C++程序设计教程

皮德常 张凤林 编著

国防工业出版社
National Defense Industry Press

C++程序设计教程

皮德常　张凤林　编著

国防工业出版社

·北京·

图书在版编目(CIP)数据

C++程序设计教程／皮德常，张凤林编著. —北京：国防工业出版社，2008.3 重印

ISBN 978-7-118-03583-4

Ⅰ. C... Ⅱ. ①皮... ②张... Ⅲ. C语言-程序设计-高等学校-教材 Ⅳ. TP312

中国版本图书馆 CIP 数据核字(2004)第 092268 号

※

国防工业出版社出版发行

（北京市海淀区紫竹院南路 23 号 邮政编码 100044）

新艺印刷厂印刷

新华书店经售

*

开本 787×1092 1/16 **印张** 18¼ **字数** 418 千字

2008 年 3 月第 2 次印刷 **印数** 4001—6000 册 **定价** 29.00 元

国防书店：(010)68428422 发行邮购：(010)68414474

发行传真：(010)68411535 发行业务：(010)68472764

前　言

C++是一种实用的程序设计语言，是高校学生学习程序设计的一门必修专业课程，同时也是编程人员最广泛使用的工具。学好C++，可以很容易地触类旁通其他语言，如Java和C#等。

本书是在作者讲义的基础上，总结教学和实践经验编写而成的。全书以面向对象的程序设计贯穿始终，针对初学者和自学者的特点，结合实例讲解了C++的基本概念和方法，力求将复杂的概念用简洁、通俗的语言描述，做到深入浅出、循序渐进。因此，适合用于大学计算机专业和非计算机专业的教学，也可供具有C语言基础的自学读者使用。

本书特点

1. 本书主要讲解了面向对象的程序设计理论和编程方法，它是计算机科学与技术专业学生的编程基础。

2. 在内容安排上，提前讲解I/O操作（许多书都是最后讲解）。因为文件是很实用，也是比较难学的一章，这为学生进行课程设计和实验做了铺垫。

3. 本书是作者教学经验的结晶。作者近年来一直从事程序设计方面的教学和科研工作，主讲过程序设计方面的多门课程，为此积累了丰富的教学经验。“从实践到理论，再从理论到实践，循序而渐进”是作者教学的心得体会，编写教材也不例外，作者深知学生的薄弱环节和学习特点。将自己的知识、授课方法和经验整理成书，是作者的梦想。

4.内容与时俱进，讲解了C++的许多新内容，这是其他教材所不具备的。例如：向量、string类和STL，许多教材都讲解不全，甚至不讲解这些新内容。作者认为，随着C++的发展，教材也应当与之同步。

5. 作业安排从易到难，环环相扣。许多学生学过C++，却不会编程。作者在教学中认识到了这一点，因此，设计了许多与实际有关的习题，并且它们彼此相关。

6. 分布实施课程设计。C++课程往往都有课程设计，许多院校是将课程设计放在课程结束以后，这就具有一个弊病：时间跨度太长，学生容易忘记学过的知识。不会做的学生，就抄袭别人的。为解决这个问题，本书直接在每章末尾给出课程设计的一部分，以使学过的知识能及时得到巩固。

7. 力求通俗易懂。编写本书的目的是让读者通过自学或在教师的讲授下，能够运用C++语言的核心要素，进行面向对象的程序设计。因此，本书围绕着如何进行C++编程展开。为了便于读者的学习，作者力求该书的语言通俗易懂，将复杂的概念用浅显的语言讲述，便于读者理解和掌握。

8. 在编排上，本书具有以下特点：

- 每章开始均引出本章要讲解的内容和学习要求。
- 每章安排的习题都具有很强的操作性，能通过计算机验证。
- 对书中重要的内容采用黑体标记，特别重要的内容采用下面加点标记。
- 本书强调程序的可读性。书中的程序全部采用统一的程序设计风格。例如：类名、方法名和变量名的定义做到“望名知义”；语句的末尾或下一句的开头放上左大括号，而右大括号自成一行，并采用缩排格式组织程序代码；此外，对程序中的语句还进行了尽可能多的注释。希望读者模仿这种程序设计风格。
- 本书强调程序的可移植性，不以某个C++开发工具为标准。
- 本书包含了大量的程序示例，并给出了运行结果。凡是程序开头带有程序名编号的程序，都是完整的程序，可以直接在计算机上编译运行。
- 本书采用了醒目的标记来显示知识点，这些标记是：警告、注意和思考等，它们穿插在内容中，帮助读者尽快找到重要的信息。

警告：这是警告信息，它们往往是容易混淆的知识点。

注意：值得你关注的地方，其级别略次于警告。

思考：提出问题，引导你思考，以培养你思考的能力。

教 学 支 持

本书的电子教案采用PowerPoint2000制作，可以在讲课时用多媒体投影演示，这可部分取代板书。教师不仅可以使用本教案，还可以方便地修改和重新组织其中的内容以适应自己的教学需要。使用本教案可以减少教师备课时编写教案的工作量，以及因板书所耗费的时间和精力，从而提高单位课时内的知识含量。

我们向使用本教材的教师免费提供本书的电子教案。需要本教案的教师可以直接与国防工业出版社联系。

本书的各章与其配套教材《C++习题解答与课程设计指导》一书中实验内容相配合，相辅相成。

感谢读者选择本书，欢迎您对本书的内容提出批评和修改建议，作者将不胜感激。作者的联系地址如下：

电子邮件地址：dc.pi@nuaa.edu.cn

通信地址：江苏省南京市南京航空航天大学信息科学与技术学院　　皮德常（收）

邮政编码：210016

内 容 简 介

C++是一种实用的程序设计语言，是高校学生学习程序设计的一门必修专业课程，同时也是编程人员最广泛使用的工具。学好C++，可以很容易地触类旁通其他语言，如Java和C#等。全书以面向对象的程序设计贯穿始终。

全书共7章，主要包括：C++程序设计基础；文件操作；类的基础部分；类的高级部分；继承、多态和虚函数；异常处理以及模板等等。书中列举了数百条可供直接使用的程序示例代码，并给出了运行结果，同时配有大量习题，还提供了该书的电子教案，使学生在学习时更为直观。

本书是在作者讲义的基础上，总结过去的教学和实践经验编写而成的。其中结合实例讲解了C++的基本概念和方法，力求将复杂的概念用简洁、通俗的语言描述，做到深入浅出、循序渐进。本书适合用做大学计算机专业和非计算机专业，并且学习过C语言的C++程序设计课程教材，也可供具有C语言基础的自学者使用。

目　录

第 1 章　C++程序设计基础

C++是在 C 语言的基础上扩充而成的，以其独特的机制在计算机领域有着广泛的应用。本章主要讲述 C++的基本知识，它是属于对 C 语言的扩充，因此有着承前启后的作用。

1.1　为什么要学习 C++程序设计

随着计算机软硬件技术的发展，计算机应用规模不断提高，在软件开发语言和工具方面不断地推陈出新，新语言、新工具层出不穷。目前，国内许多高校，无论是计算机专业或者是非计算机专业，都在开设 C 语言的基础上，陆续开设了 C++语言，并且将它作为学过 C 语言后的一门专业必修课程。

为了解决程序设计的复杂性，贝尔实验室于 1980 开始研制一种“带类”的 C，到 1983 年才正式命名为 C++。在计算机刚发明时，人们采用打孔机直接进行机器指令程序设计，当程序有几百条指令时，采用这种方法就困难了。后来人们设计了用符号表示机器指令的汇编语言，从而能够处理更大、更复杂的程序。20 世纪 60 年代出现了结构化程序设计方法（目前的 C 语言就采用这种方法），这使得人们能够容易编写较为复杂的程序。但是，一旦程序设计达到一定的程度，即使结构化程序设计方法也变得无法控制，其复杂性超出了人的管理限度。例如，一旦 C 程序代码达到了 25000 行~100000 行，系统就变得十分复杂，程序员很难控制，而发明 C++的目的就是为了解决这个问题，其本质就是让程序员理解和管理更大、更复杂的程序。因此，采用支持面向对象的 C++是时代发展的需要。

C++吸收了 C 和 Simula67（一个古老的计算机语言）的精髓，它具有 C 所无法比拟的优越性。C++在维持 C 原来特长（如效率高和程序灵活）的基础上，借鉴了 Simula67 面向对象的思想，将这两种程序设计语言的优点相结合。C++的程序结构清晰、易于扩展、易于维护同时又不失效率。目前，C++已超出了当初设计其的目的，成功地应用到数据库、数据通信等系统，并成功地构造了许多高性能的系统软件。C++与 C 相比，具有三个重要的特征，从而使其优越于 C：

- 第一个特征是支持抽象数据类型 ADT（Abstract Data Type，简称 ADT），在 C++中 ADT 表现为类，是对对象的抽象，而对象是数据和操作该数据代码的封装体，它提供了对代码和数据的有效保护，可防止程序其他不相关的部分偶然或错误地使用对象的私有部分，这是 C 所无法实现的。
- 第二个特征是多态性，即一个接口，多重算法。C++既支持早期联编又支持滞后联编，而 C 仅支持前者。

• 最后一个特征是继承性。继承性一方面保证了代码复用，确保了软件的质量，另一方面也支持分类的概念，从而使对象成为一般情况下的具体实例。

以上三个特性，我们将在后面的章节给予详细的讲解。

目前许多系统软件，如操作系统、数据库管理系统 DBMS 等都采用 C++所写，所以从事有关软件开发和计算机应用的人员，不掌握 C++简直寸步难行。因此，掌握 C++编程已成为许多专业学生的必然选择。

1.2 过程化程序设计和面向对象程序设计

C++支持过程化程序设计和面向对象的程序设计，它们是两种不同的编程模式。

在过程化程序设计中，程序员编写函数（也有些书称之为过程）。这些函数是执行某个特定任务的程序语句的集合，每个函数一般都包含局部变量，甚至还有全局变量。过程化程序设计是以过程（函数）为核心，而面向对象程序设计（Object Oriented Programming，简称 OOP）是以对象为核心。一个对象是一个包含数据和对数据操作的封装体。对象包含信息和对信息操作的能力，并且对信息的操作基于消息传递。

随着我们对 C++学习的逐步深入，将会深刻理解过程化程序设计和面向对象程序设计。

1.3 简单的输出和输入方法

C++除了具有 C 语言的输出和输入方法之外，还具有自己的输出和输入方法。简单的输出是通过 cout 对象，输入是通过 cin 对象，下面分别讲述它们的使用方法。

1.3.1 cout 对象

cout 对象只能用来输出数据，同时也称为标准输出对象，它的作用是使用标准输出设备（即显示器）输出信息。

☞ **注意**：cout 可以看作是 console output 英文单词的缩写。

cout 是输出流中的一个对象，因此也称为流对象。它的操作对象是数据流，要输出信息，只需将数据流传给 cout，例如：

```
cout << " I like programming language C++" ;
```

在上述语句中，“<<”是流插入操作符，它的功能是将字符串"I like programming language C++"送给 cout。程序 1-1 给出了输出数据的另外一种方法。

程序 1-1

```
#include  <iostream.h>
void  main ( )
{
cout << " I like programming language " << " C++" ;
}
```

程序运行结果：

I like programming language C++

 **注意**：程序的开头必须包含 iostream.h 文件，因为 cout 对象（和下一节的 cin 对象）就定义在该文件中，这就像 C 程序必须包含 stdio.h 文件一样。

通过上例可以看出，使用“<<”可以传送多个数据给 cout。例如，将程序 1-1 的 cout 语句修改如下：

```
cout << " I like programming language " ;
cout << " C++" ;
```

程序段的运行结果和程序 1-1 相同。但我们要理解一个重要的概念：虽然输出分解为两个语句，但程序仍在同一行上显示信息。除非你指定输出方式，否则送给 cout 的信息将会连续显示，程序 1-2 说明了这一点。

程序 1-2

```
#include   <iostream.h>
void   main ( )
{
    cout << "我最喜爱的东西：" ;
    cout << "computer " ;
    cout << " & tea " ;
}
```

程序运行结果：

我最喜爱的东西：computer & tea

输出结果与源代码中字符串的安排是不同的，cout 完全按照提交数据的方式输出。在上面的源代码中使用了 3 个输出语句 cout，但它仍在一行上输出信息，这是因为如果不加入换行符，cout 不会自动换行。我们有两种换行的方法：一种方法是在 **cout** 语句后加一个流操作符 **endl**；另一种方法是加入**'\n'**换行符。程序 1-3 就是这样定义的。

程序 1-3

```
#include   <iostream.h>
void   main ( )
{
    cout << "我最喜爱的东西："<< endl ;
    cout << "computer "<< '\n' ;
    cout << " & tea \n" ;
}
```

程序运行结果：

```
我最喜爱的东西：
computer
 & tea
```

注意：endl 是 end of line 的缩写，它和'\n'的功能一样都是换行。当 cout 遇到'\n'时，将输出光标移到下一行的开头。

警告：不要把反斜线 \ 和正斜线 / 弄混，'/n'是不会换行的；同时也不要在反斜线和字符 n 之间加空格，例如'\ n'就是错误的。

1.3.2 cin 对象

cout 是 **C++**的标准输出对象，**cin** 是 **C++**的标准输入对象，它的功能是从 **I/O** 控制台（即键盘）接收输入数据，如程序 1-4。

程序 1-4

```
#include   <iostream.h>
void   main ( )
{
    int   length, width, area ;

    cout << "计算矩形的面积 \n" ;
    cout << "输入矩形的长：" ;
    cin >> length ;
    cout << "输入矩形的宽：" ;
    cin >> width ;
    area = length * width ;
    cout << "矩形的面积为：" << area << "\n" ;
}
```

程序运行结果：

```
计算矩形的面积
输入矩形的长：10 [Enter]
输入矩形的宽：20 [Enter]
矩形的面积为：200
```

这个程序可以用来计算一个矩形的面积。当程序运行时，用户输入的数据将存储在 length 和 width 两个变量中，如下两行：

```
cout << "输入矩形的长：" ;
cin >> length ;
```

cout 在屏幕上显示“输入矩形的长：”，cin 为 length 变量输入值。其中“>>”称为流提取操作符，它从左边输入流对象 cin 中读一个数，并把它存储在>>右边的变量中。在上面的程序语句中，cin 将从键盘输入的数据存储在 length 变量中。

☞**注意**：插入操作符“>>”和提取操作符“<<”指定了数据的流动方向。插入操作符“>>”将输入的数据传给变量，而提取操作符“<<”将变量（或常量）传给 cout 输出。cin 对象在读取数据时，将暂停程序的运行，直到从键盘上输入数据并按 Enter 键确认。cin 对象能自动地将输入数据转换成与变量一致的数据类型，例如，用户输入“10”，cin 将分别读入字符'1'和'0'，在将该数据存储到变量之前，cin 能够自动地将它们转换成整数 10。cin 同样能够识别出，像“10.7”这样的数不能储存在整型变量中。如果用户输入一个浮点数给整型变量，那么小数点后的位数将被舍弃（也称截断）。如果用户输入浮点数，cin 通过截断浮点数后的小数部分，将整数部分存储在整形变量中。

如果程序要求输入数据，那么就应该向用户提示该输入什么样的数据。程序 1-5 是在程序 1-4 的基础上修改而成的，但缺乏提示用户的信息，因此它不是一个好程序。

程序 1-5

```
#include   <iostream.h>
void   main ( )
{
     int   length, width, area ;

     cin >> length ;
     cin >> width ;
     area = length * width ;
     cout << "矩形的面积为：" << area << "\n" ;
}
```

当运行这个程序时，用户面对的是黑屏幕，不知道要做什么事。一个功能完善的程序应当及时、友好地给用户必要的提示信息。

采用 cin 对象可以一次读入多个变量的值，例如程序 1-6 采用 cin 同时给变量 length 和 width 变量读取值。

程序 1-6

```
#include   <iostream.h>
void   main ( )
{
     int   length, width, area ;

     cout << "计算矩形的面积 \n" ;
     cout << "输入矩形的长和宽，中间用空格隔开：" ;
     cin >> length >> width ;        // 给两个变量读取值
```

```
    area = length * width ;
    cout << "矩形的面积为：" << area << "\n" ;
}
```

程序运行结果：

```
计算矩形的面积
输入矩形的长和宽，中间用空格隔开：10  20 [Enter]
矩形的面积为：200
```

下列语句等待用户输入两个数值，并把输入中的第一个数赋给变量 length，第二个数赋给变量 width：

```
cin >> length >> width ;
```

在上例输出中，用户输入 10 和 20，10 就送给了变量 length，20 送给了变量 width。当输入多个数值时，数值之间要加空格。cin 读到空格时，就能够区别输入中的各个数值，数值间的空格数无所谓，例如用户按照如下形式输入也可以：

```
10          20 [Enter]
```

但要注意的是，在最后一个数输入后要按 Enter 键。

☞ **注意**：cin 读取数据的特点和 scanf 函数类似，在上例中的输入中，如果先输入 10 并按 Enter 键，然后输入 20 再按 Enter 键，完全可以正确地输入数据。本书对数据的输入和输出格式不做过详细的探讨，因为这些不是 C++的核心和重点。

采用一个 cin，也可以同时为多个不同类型的变量读入数据，程序 1-7 给出了示例。

程序 1-7

```
#include   <iostream.h>
void   main ( )
{
    int    whole ;
    float   fractional ;
    char   letter ;

    cout << "请输入一个整数、一个浮点数和一个字符：" ;
    cin >> whole >> fractional >> letter ;
    cout << "整数："<< whole << endl ;
    cout << "浮点数："<< fractional << endl ;
    cout << "字符："<< letter << endl ;
}
```

程序运行结果：

```
请输入一个整数、一个浮点数和一个字符：100   3.14159   Y [Enter]
```