ITjob®远标软件开发系列教材

大学计算机规划教材

C++程序设计

陈国志 丛 华 单 正 编著

- 适合C++的就业技能实训
- 知识点的筛选紧密结合实际应用
- 精彩的案例与知识点无缝配合
- 作者多年的教学与实际开发经验

ITjob® 定制型人才输送中心

中国水利水电出版社
www.waterpub.com.cn

ITjob®远标软件开发系列教材
大学计算机规划教材

C++程序设计

深圳市远标培训中心（ITjob）于2003年成立，是目前深圳最早成立、师资力量最强的IT就业培训机构，也是深圳IT就业培训唯一一家在教育局注册的合法培训机构。

作为深圳市紧缺人才培训示范基地和外包培训示范基地，中心受深圳市科工贸信息委的委托，每年进行一次深圳市软件企业人才需求调研工作，以便及时掌握企业的人才需求，不断更新课程体系，以做到快速、高质量地就业。

ITjob承接深圳市计算机行业协会（SZCIA）培训中心工作，SZCIA是1987年成立的代表深圳市系统集成企业、软件企业等的组织，华为、中兴、长城、TCL、联想、研祥等深圳大型IT企业均为其会员单位，也是ITjob的合作企业。ITjob已输送10000多名大学生到深圳软件企业工作，已和300多所高校和深圳3000多家软件企业建立长期紧密的合作关系。

ISBN 978-7-5170-0930-6
9 787517 009306
定价：32.00元

上架建议：编程语言与程序设计/C++

大学计算机规划教材

C++程序设计

陈国志　丛　华　单　正　编著

中国水利水电出版社
www.waterpub.com.cn

内 容 提 要

本书根据C++的知识体系结构共分为九章，第1章主要介绍面向对象设计的四大基本特点；第2章主要介绍类的成员的访问属性、构造函数、析构函数以及几种特殊类型的成员的特征；第3章主要介绍类的作用域、友元和运算符重载；第4章主要介绍类的继承方式、派生类的访问控制、派生类对象的初始化与清除、基类对象和派生类对象的转换和赋值以及多重继承与虚基类；第5章主要介绍多态的概念、实现多态的方法（虚函数）以及虚析构函数的作用；第6章主要介绍函数模板和类模板；第7章主要介绍标准的模板库以及它们的使用方法；第8章主要介绍基本的输入输出流和对文件的基本操作；第9章主要介绍名称空间和对异常的处理。

本书中对每个概念都配有大量的案例，以帮助读者更直观地理解繁杂的概念，能够起到事半功倍的效果。

本书适用于刚接触C++并准备进行系统学习的初学者，也可作为大学计算机相关专业的C++程序设计教材。

图书在版编目（CIP）数据

C++程序设计 / 陈国志，丛华，单正编著. -- 北京
：中国水利水电出版社，2013.6
大学计算机规划教材
ISBN 978-7-5170-0930-6

Ⅰ. ①C… Ⅱ. ①陈… ②从… ③单… Ⅲ. ①C语言－程序设计－高等学校－教材 Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2013)第120198号

策划编辑：周春元　　责任编辑：李　炎　　加工编辑：于杰琼　　封面设计：李　佳

书　名	大学计算机规划教材 **C++程序设计**
作　者	陈国志　丛　华　单　正　编著
出版发行	中国水利水电出版社 （北京市海淀区玉渊潭南路1号D座　100038） 网址：www.waterpub.com.cn E-mail：mchannel@263.net（万水） sales@waterpub.com.cn 电话：（010）68367658（发行部）、82562819（万水）
经　售	北京科水图书销售中心（零售） 电话：（010）88383994、63202643、68545874 全国各地新华书店和相关出版物销售网点
排　版	北京万水电子信息有限公司
印　刷	三河市铭浩彩色印装有限公司印刷
规　格	184mm×240mm　16开本　11.5印张　253千字
版　次	2013年6月第1版　2013年6月第1次印刷
印　数	0001—2000册
定　价	32.00元

I

前　言

在计算机课程体系中，C++程序设计是一门专业必修基础课程。C++由C发展而来，与C兼容。用C语言写的程序基本上可以不加修改地用于C++。从C++的名字可以看出它是C的扩展和超越。C++既可用于面向过程的结构化程序设计，又可用于面向对象的程序设计，是一种功能强大的混合型程序设计语言。C++可用于设计性能要求比较高的系统级程序，也可用来设计应用软件，C++程序易于维护、易于重用、效率高。若设计得当，还易于移植。C++中增加了很多新概念，比如虚函数、泛型、运算符重载、异常处理等。这些概念对于刚接触C++的学生来说还是很难理解的。作者根据多年的教学和实际开发经验编写了这本通俗易懂的《C++程序设计》。

根据C++的知识体系结构将全书分为九章，第1章主要介绍面向对象设计的四大基本特点；第2章主要介绍类的成员的访问属性、构造函数、析构函数以及几种特殊类型的成员的特征；第3章主要介绍类的作用域、友元和运算符重载；第4章主要介绍类的继承方式、派生类的访问控制、派生类对象的初始化与清除、基类对象和派生类对象的转换和赋值以及多重继承与虚基类；第5章主要介绍多态的概念、实现多态的方法（虚函数）以及虚析构函数的作用；第6章主要介绍函数模板和类模板；第7章主要介绍标准的模板库以及它们的使用方法；第8章主要介绍基本的输入输出流和对文件的基本操作；第9章主要介绍名称空间和对异常的处理。全书中对每个概念都配有大量的案例，以帮助读者更直观地理解繁杂的概念，能够起到事半功倍的效果。

如何学好C++是每个学习者必须面对的一个问题，在这里笔者提出几点意见供大家参考。第一、学习者要多阅读源代码，多上机实践。在编程时不要单纯只关注语法，在上机时要多用到编译器的调试功能，多用单步执行，从内存分配的角度去观察程序的运行。第二、理解概念一定要把概念融入到程序当中去理解，这样才能真正地理解概念的内涵。第三、要树立自己能学好的信心，对于不理解的程序和概念，一定要先独立思考，实在想不通的，可以在网上去查找资料，直到弄懂为止。

本书由湖北工程学院陈国志，ITjob的丛华和单正两位资深C++工程师共同完成了大纲的研讨、内容的订正、材料的收集、程序的调试等工作。本书中若有不足甚至错误的地方，诚盼各位专家和读者指正。

II

目　录

1 类和对象

本章主要内容：

面向对象程序设计的四大特点：抽象、封装、继承、多态。类和对象的基本概念和特征。C++程序设计的基本结构。

1.1 面向对象程序设计方法

面向对象设计方法是在结构化设计方法出现很多问题的情况下应运而生的。在结构化设计方法中，求解问题的基本策略是从功能的角度审视问题域。它将应用程序看成实现某些特定任务的功能模块，其中子过程是实现某项具体操作的底层功能模块。在每个功能模块中，用数据结构描述待处理数据的组织形式，用算法描述具体的操作过程。即我们要解决某一个问题，就要确定这个问题能够分解为哪些函数，数据能够分解为哪些基本的类型，如 int、char、double、struct 等。也就是说，思考方式是面向机器的，而不是面向问题域的，需要在问题结构和机器结构之间建立联系。

面向对象程序设计方法的思考方式是面向问题结构的，它认为现实世界由对象组成。面向对象程序设计方法解决某个问题，要确定这个问题由哪些对象组成，对象间的相互关系是什么。用对象描述事物，而每个具体的对象又可以用两个特征来描述，即描述事物静态属性所需的数据结构以及对这些数据进行的操作（动态属性）。也就是说，把数据结构和对数据的操作放在一起构成一个整体，这样才能完整地反映实际问题。数据结构和对数据的操作实际上是不可分割的整体。即面向对象程序设计的思想是把数据结构和对数据结构进行操作的方法（算法）封装在一起，形成一个个的对象。

面向对象程序设计具有四大基本特点：抽象、封装、继承、多态。

1. 抽象

抽象去掉了被研究对象中与主题无关的次要部分，而仅仅抽取出与研究工作有关的实质性内容加以考虑。抽象有两类：一类是过程抽象，另一类是数据抽象。面向对象程序设计强调数据抽象，数据抽象把系统中需要处理的数据和对这些数据的操作封装在一起，根据功能、性质、作用等因素抽象成不同的抽象数据类型。每个抽象数据类型既包含数据，又包含针对这些数据的操作，是相对于过程抽象更为严格的抽象方法。

2. 封装

封装就是利用抽象数据类型把数据和基于数据的操作封装在一起，数据被保护在抽象数据类型的内部，避免了外界的干扰和不确定性。对象的某些数据和代码可以是私有的，不能被外界访问，以此实现对数据和代码不同级别的访问控制。

封装包含两层含义：

①把对象的全部属性及其行为结合在一起，形成一个不可分割的独立单位（即对象）。

②信息隐蔽。即尽可能地隐蔽对象的内部细节，对外形成一个边界（或者说形成一道屏障），只保留有限的对外接口，使之与外部发生联系。

封装的原则反应在软件上体现为对象以外的部分不能随意存取对象的内部数据（属性），从而有效地避免了外部错误对它的“交叉感染”，使软件错误能够局部化，大大减少查错和排错难度。封装性降低了程序开发过程的复杂性，提高了开发效率和软件质量，保证了数据的完整性和安全性。同时，封装性提高了抽象数据类型的可重用性，使抽象数据类型成为一个结构完整、可自行管理的有机整体。

3. 继承

继承是指一个对象从另一个对象中获得属性的过程，是面向对象程序设计的原则之一。它支持按层次分类的概念，例如，波斯猫是猫的一种，猫又是哺乳动物的一种，哺乳动物又是动物的一种。如果不使用层次的概念，每个对象需要明确定义各自的全部特征。通过层次分类方式，一个对象只需要在它的类中定义体现其唯一性的各个属性，然后从父类中继承它的通用属性。正是由于继承机制，才使得一个对象可以成为一个通用类的一个特定实例。一个深度继承的子类将继承它在类层次中的每个祖先的所有属性。

4. 多态

多态是指不同事物具有不同表现形式的能力。多态机制使具有不同内部结构的对象可以共享相同的外部接口，通过这种方式减少代码的复杂度。将在第 5 章里详细介绍多态。

1.2 类的声明和对象的定义

1.2.1 类和对象的关系

类（class）是现实世界或思维世界中的实体在计算机中的反映，它将数据以及对这些数据

的操作封装在一起。类是对象的抽象，而对象是类的具体实例（instance），即类的实例化。类是抽象的，不占用内存，而对象是具体的，占用存储空间。类是用于创建对象的蓝图，它是一个包括了一些特定方法和变量的软件模板。在C语言中int是整数的基本类型之一，例如语句“int a;”表示a是int 类型的变量，a在VC 6.0的编译环境中占用4个字节。而int只是一种数据类型，是一种基本数据类型的抽象。而 C++中的类是一种自定义的数据类型，该数据类型的变量（也就是对象）才是具体的。

1.2.2 类的声明与对象的定义

类定义的一般形式为：

```
class 类名        //class是声明类的关键字
{
private:
    //私有成员数据（属性）和私有成员函数（方法）
    /*注意，这种类型的成员不能在类外直接访问*/
protected:
    //保护成员数据（属性）和保护成员函数（方法）
    /*注意，这种类型的成员也不能在类外直接访问*/
public:
    //公有成员数据（属性）和公有成员函数（方法）
    /*注意，这种类型的成员能在类外直接访问*/
};   //用一对花括号包起来的部分是类体。类声明以分号结尾
```

例 1.1 类的声明与对象的定义。

```
#include <iostream>
#include <cstring>
using namespace std;   /* C++标准程序库中的所有标识符都被定义在一个名为 std 的名称空间中。比如下面用到的标识符 cout，如果没有这一行代码，cout << "书名：" << m_cTitle << endl;就应该写成 std::cout << "书名：" << m_cTitle << std::endl;   */
class CBook
{
private:
    char m_cTitle[20];
    char m_cAuthor[20];
    float m_fPrice;
    //float m_fPrice=0;错误。在定义变量时不能初始化。这点和C语言不一样
    /*在类中定义了三个私有的成员变量，在类外不能直接访问这三个变量，只能由类内的函数 Print、SetAuthor、SetTitle、SetPrice 访问，实现了信息的隐蔽。*/
public:
    //在类中定义了四个公有类型的成员函数，对外提供访问私有成员变量的接口
```

```
    void Print()
    {
        cout << "书名：" << m_cTitle << endl; //cout<<格式化输出
        cout << "作者：" << m_cAuthor << endl; //endl 输出回车换行
        cout << "价格：" << m_fPrice << endl;
    }
    void SetTitle (char * sz)          //访问 m_cTitle 的接口
    {
        strcpy(m_cTitle, sz);          //字符串复制
    }
    void SetAuthor (char * sz)         //访问 m_cAuthor 的接口
    {
        strcpy(m_cAuthor, sz);         //字符串复制
    }
    void SetPrice(float pr)            //访问 price 的接口
    {
        m_fPrice = pr;
    }
};

int main()
{
    CBook book1;                       //声明创建一个类对象（实例化一个对象 book1）
    //book1.m_fPrice=28.5f;            //错误。私有属性不能在类外直接访问
    book1.SetPrice(28.5f);             //调用公有成员函数 SetPrice 设置 m_fPrice
    book1.SetTitle ("C++程序设计"); //调用公有成员函数 SetTitle 设置 m_cTitle
    book1.SetAuthor ("张三");
    book1.Print();                     //调用 print()函数输出信息
    return 0;
}
```

程序运行结果如下：

```
书名：C++程序设计
作者：张三
价格：28.5
Press any key to continue
```

注意：C++中类成员的访问权限默认是 private 属性，若不加任何声明，成员默认是 private 属性，因此上述代码中的第一个 private 可以省略。

成员变量的类型前面不可使用 auto、extern 和 register 等，也不能在定义时对变量进行初始化，如果将 float m_fPrice 写成 float m_fPrice = 0，编译器会报错。

关键字 private 和 public 出现的顺序和次数可以是任意的。

类定义中提供的成员函数是函数的原型声明，若在类外完成函数实现需要使用作用域操作符（::）来标识函数所属的类，即采用如下形式：

返回类型 类名::成员函数名(参数列表)

{

函数体

}

其中，返回类型、成员函数名和参数列表必须与类定义时的函数原型一致。如上述例 1.1 代码也可以写成例 1.2 的形式。

例 1.2 类中函数原型声明与实现分离举例。

```
//文件一 ex102.h
#ifndef _EX102_H
#define _EX102_H
class CBook
{
private:
    char m_cTitle[20];
    char m_cAuthor[20];
    float m_fPrice;
public:
    void Print();
    void SetTitle (char * sz); //类定义中成员函数是函数的原型声明，起到接口作用
    void SetAuthor (char * sz);
    void SetPrice(float pr);
};
#endif
```

如下做法可以保证省略号省略的部分只被编译一次。在第一次编译此文件时，_EX102_H 没有被编译，#ifndef 为真，……部分被编译之后再编译此文件时，_EX102_H 已经被编译过，#ifndef 部分为假，……部分不会再被编译。从而有效地解决了 C++头文件中重复包含这个令人头痛的问题。

```
#ifndef  _EX102_H
#define  _EX102_H
……
#endif
//文件二 ex102.cpp
#include "ex102.h" //包含类定义头文件
#include <iostream>
using namespace std;
```

```
void CBook::Print() //成员函数的实现
{
    cout << "书名：" << m_cTitle << endl;
    cout << "作者：" << m_cAuthor << endl;
    cout << "价格：" << m_fPrice << endl;
}

void CBook:: SetTitle (char * sz)
{
    strcpy(m_cTitle, sz); //字符串复制
}

void CBook:: SetAuthor (char * sz)
{
    strcpy(m_cAuthor, sz); //字符串复制
}

void CBook::SetPrice(float pr)
{
    m_fPrice = pr;
}
//文件三 book.cpp
#include "ex102.h" //包含类 CBook 定义头文件
int main() //主函数
{
    CBook book1;
    book1. SetTitle ("C++程序设计");
    book1. SetAuthor ("张三");
    book1.SetPrice(28.5f);
    book1.print();
    return 0;
}
```

程序运行结果如下：

```
书名：C++程序设计
作者：张三
价格：28.5
Press any key to continue
```

注意：如果用 VC 6.0 环境编译程序会出现下面的错误。

```
fatal error C1010: unexpected end of file while looking for precompiled header directive
```

此致命错误提示没有找到预编译指示信息中的头文件。问题一般出在通过添加文件的方

式添加了一些 cpp 文件到一个 MFC 程序，但该 cpp 文件并不是 MFC 程序，而是标准的 C++ 程序。

可执行如下操作解决此问题，右键单击项目工程中的 cpp 文件，在菜单 Project→Settings →C/C++→Precompile Header（预编译的头文件）中将此选项设置为第一项：Not using precompile headers（不使用预偏译头文件）。

1.3 本章小结

本章阐述了面向过程设计方法存在的问题，它主要是面向机器结构的，因而不能充分解决现实世界的问题。而面向对象设计方是法针对问题域的，将问题域分解成各种对象，把握对象之间的关联。用对象描述事物，而每个具体的对象又可以用两个特征来描述，即描述事物静态属性所需的数据结构以及对这些数据进行的操作（动态属性）。在程序设计中，用类的成员变量来描述事物的静态属性，用类的成员函数来描述事物的动态属性。私有成员不能在类外直接访问，在类外必须通过类提供的公共接口来访问。

习题

1．面向对象程序设计的思想是什么？

2．什么是类？

3．对象都具有的两方面特征是什么？分别是什么含义？

4．在头文件中进行类的声明，在对应的实现文件中进行类成员函数的实现有什么意义？

5．上机运行例 1.1、例 1.2，熟悉所用系统的上机方法与步骤。掌握类中函数原型声明与实现分离的程序设计方法。

2

类的成员变量和成员函数

本章主要内容：

对象中公有变量、私有变量、常量、静态成员变量的作用和相关用法。默认构造函数、构造函数、拷贝构造函数、析构函数、普通函数的区别和相关用法。

2.1 对象的创建和撤销

在例 1.1 中，通过自定义的公有成员函数 SetTitle、SetAuthor 和 SetPrice 实现了对成员变量的初始化，实际上，C++为类提供了两种特殊的成员函数来完成同样的工作。

一是构造函数，在对象创建时自动调用，用来完成对成员变量等的初始化及其他操作（如为指针成员变量动态申请内存空间等）。如果程序中没有显式地定义它，系统会提供一个默认的构造函数。在例 1.1 中没有显式定义类的构造函数，但是系统会提供一个默认的构造函数，它的形式基本如下：

```
CBook()
{
}
```

另一个是析构函数，在对象撤销时由系统自动调用，用以执行一些清理操作，如释放成员函数中动态申请的内存等（将在本章第 3 节中详细介绍）。如果程序中没有显式地定义它，系统也会提供一个默认的析构函数。在例 1.1 中就没有显式定义类的析造函数，但是系统会提供一个默认的析构函数。它的形式基本如下：

```
~CBook ( )
{
}
```

2.2 构造函数

构造函数是类的一种特殊成员函数。一般情况下，构造函数专门用来初始化对象的成员变量。所以最好不要在构造函数中执行与初始化无关的操作。构造函数在类实例化对象时自动执行，构造函数的名字必须与类名同名，它不具有任何类型，不返回任何值，即使是 void 也不允许。

2.2.1 函数重载

函数重载是指几个同名函数完成不同的功能，编译系统在编译阶段通过参数个数、参数类型、返回值来区分应该调用哪一个函数，实现的是静态的多态性。但是要注意，不能仅仅通过函数返回值不同来实现函数重载。也就是说，进行函数重载时要求同名函数在参数个数上不同，或者参数类型上不同，否则将无法实现重载。如在例 2.1 中，可以给函数 add()定义多个函数实现，该函数的功能是求和，即求两个或者多个操作数的和。

例 2.1 函数的重载。

```
#include <iostream>
using namespace std;
int add(int x, int y)
{
    return x + y;
}
float add(float x, float y)
{
    return x + y;
}
int add(int x, int y, int z)
{
    return x + y + z;
}
float add(float x, float y, float z)
{
    return x + y + z;
}
int main()
{
    cout << add(5, 10) << endl;
    cout << add(5.0f, 10.5f) << endl;
```