刘振安　苏仕华　赵晓东　编著

程序设计

中国科学技术大学出版社

C++程序设计

刘振安　苏仕华　赵晓东　编著

中国科学技术大学出版社
1997·合肥

内 容 简 介

C++语言是为适应90年代开发和维护复杂的应用软件的需要而研制的。本书主要以没有学习过C语言，而又准备直接学习C++语言的读者为对象，重点是强调面向对象的程序设计方法及其学习方法。概念清楚，重点突出，容易理解。

全书分九章。第一章介绍C++基础知识；第二章介绍对象和类；第三章介绍对象的初始化；第四章介绍继承和派生类；第五章介绍多态性和虚函数；第六章介绍使用成员函数；第七章介绍运算符重载及流类库；第八章介绍模板。

为了掌握C++语言的基本概念及设计方法，我们还单独开辟了第九章，以生动的课程设计与实践为主，结合实例对面向对象设计方法进行综合归纳，并给出几个实用的例子以增强理解。以训练读者的独立解决问题的能力。同时针对全书各章的内容，设计了形式多样的习题。

本书既可作为大专院校及社会上各种计算机培训班的高级教学参考书，也可作为广大计算机爱好者的自学教材。

图书在版编目(CIP)数据

C++程序设计/ 刘振安　编著. —合肥：中国科学技术大学出版社，1997年8月
ISBN 7-312-00911-5

Ⅰ　C++程序设计
Ⅱ　①刘振安　②苏仕华　③赵晓东
Ⅲ　①C++程序设计　②计算机语言　③教学参考书
Ⅳ　TP

凡购买中国科大版图书，如有白页、缺页、倒页者，由本社出版部负责调换

中国科学技术大学出版社出版发行
（安徽省合肥市金寨路96号，邮编：230026）
中国科学技术大学印刷厂印刷
全国新华书店经销

开本：787×1092/16　　印张：18.5　字数：461千
1997年8月第1版　　1999年6月第3次印刷
印数：5001—8000册　　定价：19.00元

ISBN 7-312-00911-5/TP·184

前　言

C++语言是C语言的扩充,C语言最初用作UNIX操作系统的记述语言,由于UNIX的成功和广泛使用,也使C语言成为一种普通使用的程序设计语言,目前在各种机型和各种操作系统上都有C编译器.C语言是为了能够胜任系统程序设计的要求而开发的,因此有很强的表达能力,能够用于描述系统软件各方面的特性,用C语言编写的程序生成的机器代码质量也很高。例如UNIX系统只使用了很少的汇编代码,其余代码全部是用C语言写成的,这是C语言非常成功的典例。

C语言具有较高的可移植性,并提供了种类丰富的运算符和数据类型,所以极大地方便了程序设计,但同时也使得C语言较难被初学者学习和掌握。除此之外,C语言本身也存在着局限性。例如C语言的类型检查机制相对较弱,这使得程序中的一些错误不能被编译器检查出来,而这些错误若是遗留到程序的运行阶段由程序员检查,又将是很困难的。C语言本身几乎没有支持代码重用的语言结构,因此一个程序员即使有很高的程序设计技巧,并严格遵循模块化程序设计方法,为一个应用程序编写的代码也很难重用于另一个程序之中。而且C语言也不适合于开发大型程序,当程序的规模达到一定程度时,程序员就很难控制程序的复杂性。为了解决上述问题并同时保持C语言的优点,就把C语言扩充为C++语言。

C++语言是为适应90年代开发和维护复杂的应用软件的需要而研制的。它的目标是为程序员的程序开发活动提供优良的程序设计环境,以便能产生模块化程序高、重用性和可维护性均好的程序。同时,C++语言非常强调代码的有效性和紧凑性,它是程序员的语言,允许程序员决定如何实现特定的操作。因此,目前C++语言已经在各个领域得到了广泛应用,尤其适用于中等和大型的程序开发项目。已经证明,C++语言应用于C语言曾经使用过所有场合,其效果比C语言要好得多,从开发时间、开发费用到形成的软件的可重用性、可扩充性、可维护性和可靠性等方面,都显示出C++语言的优越性。

在程序设计方法方面,C++语言既支持传统的面向过程的程序设计方法,也支持新的面向对象的程序设计方法。因此,它是一种混合语言。由于C++语言的这种特性,就使得C++语言保持与C语言兼容,从而使许多C语言代码不经修改就可以为C++语言所用,用C语言编写的、众多的库函数和实用软件,也可用于C++语言中,方便了C语言用户向C++语言过渡。不过,用C++语言编写的程序的可读性更好,代码结构更为合理,可以直接地在程序中映射问题空间的结构。

确实,C程序员仅需学习C++语言的特征,就可以很快地用C++语言编写程序。但更重要的是:不要求程序员先学习C语言,然后学习C++语言。反之,没有学过C语言的程序员,学习C++语言更容易。这是因为C语言是面向过程,而C++语言是面向对象。虽然编译器是一个,但我们不是学习使用编译器,而是学习C++语言面向对象的编程方法。已经对C语言熟悉的程序员,很容易穿着C++语言的鞋子,却走着C语言的路子。没有学过C语言,反而不受C语言框框的影响。因此,学习C++语言的关键是面向对象的思维方法。

本书是假定读者没有学习过C语言,而直接学习C++语言的教材,且因受篇幅和学时的

限制，故本书的重点是强调面向对象的程序设计方法。为了掌握这些基本概念及设计方法，我们还单独开辟一章“课程设计与实践”，结合实例对面向对象设计方法进行综合归纳，并给出几个实用的例子以增强理解。

全书分九章。第一章是C++基础知识。包括C++语言的基本语法、数据类型、函数及指针等程序设计基础知识。如果读者已经学习过C语言，这一章的很多内容也需要学习。因为这一章是从C++语言的观点出发的，它包含许多C语言所没有的概念。第二章是对象和类。重点是介绍面向对象的程序设计知识及定义和使用类的方法。第三章是对象的初始化。介绍构造函数和析构函数的知识。第四章是继承和派生类。介绍单一继承、多重继承和虚基类。第五章是多态性和虚函数。介绍C++语言的多态性、虚函数、虚函数的多态性及虚析构函数。第六章是使用成员函数。介绍静态成员、友元、const对象、volatile对象、转换函数、指向类成员的指针、数组及联合等知识。第七章是运算符重载及流类库。介绍类运算符、友元运算符、重载、流类库及流的错误处理。第八章是模板。模板是将来的发展趋势，所以本书也介绍模板的基本概念。本章简要介绍函数模板、类模板、模板与继承的关系。第九章是课程设计与实践。本章先进一步综合介绍面向对象的设计方法，然后在课程设计中讨论对类的整体认识、类的组织、类和函数的设计与说明及继承和面向对象设计等问题。本章还包括课程设计实践，给出大学人员管理程序、无差异树的实现及通讯录管理程序设计等三个实际的例子。

本书的程序均经过验证，需要本书程序的读者，可以直接与我们联系。

本书旨在给学生一个独立解决问题能力的训练，所以概念清楚，重点突出，容易理解，并专门开辟一章课程设计与实践，系统训练思考问题和解决实际问题的能力，使学生对使用C++语言编程有一个完整的整体认识，并初步掌握实用程序的编制方法及大程序的设计方法。

各章均附有精心挑选的习题，为了方便学习，我们将准备编写单独的例题解答及本书的习题解答，并分别给出使用Borland C++和Visual C++的上机指导、实验题及其解答。

本书既可作为大专院校及社会上各种计算机培训班的高级教材，也可以供广大计算机爱好者自学时参考使用。

谢小娟、金辉宇和颜廷荣为本书的内容和程序花费了大量的时间和精力进行仔细地修改及验证。本书还参考了大量书籍及文献资料，在此向被引用资料的作者及给予帮助的所有同志再次表示感谢。

参加本书编写工作的还有谢小娟、金辉宇、颜廷荣、肖艳、黄南晨、周淞梅、李笑昕、刘丽丽和卢梅等。

由于我们水平有限，不妥之处再所难免，希望同行及读者指正。

刘 振 安

1997年7月1日

中国科学技术大学

目　　次

第一章 C++基础知识

我们在讲述对象和其它面向对象的构造之前，先重点讨论 ANSI C 扩展或派生出的 C++语言的主要特征，以便为学习 C++的类打下基础。因为重点是学习 C++程序设计方法，所以避免对细小问题的冗长讨论。读者无需学习 C 语言即可直接学习 C++，这还能避免 C 语言带来的副作用。本章虽是为没学过 C 语言的读者提供的 C++基础知识，但却涉及 C++的特有特征，而这些特征又与 C 语言不同，所以要求已经熟悉 C 语言的读者也应该先学习本章的内容。

1.1 C++ 概 述

70 年代以来，C 语言逐渐风靡世界，成为使用最广泛的编程语言之一。但随着问题复杂度的提高和面向对象方法的提出，C 语言显得力不从心，而 C++也就应运而生。C++保持与 C 兼容，即 C 代码及用 C 编写的库函数和实用软件也可用于 C++中；另外，用 C++编写的程序的可读性更好，代码结构更为合理，可直接在程序中映射问题空间的结构；更重要的是，C 程序员仅需学习 C++语言的特征，就会用 C++编写程序。

C++已经被应用于程序设计的众多应用领域中，它尤其适用于中等和大型的程序开发项目。从开发时间、费用到形成的软件的可重用性、可扩充性、可维护性和可靠性等方面都显示出 C++的优越性。

1.2 C++程序及项目结构

C++约定源文件使用扩展名.cpp，头文件使用扩展名.h 或.hpp。当源文件使用.c 的扩展名时，编译器就按 C 的语法编译这个程序。虽然项目结构的具体细节决定所使用的 C++编译器，但都与 ANSI C 很类似。例如使用 Borland C++，在头文件中建立结构(或类)以及常量的声明。因为 C++头文件中可能包括 ANSI C 不能编译的构造，所以用 *.hpp 来标识，以区别于通常的 ANSI C 头文件 *.h。

下面的简单程序用来说明注释、include 文件、main( )函数、变量及语句等语法现象。

【例 1.1】 说明 C++构造的示例程序。

```
/* 输入两个整数，输出其和的 C++示例程序 */
   #include <iostream.h>          // 系统头文件

   void main( )
   {
      int a, b;
```

```
    cout << "Enter two integer:";
    cin >> a >> b;
    int result;
    result = a + b;
    cout << "\nThe sum of " << a << "+" << b
        << " = " << result << endl;
}
```

1. 注释

C++编译器跳过程序中的注释，不处理注释的内容。使用注释来描述程序的功能，完全是为了增强程序的可读性并提供理解程序的线索。符号“//”告诉编译程序，本行“//”之后的所有内容都是注释；而符号“/*”和“*/”告诉编译器在“/*”和“*/”之间的内容都是注释。注释符号“//”适合短的只占一行的注释，而“/* ... */”适合于长的占用多行的注释。

程序文件的书写格式是自由的，空格起分隔单词的作用(多个空格符仅被当作一个空格符看待)。在上面的程序中，通过恰当地使用空格和空行，也可增强程序的可读性。

2. 包含文件及头文件

C++语言包含文件的格式有两种。以 Borland C++为例，第一种为：

```
#include <文件名.扩展名>
```

编译器并不是在当前目录查找，而是根据目录出现的顺序分别查找各路径。尖括号中的空格被认为是文件名的一部分。这种包含方法常用于标准头文件。例如 stdio.h、string.h 等，均是 Borland C++的组成部分。第二种为：

```
#include "文件名.扩展名"
```

这使得编译器首先在当前目录中查找，然后像上一种方式那样在标准目录中查找。因为在引号中的空格被认为是文件名的一部分，所以应避免可能引起错误的空格。

iostream.h 是 C++的系统文件，经常称其为头文件。而

```
#include <iostream.h>
```

行的作用是指示 C++编译器将文件 iostream.h 的内容插入到程序中#include 指令所在的这一行的后面，这使得程序可以使用在文件 iostream.h 中定义的标准输入和标准输出操作。只有这样，语句 cout 和 cin 才能被正确编译。

类标识符在使用之前，必须先进入作用域，这通常是用类声明来实现的。使用类的公有成员时，类标识符不仅要在活动作用域中，而且必须被完全声明。因为每个类都有自己的声明部分，而且各种模块都可以使用它，所以常把声明置于头文件中，用到该类的每个模块再包含相应的头文件。

假设一个典型的程序是由大量类和头文件组成的，常常会遇到在同一个文件中，一个头文件可能被包含多次。例如：

```
// file HEADER.HPP
class EssentialClass { /* ... */ };
// file DESKTOP.HPP
```

```
#include "HEADER.HPP"
class DeskTop { /* ... */ };

// file DRAWER.CPP
#include "HEADER.HPP"
#include "DESKTOP.HPP"      // file HEADER.HPP included
                            // twice !
class Drawer { /* ... */ };
```

为避免这种错误，C++头文件用一些预处理指令产生下面的简单而有效的分离构造：

```
// file HEADER.HPP
#ifndef HEADER.HPP
#define HEADER.HPP
class EssentialClass { /* ... */ }
...
#endif
```

所有的头文件都采取相同的机制。通常在#define 语句中使用头文件的文件名或其省略形式。这样试图再次包含同一文件的错误就可以被预处理器发现并处理。

3. 语句

在C++语言中，以分号结尾的句子称为语句。例如：

```
int a,b;
```

分号代表这个语句的结束。如果这一行的结尾没有分号，则这一个语句还没有结束。有时一个长的语句可以占不止一行，为了提醒读者注意这些行属于一个语句，应该以醒目的方式断行并在下一行往右缩排。例如：

```
hWnd = CreateWindow(
        szClassName,
        szTitle,
        ...,
        NULL;
  );
```

对于短的语句，也可以一行写多个语句。例如：

```
int a; float fp; char sc;
```

4. 标准输入与输出

cout 和 cin 称作标准输出/输入流，在 iostream.h 中定义，它表示标准输出/输入设备，标准输出一般指的是屏幕，标准输入指的是键盘。

运算符<< 把它的右边内容在屏幕上显示出来(变量与表达式的值或由双引号括起的字符串)。运算符>>将键盘中输入的一个数，送到它右边的变量中保存起来。上述程序运行之后，在屏幕上可以看到下面这条信息：

Enter two integer:

光标在冒号后面闪烁,等待输入两个数。例如输入第一个数 5,按一下空格键,再输入第二个数 25,然后按回车键,这时程序就读入所输入的数,5 送给 a,25 送给 b,然后开始执行以后的语句。

5. 新行

在程序中,"\n"称为新行符。C++系统还提供一个操纵算子 endl,它的功能和新行符一样,也是开始一个新行。

6. 主函数

这个程序中以 main 开始的部分定义了一个函数,该函数规定了该程序的功能。main 是函数名,在函数名之后紧跟一对圆括号。所有的 C++程序都必须有一个名为 main 的主函数。这是程序员和 C++系统之间的约定:程序执行的开始点是 main 函数中的第一条语句。

一个 C++函数中的任何成分被括在一对花括号("{"和"}")中,在函数 main 的后面的右圆括号后紧跟一个左花括号,表示"这个函数从这里开始",最后的右花括号表示"这个函数在这里结束",花括号括起来的部分称作函数体,而函数名 main 和它后面的一对圆括号称为函数头。函数体由一系列的 C++语句组成,这些语句描述这个函数怎样实现它的功能。

1.3 数据和表达式

程序是由数据和处理数据的操作组成的。本节先介绍 C++的最小程序单位——词法要素,然后简要介绍 C++的基本数据类型、常量、变量、运算符和表达式。

1.3.1 程序的词法符号

词法符号是程序中最小的不可再分的单位。C++共有六种词法符号,它们是:关键字、标识符、常量、字符串、运算符以及标点符号。本节介绍关键字、标识符以及标点符号的概念,常量、字符串和运算符将在后面的章节中陆续介绍。

1. 关键字

关键字是 C++的保留字,在程序中表示固定的意义,它们不能被重新定义用作它用,下面是 C++的关键字:

asm	auto	break	case	catch	char	class	const	continue
default	delete	do	double	else	entry	enum	extern	for
friend	goto	if	inline	int	lon	new	operator	overload
protect	public	register	return	short	signed	sizeof	static	struct
switch	template	typedef	union	unsigned	virtual	void	while	

其中 entry 和 catch 尚未在任何 C++中实现,保留供今后 C++发展使用。overload 在现

代的C++中已不再使用，它是旧版本的C++所使用的，最好不要在程序中作为关键字或其它名字使用。

2. 标识符

标识符是程序员定义的名字，用作变量名、函数名和类型名等。标识符由大小写字母、下划线和0～9的数字组成，组成标识符的规则是以字母或下划线开头，其后可跟数字、下划线、零个或多个字母。

标识符的长度可以是任意的，不同的C++编译器能识别的最大长度是有限的，编译器忽略掉多余的字符，而不认为是个错误。但如两个标识符的有效字符相同，则出现重定义错误。

在C++中，大小写字母是有区别的。为标识符起名字应使用有意义的单词或缩写来表明这个标识符的用途，这样就可以使程序具有很好的可读性，所以有时不得不使用较长的标识符。可以使用大小写字母或下划线分隔标识符的单词，例如：

```
DoubleList   doubleList   coord_X
```

其中coord是coordinate的缩写。

C++系统库中保存的符号信息都是以下划线开始的，最好不要定义以下划线开始的标识符，以免和C++系统库中的符号冲突，也不要在标识符内部使用连续的两个下划线。

3. 标点符号

C++的标点符号是：# ( ) { } , : " ' ; ...。

C++要求在程序的某些地方必须使用标点，但不表示任何实际的操作。例如，分号用于表示一条语句的结束。这些标点中的( )和{ }必须成对使用。

1.3.2 C++基本数据类型

C++的数据类型规定了它们的存储表示以及可以对它进行的操作，描述该类型的名字称为类型名。类型名可以是一个标识符，或像int这样的关键字，也可以是一个表达式，表示一种派生类型(详见派生一节)。基本数据类型是C++预定义的类型，是C++类型系统的基本组成部件，后面将要讲到的数组、指针和类等派生类型是程序员按C++的语法要求由基本数据类型建立起来的。基本数据类型是字符类型、整数类型(简称整型)、浮点类型、双精度类型和无值类型，描述相应类型的关键字是char，int，float，double和void。

字符类型的变量用于保存ASCII字符，整型变量用于保存整数，浮点类型和双精度类型的变量用于保存带小数点的数，但双精度类型比浮点类型有更大的值域。void类型在后面的章节讨论。

除void类型以外，其它类型前面可以加类型修饰符用于改变基本类型的含义，但它们仍然是基本数据类型。C++的类型修饰符是：signed(有符号的)、unsigned(无符号的)、short(短的)和long(长的)。修饰符signed和unsigned可以用于字符类型和整型，short和long可以用于整型，long还可以用于双精度类型。当用short、long、signed或unsigned修饰int时，关键字int可以省略。字符变量不仅可以用于存储ASCII字符的代码值，也可以用于存储－128～127或者0～255之间的整数值。

表 1.1 列出了 C++的基本数据型和类型修饰符的各种组合。

表 1.1 C++的基本数据型和类型修饰符的各种组合

类型名	位数	范围
char	8	−128 ～ 127
signed char	8	−128 ～ 127
unsigned char	8	0 ～ 255
int	16	−32768 ～ 32767
short int	16	−32768 ～ 32767
signed short int	16	−32768 ～ 32767
unsigned short int	16	0 ～ 65535
signed int	16	−32768 ～ 32767
unsigned int	16	0 ～ 65535
long int	32	−2,147,488,648 ～ 2,147,488,647
signed long int	32	−2,147,488,648 ～ 2,147,488,647
unsigned long int	32	0 ～ 4,294,967,295
float	32	3.4E−38 ～ 3.4E+38
double	64	1.7E−308 ～ 1.7E+308
long double	80	3.4E−4932 ～ 1.1E+4932
void	0	无值

用户可以通过声明数组、指针、结构或联合，以便从基本类型派生出复合类型的数据。

1.3.3 常量

常量是用来表示固定的数值或字符值的词法符号，在程序中不必进行任何说明就可以直接使用。

1. 整常量

整常量可以用十进制、八进制和十六进制来表示。十进制整常量由 0～9 的数字组成，但第一位不能以 0 开始，也没有小数部分。八进制整常量以 0 开始，由 0～7 的数字组成。而十六进制整常量以 0X(x 不分大小写)开始，由 0～9 的数字及 A～F 的字母(大小写均可)组成。在它们的前面都可用负号表示负数，例如：

555　　0545　　−0545　　0X5AF　　−0xabc

2. 浮点常量

浮点常量就是实数。在 C++中，浮点数可以用浮点表示，也允许科学表示法。例如：

−52.5　　0.515E+2　　65.5E−1　　35.8E5　　−6.8E2

浮点常量的数据类型为 double 类型，但可以在其后加后缀 F(f)使其为 float 类型，或加后缀 L(l)变为 long double 类型。例如：

−53.8F　　53.8f　　35.6E5F　　0.5L　　35.6E+5l

3. 字符常量

括在单引号中的一个字符称作字符常量，例如：' A' ,' $' ,' '（空格）。字符常量的类型是 char 类型。

在 ASCII 字符集中，有些代码表示控制命令，是不可显示的，有些是无法从键盘上直接输入的，由 C++提供的一种称作转义序列的方法来表示字符或控制代码。下面是 C++预定义的转义序列，它们都表示一个字符。

\a	响铃
\n	新行
\t	水平制表符(tab 键)
\v	垂直制表符
\b	回退符(backspace 键)
\r	回车符
\f	页馈进符
\\	反斜杠
\'	单引号
\"	双引号

必须使用双反斜杠来表示斜杠字符，例如，' \\' 。同样，若想表示单引号字符，我们应该使用' \' ' 。

可以用十六制或八进制值来表示 ASCII 码字符集中的任何一个字符或控制代码，这种转义序列以反斜杠开始：

\nnn	八进制值形式
\xnnn	十六进制值形式

其中 nnn 表示三位十六进制或八进制数。八进制转义序列在反斜杠后面紧跟八进制数组成的序列即可，而十六进制转义序列需要在反斜杠后面要接一个小写字母 x，以表示后面是由十六进制数组成的序列，例如，字符 a 的十六进制编码是 61，用转义序列表示字符 a 可以写成' \x61' 。

1.3.4 字符串

字符串常量(简称字符串)是用一对双引号括号起来的字符序列，例如：

```
"This is a string\n"
```

由于空格也是一个字符，所以字符串可以包含空格，字符串中也可以包含转义序列。

字符串是一种特殊的常量，它的类型不是基本数据类型，稍后再解释字符串的类型。

当两个字符串仅用空白相隔时，它表示一个串，例如：

```
"This is"  "a string"
```