指针和引用.....	10
复习	11
练习	12
第二章 Visual C++的开发环境	13
2.1 安装 Visual C++	13
2.2 使用 Visual 工作台	14
2.3 创建程序.....	14
2.4 选择工程文件类型.....	16
2.5 程序的编译和连接.....	17
2.6 程序的运行.....	18
2.7 使用工程文件.....	18
2.7.1 创建工程文件.....	18
2.7.2 使用 Visual C++ App Wizard	20
2.8 使用 Visual 工作台调试工具	21
2.8.1 单步调试程序.....	21
2.8.2 监视变量.....	22
2.8.3 设置和删除断点.....	22

2.9	使用编辑快捷.....	23
2.9.1	设置、删除标志及跳转至标志	25
2.10	编辑器图示行的使用	25
2.11	编辑器和字型的选择	26
2.12	使用 Help 系统	26
2.13	APP Studio	26
2.14	基类	27
	小结	27
	测验	28
第三章	C++编程基础	29
3.1	C++程序剖析	29
3.1.1	Hello 程序的运行	30
3.1.2	main()函数.....	30
3.1.3	语句.....	32
3.1.4	注释句.....	33
3.1.5	预处理器指令.....	34
3.1.6	源代码格式化.....	35
3.2	变量和常数.....	35
3.2.1	程序中怎样(为什么)定义变量.....	36
3.2.2	命名变量和其他程序元素.....	38
3.2.3	理解 C++数据类型.....	40
3.3	语句.....	41
3.3.1	I/O 语句	41
3.3.2	Assignment 语句(称为分派语句或赋值语句)	41
3.3.3	比较语句.....	41
	复习	41
	练习	42
第四章	简单数据类型	43
4.1	C++的预先定义数据类型	43
4.1.1	int 型	44
4.1.2	短型和长型.....	47
4.1.3	unsigned 性质	47
	练习	48
4.1.4	Char 型	49
4.1.5	float 型	53
4.1.6	Void 型	53
4.1.7	"True(真)" "False(假)" 型	53
4.1.8	用 typedef 来产生数据型的新名字	54
4.2	改变变量的数据类型.....	54

4.3 再谈定义常数.....	56
4.3.1 为什么要操心常数定义呢?	56
4.3.2 用 const 和 #define 生成常数	56
练习	57
第五章 简单 C++ 语句的使用	58
5.1 基本语句型.....	58
5.1.1 数据移动语句.....	58
5.1.2 程序移动语句.....	58
5.1.3 返回语句.....	58
5.2 赋值语句.....	60
5.2.1 C++ 不检查赋值错误	61
5.2.2 在赋值语句中使用其他操作符.....	63
5.2.3 给字符串变量赋值.....	64
5.3 I/O 语句	67
5.3.1 用 cout 输出	68
5.3.2 用换码序列格式化输出.....	69
5.3.3 用控制 I/O 格式化输出	71
5.3.4 使用 cin 的标准输入	73
5.3.5 用 cin 成员函数检查输入	73
5.3.6 控制输入空格.....	75
复习	76
练习	76
第六章 操作符与表达式	78
6.1 单目操作符与双目操作符.....	79
6.2 算术符.....	79
6.2.1 熟悉的操作符 +, -, *, /.....	79
6.2.2 表达式中的混合数据类型.....	86
6.2.3 增量符.....	86
6.2.4 前置、后置和中间注释	88
6.2.5 减量符.....	89
6.3 逻辑符.....	90
6.3.1 理解真伪表.....	91
6.3.2 ! 符.....	91
6.3.3 && 符	91
6.3.4 符.....	92
6.4 关系运算符.....	94
6.4.1 == (相等) 运算符.....	94
6.4.2 != (不等) 运算符.....	94
6.4.3 >, <, >= 和 <= 运算符.....	95

6.5	赋值符仅仅是运算符.....	95
6.5.1	程序 6.8 分析:显示赋值表达式的值	96
	复习	98
	练习	98
第七章	程序的 for、do 和 while 循环	101
7.1	for 循环.....	101
7.1.1	for 语句的组成部分.....	101
7.1.2	当程序遇到 for 循环时会发生什么	102
7.1.3	语句中的一些部分可省略	104
7.1.4	在循环检测表达式中使用变量或常数	104
7.1.5	向下计数而不是向上计数	105
7.1.6	循环的嵌套	106
7.1.7	使用多循环变量	107
7.1.8	何时使用或不使用 for 循环	108
7.1.9	危险! 无穷 for 循环	109
7.2	从循环中跳出	109
7.2.1	用 Continue 跳过几步	110
7.2.2	用 break 中断循环	110
7.3	while 循环.....	111
7.3.1	使用基本的 while 循环	114
7.3.2	while 循环中的多重条件.....	117
7.3.3	何时使用 while 循环	118
7.4	do 循环	118
7.4.1	使用一个简单的 do 循环.....	119
7.4.2	危险! 无穷 do 循环.....	120
7.4.3	何时使用 do 循环.....	121
	复习.....	121
	测验.....	121
	练习.....	121
第八章	用 if else case 进行程序判定	123
8.1	if 语句	123
8.1.1	用 if 语句来做一个简单的事...if	124
8.1.2	if 语句中的多重条件	125
8.2	if...else 语句	126
8.2.1	用嵌套 if...else 语句处理更多的选项	128
8.2.2	工作顺序	129
8.2.3	条件运算符	131
8.3	Switch 语句	133
8.3.1	比 if...else 更有效地处理多重选择	134

8.3.2	break 的重要性	136
8.3.3	default 情况	136
8.3.4	处理具有同样结果的多种情况	137
8.4	if、switch 语句比较	139
8.5	其他控制语句及函数	139
8.5.1	用 goto 跳行(并非一个好方法)	139
8.5.2	使用 exit() 来停止程序	140
	复习	141
	测验	142
	练习	142
第九章	复杂数据类型和用户自定义数据类型	143
9.1	使用数组	143
9.1.1	给数组赋值	145
9.1.2	在数组中使用数据	149
9.1.3	当数值是未知时	152
9.1.4	使用哨卡	152
9.1.5	将未知数值放入数组	152
9.1.6	危险! C++ 不检查数组边界	156
9.1.7	多维数组	158
9.2	使用结构	158
9.2.1	定义类型, 然后定义变量	160
9.2.2	结构变量的运用	161
9.2.3	将数据放入结构	162
9.2.4	从结构中取出数据	162
9.3	结构数组	163
	复习	166
	测验	166
	练习	166
第十章	指针和动态分配	168
10.1	什么叫指针?	168
10.1.1	指针使程序动态地建立新变量	168
10.1.2	传递变量给函数	169
10.1.3	地址操作符	169
10.1.4	声明指针变量	171
10.1.5	在赋值语句中使用指针	172
10.1.6	在同一行中声明多个变量	174
10.1.7	指针的数据类型	175
10.1.8	指针运算	177
10.2	用 new 和 delete 建立动态变量	181

10.2.1 使用操作符 new	182
10.2.2 使用 delete 操作符	183
10.2.3 使用指针建立可变长度的数组	185
复习	188
测验	189
练习	189
第十一章 在程序设计中 使用函数	190
11.1 引言	190
11.2 函数是什么?	190
11.3 函数与结构化程序	190
11.3.1 面向对象扩充了结构化程序设计	191
11.3.2 C++ 函数的声明	194
11.4 同时接收和返回值的函数	200
11.4.1 接收并返回值的实例: 程序 11.3	202
11.4.2 为什么使用参数?	203
11.5 接收数据的函数	207
11.5.1 给函数传递数据: 在程序 11.5 中	209
11.5.2 函数缺省时传递数据	209
11.6 使函数返回多个值	209
11.6.1 用指针获得变量本身的函数: 在程序 11.6 中	211
11.6.2 传递数组: 传递指针的一种特殊情况	212
11.6.3 传递结构给函数	213
11.6.4 通过使用别名来接收变量的函数	215
11.7 内部函数	218
11.7.1 当可以不使用指针时	221
11.7.2 作用域和存贮类型的思想	221
11.7.3 局部(自动)变量	221
11.7.4 全局(外部)变量	223
11.7.5 静态变量	224
复习	225
测验	226
练习	226
第十二章 函数重载与缺省参数	227
12.1 什么叫函数重载?	227
12.1.1 没有函数重载的情况	227
12.1.2 重载带有不同类型参数的函数	228
12.1.3 参数数量不同的函数重载	230
12.2 使用缺省参数	232
12.2.1 使用缺省参数的规则	232

复习.....	233
练习.....	233
第十三章 类、对象和继承	234
13.1 类和面向对象的程序设计.....	234
13.1.1 OOP 与结构化程序设计的比较:一个实例	235
13.1.2 OOP 和非 OOP 程序的关键区别	238
13.2 声明类.....	239
13.2.1 定义类的成员函数.....	242
13.2.2 调用类的成员函数.....	243
13.2.3 类的公有成员和私有成员的比较.....	244
13.2.4 对象数据的有效性检验:在清单 13.3 中.....	248
13.2.5 用户应定义哪个类?	249
13.2.6 每个类应有什么样的成员函数?	249
13.3 构造函数和析构函数.....	250
13.4 操作对象.....	254
13.4.1 将一个对象的内容赋给另一个对象.....	254
13.4.2 将对象用作函数参数.....	255
13.4.3 将对象用作函数返回值.....	255
13.4.4 对象的动态分配.....	255
13.5 使用类继承.....	257
13.5.1 使用类继承的好处.....	257
13.5.2 结构化程序与面向对象程序的比较:一个例子	257
13.5.3 继承的主要特征.....	262
13.5.4 基类的数据成员从 private 改为 protected	262
13.5.5 声明派生类(或子类).....	263
13.5.6 派生类具有基类的一切.....	263
13.5.7 覆盖基类(祖先)函数.....	264
13.5.8 如何覆盖继承来的函数.....	268
13.5.9 在需要时调用被覆盖的函数.....	269
13.5.10 将常用代码写进全局函数	269
复习.....	270
测验.....	270
练习.....	270
第十四章 缺省参数、友函数和操作符重载	272
14.1 缺省参数.....	272
14.2 友函数.....	275
14.2.1 使用友函数:在程序 14.3 中.....	277
14.3 操作符重载.....	277
复习.....	280

测验.....	280
练习.....	280
第十五章 虚函数和多态性.....	282
15.1 为什么要用虚函数?	282
15.1.1 虚函数用在哪里:一个例子	283
15.1.2 不用虚函数改变程序.....	287
15.2 声明虚函数.....	292
复习.....	296
测验.....	296
练习.....	297
第十六章 使用文件和流.....	298
16.1 文件类型.....	298
16.1.1 头文件 fstream.h	299
16.2 处理文本文件.....	299
16.2.1 打开文件用于写.....	299
16.2.2 关闭文件.....	300
16.2.3 打开文件用于读.....	301
16.2.4 在打开文件时检查错误.....	302
16.2.5 提示用户输入文件名.....	303
16.2.6 打开文件的另一种方法.....	304
16.3 处理二进制文件.....	305
16.3.1 写二进制文件.....	305
16.3.2 读二进制文件.....	306
16.3.3 用 seekg()函数寻找二进制文件数据	307
16.3.4 对二进制文件写和读结构变量.....	308
16.3.5 对二进制文件写和读对象变量	308
复习.....	311
测验.....	312
练习.....	312
第十七章 Windows 程序设计的基本概念	313
17.1 一个简单的 Windows 程序	313
17.1.1 建立 Visual C++工程文件	316
17.1.2 建立和运行程序.....	319
17.2 Windows 程序的基本结构	320
17.2.1 头文件 Windows.h	320
17.2.2 窗口过程原型 WndProc()	321
17.2.3 WinMain()函数	321
17.2.4 窗口过程 WndProc()	323
17.2.5 用 Paint()函数显示文本	324

17.2.6	用 PostQuitMessage()退出	324
17.2.7	模块定义文件	325
17.2.8	资源文件	325
	复习	325
	测验	326
	练习	326
第十八章	Visual C++ AppWizard 简介	327
18.1	启动 AppWizard	327
18.1.1	给应用程序增加特征	329
18.1.2	建立并运行应用程序	331
18.2	使用 App Studio	331
18.3	对进一步学习的指导	334

第一章 序 论

在这一章中,我们先准备对 C++ 作一个综述,即讨论它的演化过程、对流水线技术的采纳以及对新的面向对象编程规范的支持。之后我们比较一下 C++ 和 C 语言的异同点,看一看怎样才能用 Visual C++ 来达到最好的编程效果。

作者先提醒一点,本章的目的只是给出一个概念性的综述。虽然文中不时地会出现一些短的 C 和 C++ 程序代码,详细讨论编程的内容不属本章范畴,而是留到后续各章中。如果读者不喜欢“概念性”的讨论(正如绝大多数的程序员一样),那就可能总想着跳过本章内容而直接进入第二章,在那里将介绍 Visual C++ 引入的一些专用工具。不过,作者还是请各位至少把本章的内容大致浏览一下,因为书中的内容安排是前后连贯的,了解本章就有助于更好地理解后面的内容。有些例子中的代码是完整的程序,读者如果有兴趣还可以试着进行调试运行。这些程序有意设计得比较简单,以便把要说明的观点说清楚。

1.1 C++ 历史的简单回顾

首先请读者注意:虽然 C++ 是基于 C 语言的一门编程语言,但是学 C++ 之前不必一定要先懂 C 语言。C 语言确实包含了 C++ 的许多特点,但其中的编程技巧有大量是在 C++ 中不需要的,故学过 C 的读者又得放弃这些技巧。尽管如此,读者们还是能体会到 C++ 编程技巧比 C 更为简单明了。正如 C++ 的创立者 Bjarne Stroustrup 所回忆的:“当初提出 C++ 主要是为了使(我自己和)我的朋友们可以不必采用汇编、C 和许多现代的高级语言编程,它的主要的目的是使程序员更容易,而且心情更舒畅地编写出好程序来。”

C++ 对 C 作了大量的改进,其中的一些内容将在本章讨论到。但 C++ 最重要的创新之处既是语法性的,又是概念性的:它定义了一种新的称为“类”的数据类型,从而对“面向对象的程序设计”提供了显式的支持。事实上,当 C++ 在 1980 年刚提出时,它叫做“带有类的 C”。“C++”这个名字是后来形成的,“++”是指 C 语言的递增操作符(++),以表明 C++ 与 C 的演变继承关系。

似乎在地球上住着的人都应知道,面向对象程序设计是近几年在程序设计领域最热门的话题。但它不仅仅是一种时尚,不会像信息资源管理和时下许多其他唬人的新名词这样的时尚很快就会为世人所遗忘。它也不是只通过学一下 C++ 的“class”这个关键字的语法就能掌握的东西。它需要的是头脑中的概念革命,即在思考和设计程序的方法上来一次彻底的变革。

1.2 过程式设计

传统上用 BASIC、Pascal 和 C 等过程语言进行编程时,“处理过程”和“数据”这两者是有严格界限的。程序从用户那里接受一些数据,然后对这些数据执行一系列指令,如“把 X 加到 Y 中,把和拷到另一个地址,再乘以 0.05,最后打印结果”。最早的程序其复杂性莫过于此。

1.2.1 结构化程序设计

随着程序变得越来越庞大和复杂,这种简单的逐条排列程序指令的顺序方法就显得太庞杂和笨拙。对于一个 50 行的程序,要理解其中的思路并不太难,但要是碰到一个 50000 行的程序,我们就需要找出一种方法来把程序分解为几个功能块,以使之更容易理解。

结构化程序设计具有这一优点,它把程序分解为一些子例程,这些例程在 C 和 C++ 中称为函数,在 Pascal 中称为函数或过程。这样分解之后,数据就可以传绘各个子例程,子例程当然要比整个程序小得多也简单得多。处理数据的每一步都可以观察清楚,而子例程的思路也容易理解了,因为它的长度和难易程度能为我们所驾驭。

1.2.2 被动式的数据

结构化程序设计尽管十分重要且有用,仍然只是一个进步,而不是一次概念上的革命。和非结构化程序设计一样,在结构化程序设计中,“处理”就是做某件事,“数据”就是处理的对象。程序中的数据是一个被动的旁观者,就像躺在手术车上的病人一样。病人被推着从一个科室换到另一个科室,又是拍 X 光片,又是听诊,又是打针——我们可以推想他在整个过程中的被动作用多少会使他心里感到有些难受——数据也是如此,它只是等在那儿看着程序对它进行各种处理。即使它想发挥更积极的作用,它也不知道该怎么做:它毕竟只是一堆数而已。

被动式的数据还有另外一层意思:它没有抵抗错误处理的内在保护功能。如果某个子例程经修改后对数据进行了不正确的处理,那么数据就会遭到破坏。正如一个病人吃错药之后疾病会加剧——医生称之为“庸医病”——一样,被动接受各种处理的数据其安全性也低于应有的标准。

在结构化程序设计中,解决这一问题有一个方法,就是使数据只属于用到它的那些子例程,如图 1.1 所示。这样,任何其他的子例程就无法访问这些数据了。但是,这种方法经常会做过头:有些数据项需要在程序中多处被引用。例如,一个职员资源管理系统可能需要包括多个子例程,用于分别计算员工薪水、统计病假天数、记录值得表彰的事绩等等。如果员工的纪录只能被某个子例程所访问,那就排除了需要这些数据的所有其他子例程访问这些数据的可能性;但如果这些数据能为所有子例程所用(即成为全局数据),则可能将从意想不到的地方招致对数据的破坏。