高等院校信息管理与信息系统专业系列教材

C++ 程序开发教程

张基温 编著

清华大学出版社
http://www.tup.tsinghua.edu.cn

高等院校信息管理与信息系统专业系列教材

C++程序开发教程

张基温 编著

清华大学出版社

(京)新登字 158 号

内 容 简 介

本书以程序设计的初学者为对象，以面向对象的程序设计方法为主线介绍了 C++ 程序设计的基础知识和方法，从一开始就引入面向对象的概念，并将面向对象的思想贯穿全书程序开发教程。全书共分 7 章：C++ 程序开发初步、算法与程序结构、数据结构、重载与模板、程序的类层次结构、I/O 流类库和 Visual C++ 程序设计。书中例题丰富，循序渐进，通俗易懂，思路新颖，每章都配有一定数量的习题或思考题。

本书适合作为本、专科学生程序设计课程的入门教材，也可作为计算机爱好者学习面向对象程序设计的自学教材。

图书在版编目(CIP)数据

C++ 程序开发教程/张基温编著. —北京：清华大学出版社，2002

高等院校信息管理与信息系统专业系列教材

ISBN 7-302-06032-0

Ⅰ. C… Ⅱ. 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(2002)第 084402 号

出 版 者：清华大学出版社（北京清华大学学研大厦，邮编 100084）

http:// www. tup. tsinghua. edu. cn

责任编辑：范素珍

印 刷 者：北京顺义振华印刷厂

发 行 者：新华书店总店北京发行所

开　　本：787×1092 1/16　**印张**：21　**字数**：507 千字

版　　次：2002 年 12 月第 1 版　2002 年 12 月第 1 次印刷

书　　号：ISBN 7-302-06032-0/TP・3598

印　　数：0001～8000

定　　价：26.00 元

出版说明

20世纪三四十年代，一直摸索着前进的计算技术与刚走向成熟的电子技术结缘。这一结合，不仅孕育了新一代计算工具——电子计算机，还产生了当时谁也没有料到的巨大效应：电子计算机——这种当初为计算而开发出来的工具，很快就超出计算的范畴，成为"信息处理机"的代名词；人类开始能够高效率地开发并利用信息；信息对人类社会的作用得以有效地发挥，并逐步超过材料和能源成为人类社会的重要支柱；信息产业急剧增长，信息经济高度发展，社会生产力达到了新的高度；人们的信息化意识不断加强，人类在信息资源方面开始更加激烈地竞争，社会发展走上信息化轨道。

文化是时代的精髓，是特定的人群在一定的历史时期、一定的地域范围对其生产和生活模式、思维和行为方式的觉悟和理性化，它伴随着人类创造和使用工具能力的提高而不断发展。文者，经天纬地也；化者，变化、改变、造化、习俗、风气也。也可以说，文化作为社会的人们在生产和生活中思维和行为方式的理性化，是文治和教化的结果。因此，文化具有区域性、群体性和时代性。在信息时代的帷幕刚刚拉开、新时代的气息开始弥漫社会各个角落的20世纪70年代，先觉们就已开始创办以加速信息化的进程为宗旨、以培养信息资源开发人才为目标的信息管理与信息系统专业。

从与信息有关的学科纵向来看，信息管理与信息系统专业处于信息学、信息技术、信息管理、信息经济、信息社会学这个层次结构的中间，它下以信息学和信息技术为基础，上与信息经济和信息社会学相联系。从其涉及的学科横向来看，它处在管理学、信息科学与技术和有关专业领域的交叉点上。它对技术有极高的要求，又要求对组织有深刻的理解，对行为有合理的组织，反映了科学与人本的融合的特点。这种交叉与融合正是信息管理与信息系统专业的最重要的特征，是别的学科或专业难以取代和涵盖的。

我国的信息管理与信息系统专业创建于20世纪70年代末。在近20年的时间里，已发展到151个点，成为培养信息化人才的重要领域。其发展速度之快、影响之深远已令世人和学术界刮目相看。然而作为一个新的、特别是与各行各业关系极为密切的专业，其课程体系、教学内容以及教学方法、手段，都要经历一个逐步完善、逐步成熟的过程，其教材体系的建设更需要较长期的实践和探索。没有这样一个过程，具有专业特点、符合中国实际的教材体系是不会被建立的。近20年来，大家一直在课程体系的完善和建设有自己专业特点的教材方面不断进行探讨。1991年全国10所财经类院校的经济信息管理专业的负责人在太原召开第一次研讨会。以后，1993年在大连、1995年在武汉、1997年在烟台，又有更多的院校参加到了这一研讨之中。这些研讨活动得到了国家教委有关部门的赞许和支持。通过研讨，大家在建设具有专业特点的教材体系、改变简单照搬其他专业教材上取得了共识。在武汉会议之后，即着手进行系列教材的编写工作。经协商，由张基温教授担任主编，由魏晴宇教授、陈禹教授担任顾问。

这套教材是我国信息管理与信息系统专业的第一套教材。尽管编写者为它付出了巨大的辛劳，但在实践中我们也深深地感到了时代的鞭策和工作的难度。一方面，席卷全球的信

息化大潮已经使信息、信息管理、信息系统成为全社会关注的热点，人们对其期望和要求越来越高；另一方面，在世纪之交的今天，作为现代社会先导技术的信息技术和相关学科的更新速度在不断加速，多种社会因素相互渗透、相互影响，前所未有的新情况、新问题给专业的建设带来很大的困难。当然，这些对我们专业的发展和建设也是一种动力和机遇。为此，在这套教材问世之际，我们再一次表示一个心愿：希望与全国的同行共勉，在教材和专业建设上齐心协力，做出更大贡献。也由于如上种种，这套教材不会是完整的，也不会是完美的，一定存在这样那样的不足或错误，我们将会不断补充，不断修改，不断完善。对于它的任何建设性意见，都是我们非常期盼的。为此，这一套教材将具有充分的开放性：每一本教材都是一个原型，每一位有志者对它的建设性意见都将会被采纳，并享有自己的知识产权，以使它们逐步成为精品。

全国高等院校计算机基础教育研究会
财经信息管理专业委员会
信息管理与信息系统专业系列教材编委会
1997年8月

前　言

一般说来，用面向对象的程序设计语言描述客观世界中的事物，容易理解，比较自然。但是，许多人的教学实践却表明，情况并非如此。多数学生反映，面向对象的语言难于理解，不容易掌握。

究其原因，一方面在于面向对象的语言机制一般要比面向过程的语言复杂，另一方面在于教学内容的组织上。许多人认为，面向对象的程序设计要以面向过程的程序设计为基础，学习 C++ 要先学好 C 语言。按照这样的思想组织教学的结果，可以说使学生初步奠定了一些关于程序设计的基本能力。但同时也使学生养成了按照面向过程思路解题的习惯。等到学习面向对象的程序设计时，这种先入为主的解题思路和习惯反倒成了学生掌握面向对象的程序设计方法的一条沟壑。本书就是在这方面进行的一个尝试，开门见山地将面向对象的思想引入，以先入为主的方式建立面向对象的思维方式，并将这种思想贯穿到整个教学过程中。

基于目前已经有一部分人在使用 Visual C++ 编写应用程序，本书中的程序就是基于这一背景的。但是，使用可视化的程序设计语言进行程序设计教学，存在一个陷阱和误区，容易将学生一开始就引入到程序界面的设计中去。其结果是，程序设计作业华而不实，界面虽漂亮，内容却空洞无物。这种引导是近年来学生程序设计能力普遍降低的一个重要原因。因此，本人不主张采用可视化的语言作为程序设计课程的教学语言。即使要采用可视化的语言作为教学语言，也要先隐藏界面程序设计部分，起码不要突出界面程序设计部分，要把学生的注意力吸引到算法和数据结构的设计这样一些最基本的程序设计的基本功方面，并以此作为对学生考核的主要内容。本书也是在这一方面的一个尝试。

以上两点就是本人写作本书的初衷所在。在本书的编写过程中，张伟、杨开莜、郎贵义帮助调试了书中的大部分程序，并提出了一些修改意见。在此对他们的辛勤劳动谨表谢意。

由于众多的原因，本书所进行的尝试一定还会有许多不足。希望广大同行、专家和读者不吝赐教，多提意见和建议，共同把这一教学改革做好。

张基温

2002 年 9 月

目　录

第1章 C++程序开发初步

计算机程序是用于描述计算机解题方法和方式的代码，这些代码可以被计算机直接或间接地理解并执行。从具体的问题出发，直到得到正确的程序代码的过程，称为程序设计。它是一个从具体(问题)到抽象再到具体(具体的程序设计语言描述的程序)的过程。

这里，“抽象”是指将具体问题抽象为一种模型。目前，程序设计中主要应用两大类模型：面向过程的模型和面向对象的模型。面向过程的模型世界是由一些数据(有简单的数据，有组合的数据)描述的；一个问题从原始状态到目标状态，是通过对数据的一系列操作——过程实现的。面向对象的模型就是经过对问题的分析，并构造出问题的数据及其间的联系——数据结构，以及对数据结构的操作过程——算法。因此人们把面向过程的程序概括为：

数据结构 + 算法

面向对象的模型认为世界是由一些对象(objects)组成的，每一个对象包括属性和方法两部分。属性是描述对象特征的一些数据或数据结构，方法表明对象的变化或一个对象对其他对象的作用，对象间通过消息传递进行通信。一个问题从原始状态到目标状态，是通过对象间相互传递消息时进行的。因此人们把面向对象的程序概括为

对象 + 消息传递

一般说来，面向过程的模型要用面向过程的程序设计语言描述，面向对象的模型要用面向对象的程序设计语言描述。C++是一种多范型的程序设计语言，既可以描述面向过程的程序，也可以描述面向对象的程序。本章介绍用C++描述程序的初步方法。

1.1 最简单的C++程序结构

1.1.1 一个简单的C++程序

下面是一个求两个整数相加的C++程序：

```
# include <iostream. h>

int main ()
{
    int x,y,s;                    // 定义3个整数:x和y代表2个加数,s代表和
    cout << "输入两个整数:";      // 给用户发出输入2个数的提示
    cin >> x >> y;                // 输入2个整数到x和y
    s = x + y;                    // 将x和y相加,送到s中
    cout << "x + y ="             // 输出提示
        << s                      // 输出s的值
        << endl;                  // 换行
    return 0;
}
```

说明：

(1) 程序中的"//"称为 C++ 的注释符，它后面的一串符号称为注释，即说明。C++ 也可以使用另一种注释格式，将注释写在"/ * "和" * /"之间。如程序中的第一句写为

```
int x,y,s;          /* 定义 3 个整数:x 和 y 代表 2 个加数,s 代表和 */
```

两者的区别是：前者表示单行注释，只能用于一行的尾部或独立一行；而后者表示段和多行注释，可以写在程序的任意位置，表示在"/ * "和" * /"之间的这段文字为注释。如：

```
/*定义 3 个数据 */int/* 表明后面的是整型数据 */ x /* 被加数 */,y/* 加数 */,s/* 和 */;
```

或

```
int x,y,s;
/* 该语句定义 3 个整型数据。其中" int " 表明后面的数据其数
 据类型为整型,"x" 代表被加数,"y"代表加数,"s"代表和 */
```

但前者不可这样写。

(2) ";"是 C++ 语句的结束符，表明一个语句的结束。因此，从一个语句开始，C++ 只有遇到";"才认为是该语句结束，而不管该语句占有几行。

(3) "x"、"y"和"s"称为 3 个变量，是 3 个存储数据的空间的名字。由于这些存储空间中存储的数据可以变化，所以称为变量。前面用"int"加以声明，表明这 3 个变量都只能用于存储整数。如果要存储带小数点的数——实数，应当用"float"或"double"进行说明。因为系统对于整数和实数分配的存储空间大小不一样：整数的存储空间较小，实数的存储空间较大。在实数中，用 float 说明为单精度数，它比双精度 double 声明的变量占用的存储空间小，但数的精度也低。

(4) "cout"称为标准输出设备，通常指显示器。"<<"称为插入符，即将程序中输出的字符插入——输出到显示器上的字符流中。"'输入两个整数：'"称为字符串。C++ 会将"<<"后面的字符串原样输出到标准输出设备——显示器上。即执行这一句后，将在显示器上显示：

```
输入两个整数:
```

意即提示程序用户要输入两个整数——被加数和加数。

(5) "cin"称为标准输入设备，通常指键盘。">>"称为提取符，即将由键盘上敲入的字符提取到内存中，分配给后面的变量(或者说保存到后面的变量代表的存储区中)。例如，当执行语句

```
cin >> x >> y;
```

后，将把从键盘上输入的两串数字——两个整数，分别分配给 x 和 y。要求两串数字之间用空格分隔。例如，输入

```
123  456↙
```

后，即将 123 分配给 x，将 456 分配给 y。

这时只能输入表示整数的数字串，因为前面已经说明 x 和 y 都是整型变量。输入非整

数字符串(如 1.23、abc),就将出错。

(6) 语句

```
s = x + y;
```

表示"将 x 和 y 的值相加,将结果赋值给变量 s"。这里,"+"为"加运算符";"="为"赋值运算符",即将其后面的数据送到其前面的变量中。

(7) 显然,整个程序中的语句都写在下面的框架中:

```
int main ()
{
    ……
}
```

main () 称为主函数。主函数是 C++ 程序的标志,每一个 C++ 程序都有并且只有一个主函数(Windows 程序中的主函数为 WinMain())。当然,除了主函数外,C++ 程序还可以有其他函数。有关内容将在 1.2 节中介绍。

(8) "endl"表示换行,如果后面还有输出语句,将另起一行。

(9) "# include"称为文件包含,即要把文件"iostream.h"中的内容包含到本程序中。".h"称为文件后缀,表明文件 iostream 是一个头文件。在 iostream.h 中含有对 cout 和 cin 的定义。为了使用 cout 和 cin,必须将 iostream.h 包含到程序中,否则程序员就要自己写关于 cout 和 cin 的定义,那是非常麻烦的事情。

1.1.2 运算符与表达式

运算符是一种程序记号,它作用于操作数而触发某种操作。由运算符和操作数组成的符号序列称为表达式。C++ 具有丰富的运算符。在本节中,先介绍几种最基本的运算符:算术运算符、赋值运算符、复合赋值运算符。

学习运算符,一是要了解运算符的含义;二是要了解运算符能作用于哪些数据,如求余只能作用于整数;三是要了解当一个表达式中含有多个运算符时,以什么顺序运算。

在 C++ 中,运算符在表达式中的运算顺序由运算符的两个主要特性决定:

- 运算符的优先级别;
- 运算符与操作数的结合方向;
- 运算符的先后排列顺序。

1. 最基本的 3 种运算符

表 1-1 中给出了算术运算符、赋值符和标准输入输出符的优先级别以及它们与操作数的结合方向。

对表 1-1 说明如下:

(1) C++ 提供有如下一些算术运算符:

- 单目算术运算符:-(负)
- 双目运算符:*(乘),/(除),%(模,求余),+(加),-(减)

加、减、乘、除四则运算的意义与算术中相同,只是符号略有差别。模(求余)运算是求两个整数相除的余数。例如:3 对 5 求模(3 % 5)得 3,29 对 8 求模(29 % 8)得 5 等。

表 1-1　算术运算符、赋值符和标准输入输出符的优先级别和结合性

运　算　符			结合性	优先级别
单目减	−		←	高 ↓ 低
双目算术运算符	*	/	→	
	+	−	→	
标准输入输出	≫	≪	→	
赋值	=		←	

(2) 赋值运算是一种改变存储区域内容的操作，即用右操作数来修改左操作数表示的存储空间的内容。该左操作数必须是一个可修改的左值——变量。

应当注意，在 C++ 中，“=”是赋值运算符而不是等号，下面的表达式是将 x 中的值加y，再存(送)到 x 中。

```
x = x + y;
```

(3) 优先级别由高到低依次是：单目减→乘、除→加、减→输入/输出→赋值。例如，表达式 x = 20 + 2 * − 6 的执行顺序为

```
x = 20 + 2 * −6
        ①
      ②
    ③
  ④
```

最后进行的是赋值运算，结果是 x 的值为 8。

(4) 对于结合性的判断原则是：相同的几个运算符相邻时，先对哪一个进行操作。算术运算符具有从左向右的结合性，也就是说几个相同的算术运算符相邻时，要按“从左至右”的规则进行运算。例如，有下面的表达式：

```
x= 4 + 5 + − 8
   ②   ①
     ③
   ④
```

运算符≪也具有从左向右的结合性，因此在下面的表达式中要先进行圆括号中的运算，求得变量 a 的值将之输出，然后才输出一个换行操作。

```
cout ≪ ( a = 20 + 2 * − 6) ≪ endl;
```

这里，圆括号是强迫先进行其中的运算，否则将因赋值运算的优先级低于插入运算符而出错。

赋值运算符是从右向左结合的。因此表达式

```
i = j = k = 5;
```

相当于表达式序列

```
k = 5 ; j = k ; i = j;
```

2. 复合赋值运算符

为了说明复合赋值运算符的功能，先看下面的例子。对于赋值表达式

```
x = x + y
```

可以简洁地表示为

```
x += y
```

这里，“+=”就是一个复合赋值运算符。它既有赋值功能，又有加的功能。除加号外，其他算术运算符都可以与赋值号复合成复合赋值运算符。

应当注意，复合赋值运算符是把其右表达式作为一个整体进行运算的。

例 1.1.1

```
#include <iostream.h>
int main()
{
    int x, y;
    cout << "x = 3, y = 8, x *= y + 1:" << ( x = 3, y = 8, x *= y + 1 )<< endl;
    cout << "x = 3, y = 8, x = x * y + 1:" << ( x = 3, y = 8, x = x * y + 1) << endl;
    return 0;
}
```

运行结果：

```
x = 3, y = 8, x *= y + 1:27
x = 3, y = 8, x = x * y + 1:25
```

3. 增 1 与减 1 运算符

对于双目运算符表达式

```
x = x + 1 或 x += 1
x = x - 1 或 x -= 1
```

一种更简洁的表达是

```
x ++ 或 ++ x
x -- 或 -- x
```

“++”、“--”称为增 1 与减 1 运算符。它们是单目运算符。结合性与简单赋值运算符相同，但优先级高于任何双目运算符。

使用增 1、减 1 运算符应注意：

(1) ++与--有两种使用形式：前缀形式，即它们在操作数前，如++x，--x；后缀形式，即它们在操作数后，如 x++，x--。++表达式与--表达式独立使用时，前缀形式与后缀形式无区别，但它们在表达式中被引用时，结果是不同的。前缀形式是先增(减)1，后被引用；后缀形式是先被引用，后增(减)1。

例 1.1.2

```
#include <iostream.h>
int main()
{
    int x, y;
    x = 5, y = x++;                          // 先引用后增 1
    cout << "x=5,y=x++:" << y << ",x=" << x << endl;
    x = 5, y = ++x;                          // 先增 1 后引用
    cout << "x=5,y=++x:" << y << ",x=" << x << endl;
    return 0;
}
```

运行结果：

```
x=5,y=x++:5,x=6          (先引用后增 1)
x=5,y=++x:6,x=6          (先增 1 后引用)
```

(2) ++与--操作只能用于可修改的左值，下面两个表达式语句是非法的：

```
++(i+j);                 // i+j 不是左值
++i++;                   // i++不是左值
```

但下面的写法是合法的：

```
++i = 5;                 // ++i 是左值
```

1.1.3 数据类型

C++ 程序中的数据都属于一定的类型。使用类型的意义在于在如下几个方面对数据进行规范化：

- 为数据分配相应大小的存储空间；
- 确定数据值的范围；
- 确定数据的表示精度；
- 确定数据可以进行的运算种类。

C++ 中数据类型非常丰富，表 1-2 为 32 位系统中几种最常用的数据类型的比较。

表 1-2 32 位系统中几种最常用的数据类型的比较

数据类型关键字	意 义	存储空间	数的精度(有效数字)	数 的 范 围	运算类型
char	字符	8 位			+、-、%
short	短整数	16 位		-32 768～32 767	+、-、*、/、%
int	整数	32 位		-2 147 483 648～2 147 483 647	
float	单精度浮点	32 位	7 位	$\pm(3.4\times10^{-38}\sim3.4\times10^{38})$	+、-、*、/、%
double	双精度浮点	64 位	16 位	$\pm(2.23\times10^{-308}\sim1.79\times10^{308})$	
long double	长双精度	80 位	18 位	$\pm(3.37\times10^{-4\,932}\sim1.18\times10^{4\,932})$	