面向21世纪高职高专计算机系列规划教材

COURSES FOR VOCATIONAL HIGHER EDUCATION: COMPUTER

C++语言程序设计

C++ PROGRAMMING

尹季昆　主编

科学出版社
www.sciencep.com

面向21世纪高职高专计算机系列规划教材
COURSES FOR VOCATIONAL HIGHER EDUCATION: COMPUTER

C++语言程序设计

尹季昆　主编

科学出版社
北　京

内 容 简 介

本书是学习 C++语言以及面向对象程序设计语言的较好教材。全书共分为两大部分：第一部分分为 6 章，分别介绍了 C++概述、类和对象、派生类与继承、多态性、模板、C++的 I/O 流类库；第二部分有 5 个实验，读者通过实验与练习加深对第一部分的理论知识的理解。书中所有例题均已通过上机测试，为帮助学生巩固学习效果，每章后还配有习题。

本书内容新颖，体系合理，逻辑性强，文字流畅，通俗易懂，可作为二年制或三年制高职高专院校计算机类各专业的教材，也可作为非计算机专业学生的选修教材，并且可供自学人员参考。

图书在版编目(CIP)数据

C++语言程序设计/尹季昆主编. —北京：科学出版社，2006
(面向 21 世纪高职高专计算机系列规划教材)

ISBN 7-03-016592-6

Ⅰ. C… Ⅱ. 尹… Ⅲ. C 语言-程序设计-高等学校：技术学校-教材
Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2005）第 146086 号

责任编辑：万国清 孙露露 / 责任校对：耿 耘
责任印制：吕春珉 / 封面设计：飞天创意

科学出版社 出版
北京东黄城根北街16号
邮政编码:100717
http://www.sciencep.com
双青印刷厂 印刷
科学出版社发行 各地新华书店经销
*
2006 年 1 月第 一 版 开本：787×1092 1/16
2006 年 1 月第一次印刷 印张：8 1/4
印数：1—3 000 字数：177 000

定价：15.00 元

（如有印装质量问题，我社负责调换<双青>）

销售部电话 010-62136131 编辑部电话：010-62138978-8004（V102）

面向21世纪高职高专规划教材专家委员会

主　任　李宗尧

副主任（按姓氏笔画排序）

丁桂芝　叶小明　张和平　林　鹏
黄　藤　谢培苏

委　员　（略）

信息技术系列教材编委会

主　任　丁桂芝

副主任（按姓氏笔画排序）

万金保　方风波　徐　红　鲍　泓

委　员（按姓氏笔画排序）

于晓平　马国光　仁英才　王东红　王正洪
王　玉　王兴宝　王金库　王海春　王爱梅
邓　凯　付百文　史宝会　本柏忠　田　原
申　勇　任益夫　刘成章　刘克敏　刘甫迎
刘经玮　刘海军　刘敏涵　安志远　许殿生
何瑞麟　余少华　吴春英　吴家碧　吴瑞萍
宋士银　宋锦河　张红斌　张环中　张海鹏
张蒲生　张德实　李云程　李文森　李　洛
李德家　杨永生　杨　闯　杨得新　肖石明
肖洪生　陈　愚　周子亮　周云静　胡秀琴
赵从军　赵长旭　赵动庆　郝　梅　唐铸文
徐洪祥　徐晓明　袁德明　郭庚麒　高延武
高爱国　康桂花　戚长政　曹文济　黄小鸥
彭丽英　董振珂　蒋金丹　韩银峰　魏雪英

出版前言

随着世界经济的发展，人们越来越深刻地认识到经济发展需要的人才是多元化、多层次的，既需要大批优秀的理论型、研究型的人才，也需要大批应用型人才。然而，我国传统的教育模式主要是培养理论型、研究型的人才。教育界在社会对应用型人才需求的推动下，专门研究了国外应用型人才教育的成功经验，结合国情大力度地改革我国的“高等职业教育”，制定了一系列的方针政策。联合国教科文组织1997年公布的教育分类中将这种教育称之为“高等技术与职业教育”，也就是我们通常所说的“高职高专”教育。

我国经济建设需要大批应用型人才，呼唤高职高专教育的崛起和成熟，寄希望于高职高专教育尽快向国家输送高质量的紧缺人才。近几年，高职高专教育发展迅速。目前，各类高职高专学校已占全国高等院校的近1/2，约有600所之多。教育部针对高职高专教育出台的一系列政策和改革方案主要体现在以下几个方面：

- “就业导向”成为高职高专教育的共识。高职高专院校在办学过程中充分考虑市场需求，用“就业导向”的思想制定招生和培养计划。
- 加快“双师型”教师队伍建设。已建立12个国家高职高专学生和教师的实训基地。
- 对学生实行“双认证”教育。学历文凭和职业资格“双认证”教育是高职高专教育特色之一。
- 高职高专教育以两年学制为主。从学制入手，加快高职高专教学方向的改革，充分办出高职高专教育特色，尽快完成紧缺人才的培养。
- 开展精品专业和精品教材建设。已建立科学的高职高专教育评估体系和评估专家队伍，指导、敦促不同层次、不同类型的学校办出一流的教育。

在教育部关于“高职高专”教育思想和方针指导下，科学出版社积极参与到高职高专教材的建设中去，在组织教材过程中采取了“请进来，走出去”的工作方法，即由教育界的专家、领导和一线的教师，以及企事业从事人力资源工作的人员组成顾问班子，充分分析我国各地区的经济发展、产业结构以及人才需求现状，研究培养国家紧缺人才的关键要素，寻求切实可行的教学方法、手段和途径。

通过研讨认识到，我国幅员辽阔，各地区的产业结构有明显的差异，经济发展也不平衡，各地区对人才的实际需求也有所不同。相应地，对相同专业和相近专业，不同地区的教学单位在培养目标和培养内容上也各有自己的定位。鉴于此，适应教育现状的教材建设应该具有多层次的设计。

为了使教材的编写能针对受教育者的培养目标，出版社的编辑分不同地区逐所学校拜访校长、系主任和老师，深入到高职高专学校及相关企事业，广泛、深入地和教学第

一线的老师、用人单位交流，掌握了不同地区、不同类型的高职高专院校的教师、学生和教学设施情况，清楚了各学校所设专业的培养目标和办学特点，明确了用人单位的需求条件。各区域编辑对采集的数据进行统计分析，在相互交流的基础上找出各地区、各学校之间的共性和个性，有的放矢地制定选题项目，并进一步向老师、教育管理者征询意见，在获得明确指导性意见后完成“高职高专规划教材”策划及教材的组织工作：

- 第一批“高职高专规划教材”包括三个学科大系：经济管理、信息技术、建筑。
- 第一批“高职高专规划教材”在注意学科建设完整性的同时，十分关注具有区域人才培养特色的教材。
- 第一批“高职高专规划教材”组织过程正值高职高专学制从3年制向2年制转轨，教材编写将其作为考虑因素，要求提示不同学制的讲授内容。
- 第一批“高职高专规划教材”编写强调
  - 以就业岗位对知识和技能需求下的教材体系的系统性、科学性和实用性。
  - 教材以实例为先，应用为目的，围绕应用讲理论，取舍适度，不追求理论的完整性。
  - 提出问题→解决问题→归纳问题的教、学法，培养学生触类旁通的实际工作能力。
  - 课后作业和练习（或实训）真正具有培养学生实践能力的作用。

在“高职高专规划教材”编委的总体指导下，第一批各科教材基本是由系主任或从教学一线中遴选的骨干教师执笔撰写。在每本书主编的严格审读及监控下，在各位老师的辛勤编撰下，这套凝聚了所有作者及参与研讨的老师们的经验、智慧和资源，涉及三个大的学科近200种的高职高专教材即将面世。我们希望经过近一年的努力，奉献给读者的这套书是他们渴望已久的适用教材。同时，我们也清醒地认识到，“高职高专”是正在探索中的教育，加之我们的水平和经验有限，教材的选题和编辑出版会存在一些不尽人意的地方，真诚地希望得到老师和学生的批评、建议，以利今后改进，为繁荣我国的高职高专教育不懈努力。

科学出版社

前　言

“C++程序设计”是二年制和三年制高职高专院校计算机专业的一门专业课程。随着计算机软件行业中面向对象程序设计思想的普及和发展，C++语言的强大功能和实用性越来越得到重视。考虑到高职高专学生的知识基础和就业需要，以及学制短、教学时数有限的实际情况，结合编者从事高职高专教学的实际经验，本书的编写突出实用性、技能性，在内容上便于二年制和三年制高职高专院校根据不同的教学时数选讲其中有关章节。

本书的主要内容和特点归纳如下：

1. 第 1 章主要讲述了面向对象思想的概况以及 C++语言的发展及特点，通过大量的程序实例，结合详细的理论讲解，使读者能尽快了解及熟悉 C++的编程思想和特色。

2. 第 2 章主要讲述 C++语言中类与对象的概念，并重点对构造函数和析构函数进行讲解。

3. 第 3 章至第 6 章是 C++语言的实际应用部分，其中第 3 章介绍类的继承与派生；第 4 章介绍多态性；第 5 章介绍模板；第 6 章介绍 I/O 流。这部分内容在理论教学的基础上力求突出实用性，以满足学生实际工作的需要。

4. 书后附有“上机实验指导”，共包括 5 个实验，对各重要知识点提供了加深理解的练习与实践，帮助学生熟悉和掌握 Visual C++的编程与测试环境。

书中所有例题及实验练习均经过上机调试。

本书通俗易懂，可作为二年制和三年制高职高专院校计算机专业教材，也可供相关专业学生或夜大、电大、函大学生以及自学考试人员等使用。

参加本书编写工作的有尹季昆、曹石、余江、文欣、程小梅、韩玉慧、武卫华、李美艳、高文玲等。

由于编者水平有限，书中难免存在不足和疏漏之处，希望广大读者批评指正。

目　录

第 1 章　C++概述

知识点

- 面向对象程序设计的基本概念
- C++的起源和特点
- C++源程序的构成
- C++在非面向对象方面的一些特性

难点

- 面向对象的概念
- C++源程序的构成
- C++在非面向对象方面的特性

要求

掌握：

- 面向对象的概念
- C++程序的格式与结构特点

了解：

- 类和对象的概念
- 面向对象的特性
- C++在非面向对象方面的一些特性

或许你已经学过 C 语言或 Pascal 语言，能用这些语言编写简单程序，解决某些具体问题。但在实际应用中，特别是要编制一些大型的程序或系统软件时，就会感到仅有这些是不够的，需要有新的设计方法来提高编程能力，以便适应软件开发规模日益庞大的趋势。20 世纪 90 年代以来，在计算机软件行业，面向对象程序设计思想已被越来越多的软件设计人员所接受。它是目前最先进的计算机程序设计思想和理念，这种新的思想更接近人的思维活动，利用这种思想和方法进行程序设计时，可以极大地提高编程能力，减少软件维护的开销。C++能完美地体现面向对象的各种特性。

1.1　面向对象程序设计的基本概念

面向对象的设计思想是在原来结构化程序设计方法基础上的一个质的飞跃，是一种新的程序设计理念，是软件开发的一种方法，其本质是把数据和处理数据的过程当成一个整体——对象。

面向对象程序的主要结构特点有两个：一个是程序一般由类的定义和类的使用两部分组成，在主程序中定义各对象并规定它们之间传递消息的规律；另一个是程序中的一切操作都是通过面向对象发送消息来实现的，对象接收到消息后，启动有关方法完成相应的操作。

面向对象设计的最大优点就是软件具有可重用性。当人们对软件系统的要求有所改变时，并不需要程序员做大量的工作就能使系统做相应的变化。

1.1.1　类与对象

类与对象是面向对象程序设计中最重要的概念，如果要掌握面向对象程序设计的技术，首先必须要很好的理解这两个概念。

1. 对象

从概念上讲，对象代表着正在创建的系统中的一个实体。在日常生活中，对象就是我们认识世界的基本单元。对象是现实世界中的一个实体，整个世界就是由各种各样的对象构成的。例如，一个人，一辆汽车，一个足球等。

2. 类

类是对象的模板，是对一组具有共同的属性特征和行为特征的对象的抽象。例如，由一个个大学生构成的“大学生”类，其中的每一个大学生是“大学生”类的一个对象。一个类的所有对象都有相同的数据结构，并且共享相同的实现代码。

类和对象之间的关系是抽象和具体的关系。

1.1.2　面向对象的特性

面向对象系统中最主要的特性是封装性、继承性和多态性。

1. 封装性

在面向对象程序设计中，数据的抽象是在确定类时强调对象的共同点而忽略了它们的不同点的结果。数据的封装则是隐藏了数据的内部实现细节的结果，将数据抽象的外部接口与内部的实现细节清楚的分开。

2. 继承性

以面向对象程序设计的观点来看，继承所表达的是对象与类之间的关系，这种关系使得某类对象可以继承另外一类的特征和能力。继承机制为程序提供了一种组织、构造和重用类的手段。继承使一个类（基类）的数据结构和操作被另一个类即派生类重用，在派生类中只需描述其基类中没有的数据和操作。这样一来，就避免了公用代码的重复开发，减少了代码和数据冗余。

3. 多态性

面向对象程序设计中的多态性，是指不同的对象收到相同的消息时所产生的多种不

同的行为方式。C++语言支持两种多态性，即编译时的多态性和运行时的多态性。编译时的多态性通过重载来实现；运行时的多态性是通过虚函数来实现的。程序运行的到底是函数的哪个版本，需要在运行时通过对象发送的消息来确定。

1.1.3　面向对象程序设计语言

我们要进行面向对象程序设计，必须使用面向对象程序设计语言。面向对象程序设计语言应该具备以下特征：

1）它支持对象的概念（包括对象的所有特性，如封装）。

2）它要求对象属于类。

3）它提供继承机制。

1.2　C++的起源和特点

1.2.1　C++的起源

1980 年，美国贝尔实验室的 Bjarne Stroustrup 博士在 C 语言的基础上，开发出一种过程性与对象性相结合的程序设计语言。这种语言弥补了 C 语言存在的一些缺陷，并增加了面向对象的特征。1983 年，这种语言正式定名为“C++”。

C 语言是 C++语言的基础，最初用作 UNIX 操作系统的描述语言。C 语言功能强、性能好，支持结构化程序设计，又能像汇编语言那样高效，伴随着 UNIX 的成功和广泛使用，诞生后立即获得了广泛的支持和好评。到了 20 世纪 80 年代，C 语言已经广为流行，成为一种应用最广泛的程序设计语言。

但是 C 语言也存在着一些局限：

1）C 语言的类型检查机制相对较弱，使得程序中的一些错误不能在编译时由编译器检查出来。

2）C 语言缺乏支持代码重用的语言结构。

3）C 语言不适合开发大型程序，当程序的规模达到一定程度时，程序员很难控制程序的复杂性。

C++语言正是为了解决上述问题而设计的。C++语言继承了 C 语言的精髓，如高效率、灵活性等，并增加了面向对象机制，弥补了 C 语言不支持代码重用的不足，这对于开发大型的程序非常有效。C++语言成为一种既可用于表现过程模型，又可用于表现对象模型的优秀的程序设计语言。

1.2.2　C++语言的特点

C++语言现在得到了越来越广泛的应用，它除了继承 C 语言的优点之外，还拥有自己独到的特点，最主要的有：

1）C++语言保持与 C 语言兼容，这就使许多 C 程序代码不用修改就可以为 C++语言所用，特别是一些用 C 语言编写的库函数和实用软件可以用于 C++语言中。

2）用 C++语言编写的程序可读性更好，代码结构更为合理。

3）C++语言生成代码的质量高，运行效率仅比汇编代码慢 10%～20%。

4）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、可靠性等方面有了很大的提高，使得大型的程序开发变得更加容易。

5）支持面向对象的机制。

总之，目前 C++语言的优点正越来越得到人们的认可和推崇，它已经成为被广泛使用的通用程序设计语言。在国内外使用和研究 C++语言的人数正迅猛增加，优秀的 C++版本和配套的工具软件也不断涌现。

1.3　C++源程序的构成

1.3.1　C++程序的一般格式

C++是 C 的一个超集，它几乎保留了 C 的所有特性。下面通过一个求两个数中较大值的简单的 C++程序，来对 C++程序的格式有一个初步的了解。

例 1.1

```
//max.cpp
#include <iostream.h>
int max(int a,int b);          //函数原型的说明
void main( )                   //主函数
{
  int x,y,temp;                //定义三个整型变量
  cout<<"Enter two numbers:"<< "\n";  //提示用户输入两个数
  cin>>x;                      //从键盘输入变量 x 的值
  cin>>y;                      //从键盘输入变量 y 的值
  temp=max(x,y);               //调用 max 函数，将得到的值赋给变量 temp
  cout<<"The max is:"<<temp<< "\n" ; //输出两个数中较大的值
}
int max(int a,int b)           //定义 max 函数，函数值为整型
{
  int c;                       //定义一个整型变量
  if(a>b)
  c=a;
  else
  c=b;                         //判断两个数的大小，将较大值赋给 c
  return c;                    //将 c 的值返回
}
```

本程序用来计算两个整数中较大的值。它由两个函数组成：主函数 main()和被调用函数 max()。函数 max()的作用是判断 a 与 b 的大小，把其中较大的值赋给变量 c。return 语句把 c 的值返回给主函数 main()。返回值通过函数名 max 带回到 main()函数的调用处。

程序的第 1 行是注释语句，由“//”开始，到行尾结束，这条注释语句注明了本程序的文件名为 max.cpp。

程序的第 2 行是预编译命令#include <iostream.h>，它的作用是，指示编译程序把头文件 iostream.h 包含进#include 命令所在的源程序中。iostream.h 是 C++系统定义的一个头文件，用于定义程序的输入和输出。

第 7 行中出现的“cout”语句的作用是，在屏幕上显示出字符串“Enter two numbers:”，“\n”是换行符，即输出上述信息后回车换行。

第 8 行和第 9 行中的“cin”的作用是分别输入变量 x 和 y 的值，即执行 cin 后，把从键盘输入的两个数值分别赋给变量 x 和 y。

第 10 行用来调用 max 函数，调用时把实际参数 x 和 y 的值传给函数 max()中的形式参数 a 和 b，执行 max 函数后得到一个返回值（即 max 函数中的 c），把这个值赋给 temp，然后第 11 行输出 temp 的值。程序运行情况如下：

```
Enter two numbers:
3↙
5↙
The max is:5
```

1.3.2　C++程序的结构特点

通过例 1.1，我们可以看出 C++程序的结构有以下特点：

1）C++程序由一组函数组成，函数是构成 C++程序的基本单位。其中有且仅有一个名为 main 的函数，称为主函数。程序运行时第一个被执行的函数必定是主函数，不论它在程序的什么部位。被调用的函数可以是系统提供的库函数，也可以是用户自己编写的函数（如例 1.1 中的函数 max()）。对于用户自己定义的函数，使用前应提供“声明”，如例 1.1 中的“int max(int a,int b);”。

2）C++函数由函数的说明部分和函数体两部分组成。

函数的说明部分包括函数名、函数类型、函数参数（形式参数）及其类型。例如在例 1.1 中的 max()函数的说明部分为

```
int      max     (int     a,      int     b)
 ↑        ↑       ↑       ↑       ↑       ↑
函数类型  函数名  形参类型  形式参数  形参类型  形式参数
```

函数类型规定为函数返回值的类型，如 int,float 等。无返回值的函数是 void 类型。

函数可以没有参数，但对于无参函数，函数名后面的圆括号不能省略。

函数说明部分下面的花括号内的部分称为函数体。函数体中的内容也就是函数的定义部分，主要是给出该函数的功能和执行流程。

3）C++中每一个语句和数据定义必须以分号结束。一行程序内可以写多个语句，一个语句也可以分写在多行上。

说明：

1）C 源程序文件扩展名为.c，而 C++源程序文件扩展名为.cpp。

2）常用的 C++版本，如 Visual C++或 Borland C++都带有 C 和 C++两种编译器，当源程序文件扩展名为.c 时，启动 C 编译器；当源程序文件扩展名为.cpp 时，启动 C++编译器。

1.4　C++在非面向对象方面的一些特性

C++语言是从 C 语言发展而来，因此 C 程序中大部分的特点和功能，在 C++中仍可

以使用。但C++语言还增加了很多C语言不具备的新特性，这些特性中除了“面向对象”的概念外，同时也包括一些非面向对象的新特性。我们下面就来介绍这些非面向对象的新特性，它们使得C++程序比C程序更简洁、安全、强大。

1.4.1 注释

在C语言中，我们用“/*”及“*/”作为注释分界符号，例如：

```
/*  This is
    just a
    test for program   */
```

C++语言保留了这种注释方式，同时还增加了另一种注释方式，该注释以“//”开始，到行末结束。例如：

```
Temp = a+b;  //This is just a comment
```

“//…”注释方式适合于注释内容不超过一行的注释，使用简洁方便。

1.4.2 I/O流

C语言中进行输入/输出，是依靠系统提供的函数来完成，如标准输入和输出函数scanf和printf。相比C语言，C++语言使用了更安全和强大的方法来进行输入/输出操作，也就是“流”的概念。例如：

```
int i;
double f=8.5;
cin>>i;
cout<<f;
```

这里的cin是标准输入流，在程序中用于代表标准输入设备，即键盘。运算符“>>”称为“提取运算符”，表示将从标准输入流（即键盘）读取的数值传送给右方指定的变量。也就是说，对于语句“cin>>i;”，用户从键盘输入的数值会自动地转换为变量i的数据类型，并存入变量i内，类似于C语言中的scanf(“%d”,&i);。

运算符“>>”允许用户连续输入一连串数据，例如：

```
cin>>a>>b>>c;
```

它将按顺序从键盘上接收所要求的数据，并存入对应的变量中。两个数据间用空白符（空格、回车或Tab键）分隔。

cout是标准输出流，在程序中用于代表标准输出设备，通常指屏幕。运算符“<<”称为“插入运算符”，表示将右方变量的值显示在屏幕上。例如，执行下面的语句后：

```
cout<<f;
```

变量f的值将显示在屏幕上，类似于C语言中的printf(“%f”,f);。f必须是基本数据类型，而不能是void类型。运算符“<<”允许用户连续输出一连串数据，也可以输出表达式的值，例如：

```
cout<<a+b<<c;
```

它将按照顺序将数据依次输出到屏幕上。

说明：

1）程序中如果需要使用cin或cout进行输入/输出操作时，则程序中必须嵌入头文件iostream.h，否则编译时要产生错误。下面是一个输入/输出流的例子。

例 1.2

```
#include <iostream.h>
void main()
{
  char name[20];
  cout<<"please input your name:";
  cin>>name;
  cout<<name<<endl;
}
```

2）在 C++程序中，我们仍然可以用 C 语言的传统方式进行输入/输出操作，即沿用 stdio 函数库中的 I/O 函数，如 printf()函数、scanf()函数或其他的 C 语言输入/输出函数。

3）在 C 中，常用“\n”实现换行，C++中增加了换行控制符 endl，其作用与“\n”一样。它的使用很方便，只要插入在输出语句中需要换行的相应位置即可。例如，以下两个语句的操作是等价的：

```
cout<<"x="<<x<<endl;
cout<<"x="<<x<<"\n";
```

1.4.3　灵活的局部变量说明

在 C 语言中，所有的局部变量说明必须置于可执行代码段前面，而不允许局部变量的说明出现在可执行代码的中间或后面。例如，在 C 语言中，下面的程序段是不正确的：

```
void func()
{
  int m;
  m=8;
  int n;
  n=4;
}
```

C 语言的编译器在编译时会指示有错，因为其中变量定义语句“int n;”插在“m=8;”这句可执行语句之后了。但在 C++语言中，这是允许的，也就是说上面的程序编译时不会出错。

C++语言允许在代码块中的任何地方对局部变量进行说明，该变量从说明点到该变量所在的最小分程序末的范围内有效。允许这种灵活的局部变量定义，对于程序员编写较大型的复杂的函数时非常有效。

1.4.4　const 运算符

在 C 语言中，经常使用宏定义，也就是用#define 来定义常量，例如：

```
#define time 100;
```

而 C++语言提供了一种更灵活、更安全的方式来定义常量，即用 const 修饰符来定义常量。const 型的常量相比#define 的宏定义要灵活得多，同时提供了更强大的安全性。const 可以创建有类型的常量，例如：

```
const int time = 100;
```

这样定义后的 time 的值将不能修改。

注意：用 const 定义的常量必须在声明的时候初始化它的值，并且一旦初始化完成