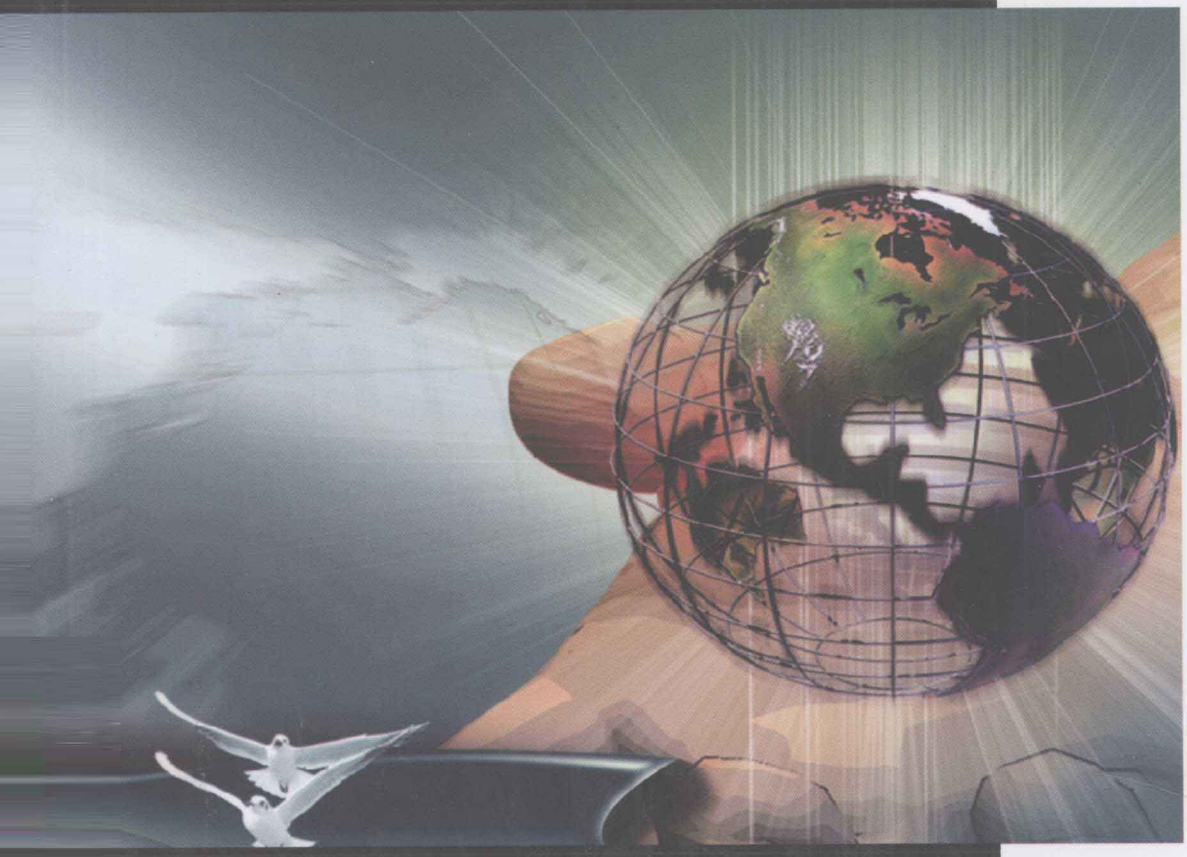
21世纪重点大学规划教材

刁奕　刁成嘉　等编著

C++面向对象程序设计



机械工业出版社
CHINA MACHINE PRESS

配电子教案

21 世纪重点大学规划教材

C++面向对象程序设计

刁　奕　刁成嘉　等编著

机械工业出版社

本书系统、详细地讲述了C++面向对象程序设计语言的基本语法格式和功能，通过大量的程序实例介绍如何利用C++语言编写一个高效率、高质量的面向对象的程序，以及一些编程技巧。还介绍了C++中的异常处理机制、强大的字符串处理功能、STL（标准模板库）提供的各种功能及其提供的通用算法和容器，以及通用设计模式和微软基础类库（MFC）等高级C++编程技术。

本书可以作为高等院校计算机和信息技术专业相关课程的教材，也可作为广大软件开发人员学习面向对象C++编程技术的自学指导书和技术参考书。

图书在版编目（CIP）数据

C++面向对象程序设计 / 刁奕等编著. —北京：机械工业出版社，2011.8
21世纪重点大学规划教材
ISBN 978-7-111-34359-2

Ⅰ.①C… Ⅱ.①刁… 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（2011）第137863号

机械工业出版社（北京市百万庄大街22号 邮政编码100037）
责任编辑：唐德凯 范成欣
责任印制：李 妍

唐山丰电印务有限公司印刷

2011年10月·第1版第1次印刷
184mm×260mm · 21.75印张 · 534千字
0001—3000册
标准书号：ISBN 978-7-111-34359-2
定价：42.00元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

电话服务
社服务中心：（010）88361066
销售一部：（010）68326294
销售二部：（010）88379649
读者购书热线：（010）88379203

网络服务
门户网：http://www.cmpbook.com
教材网：http://www.cmpedu.com

出 版 说 明

“211 工程”是“重点大学和重点学科建设项目”的简称，是国家“九五”期间唯一的教育重点项目。

进入“211 工程”的 100 所学校拥有全国 32%的在校本科生、69%的硕士、84%的博士生，以及 87%的有博士学位的教师；覆盖了全国 96%的国家重点实验室和 85%的国家重点学科。相对而言，这批学校中的教授、教师有着深厚的专业知识和丰富的教学经验，其中不少教师对我国高等院校的教材建设做过很多重要的工作。为了有效地利用“211 工程”这一丰富资源，实现以重点建设推动整体发展的战略构想，机械工业出版社推出了“21 世纪重点大学规划教材”。

本套教材以重点大学、重点学科的精品教材建设为主要任务，组织知名教授、教师进行编写，教材适用于高等院校计算机及其相关专业，选题涉及公共基础课、硬件、软件、网络技术等，内容紧密贴合高等院校相关学科的课程设置和培养目标，注重教材的科学性、实用性、通用性，在同类教材中具有一定的先进性和权威性。

为了体现建设“立体化”精品教材的宗旨，本套教材为主干课程配置了电子教案、学习指导、习题解答、课程设计、毕业设计指导等内容。

机械工业出版社

前　言

C++是一种使用广泛的程序设计语言，只有全面深入地了解和掌握 C++面向对象的基础知识和基本编程技巧，才能开发出一个高效率的面向对象的软件系统。

本书不仅系统地介绍了如何用 C++语言编写一个面向对象程序的基本编程技巧，还重点介绍了 C++中的异常处理机制、强大的字符串处理功能、STL（标准模板库）提供的各种功能及其提供的通用算法和容器，以及通用设计模式和微软基础类库（MFC）等高级 C++编程技术。本书通过大量的程序实例介绍如何利用 C++语言的这些高级对象技术开发一个高效率、高质量的面向对象的程序。

本书共分 8 章，各章内容如下：

第 1 章详细地讲述了 C++面向对象程序设计语言的基础语法格式和基本功能，通过大量程序实例介绍如何利用 C++语言编写一个面向对象的程序，以及一些编程技巧。重点介绍了 C++语言中关于类的封装、继承和多态性的基本原理的描述和实现。

第 2 章详细地介绍了 C++的异常处理机制。异常处理是 C++的主要特征之一。当系统产生致命错误时，该机制允许函数“抛出”一个异常对象，对应于不同的错误抛掷不同类型的对象；函数的调用者在独立的出错处理子程序中“捕获”并处理这些异常对象。采用异常处理机制是创建健壮程序的重要方法。

第 3 章深入地介绍了 C++强大的字符串处理功能。C++字符串类提供的各种成员函数可以对字符串进行方便、简洁、灵活的处理，具有极强的文本处理能力。C++也支持对宽字符和区域字符的操作。

第 4 章介绍了基于模板的 C++编程基础。模板是 C++面向对象程序设计语言的一个重要特征，也是学好面向对象程序设计需要掌握的。模板能够使程序员快速建立类型安全的函数集合和类的集合。同时，模板也是 C++语言支持参数化多态性的工具。模板功能强大，它的实现方便了大规模的软件开发工作，而且用模板技术开发的软件更容易维护。

第 5 章介绍了 C++标准模板库（STL）提供的标准通用算法。C++借助其模板功能提供了一大批功能强大、高效且易用的通用算法。用户也可以通过函数对象自定义标准算法。本章通过具体实例介绍了一些典型的算法。

第 6 章对 C++ STL 提供的标准通用容器和迭代器进行了较细致的介绍。C++容器采用了最简单的设计，以一种类型安全的方式提供对所有常见数据结构的支持。系统自动保证了容器中对象的同一性。独立于容器自身的迭代器可方便、灵活地对容器进行遍历，这是模板的又一杰作。这种巧妙的安排能够将标准通用算法灵活地应用于容器。

第 7 章介绍了通用设计模式的概念和实例。自从面向对象编程方法产生以来，最具革命性的飞跃是设计模式的引进。设计模式是对公认编程问题的有效解决方案，它独立于语言之外。因此，像单件（Singleton）、工厂方法（Factory Method）和观察者（Observer）等设计模式现在都已被一般的程序员所接受和使用了。

第 8 章简要地介绍了微软基础类库（MFC）的系统体系结构框架和使用 MFC 创建基于

图形界面系统的方法。MFC 实际上是一套开发模板，针对不同的应用和目的，程序员可以选择采用不同的模板。对程序的控制主要由 MFC 框架完成，MFC 提供了开发图形界面的大部分功能，并预定义或实现了许多事件和消息处理等。MFC 是 C++类库，程序员通过使用、继承和扩展适当的类来实现特定的目的，这大大减轻了开发人员的编程工作量。

本书的内容安排与标准 C++规范保持一致，1～7 章的程序实例在 Visual C++ 6.0 集成开发环境下通过调试运行。基于 MFC 图形界面系统的程序实例在 Visual Studio.NET 2003 集成开发环境下通过调试运行。

本书的第 1、3、4、8 章由刁奕编写，第 2 章由刁成嘉编写，第 5、6、7 章由赵亮编写，代会东、姚利娟、张亮、韩志远、刘政、邢远扬、杨志真、宋雪松、赵青、高建国、旷昊等参加了部分章节实例和习题的编写与调试工作，全书由刁成嘉统稿。由于作者水平所限，疏漏、欠妥、谬误之处在所难免，敬请读者批评指正。

编　者

目　录

第 1 章　C++面向对象程序设计基础

面向对象程序设计的 3 个重要特性是封装性、继承性和多态性。将所描述的客观事物的不同类型数据和对它们的操作封装成为一个集合体——类和对象，通过类的封装性，可以实现数据隐藏，便于程序的维护和修改。继承特性的引入，为代码的重复利用提供了更有效的手段，派生类继承了基类的所有特性，并改造基类的成员和添加新的成员。C++语言支持编译时的多态和运行时的多态，运行时的多态通过虚函数来实现，利用动态绑定实现的多态具有很高的灵活性。

本章目的

- 掌握类的说明和对象的创建方法及构造函数和析构函数的用法。
- 熟悉并掌握类的作用域、友元类的定义与使用。
- 掌握类继承的说明和定义方法及不同继承方式下的基类成员的访问控制机制。
- 熟悉并掌握派生类对象的初始化、清除和用虚拟继承方式解决多重继承产生的问题。
- 熟悉并掌握运算符重载的形式及虚函数的实现原理和 VPTR、VTABLE 的概念。
- 掌握纯虚函数和抽象类作为接口在面向对象编程中使用的一般方法。

1.1　类的定义和对象的创建

类是面向对象系统中最重要的概念，面向对象程序设计中的所有操作都归结为对类的操作。类是一组客观对象的抽象，组成类的对象均为该类的实例。本节重点讲述类的定义格式，成员的说明和定义方法，类成员的访问控制特性等内容。

1.1.1　类设计的基本概念

类的构造与设计是面向对象程序设计的支柱，是实现数据隐藏和封装性的基本单元。类的说明是一个抽象的概念。它声明了一种新的“数据类型”，描述了一类对象的共同特性，把数据项及其相关函数包容在一起。类是通过其成员的访问控制来实现隐藏与封装的。类的成员包括数据成员（也称属性）和函数成员（也称方法或操作）。类成员按其使用或存取方式分类，可以分为私有、公有、保护等，分别用关键词 private、public 和 protected 来修饰，为实现封装和继承机制提供条件。类的私有成员对外界来说是不可见的，只有在本类内定义的函数可以访问，不允许类外的代码操作和访问。类的 public 部分定义的数据变量和成员函数可以被类外代码访问，对外界来说是可见的，是类对象的外部接口。

在统一建模语言（Unified Modeling Language，UML）中，类的标识图符用矩形框表示。类的矩形框分为 3 部分：最上面是类名，中间为数据成员（属性），下面为函数成员（操作）。分别用+、-和#对应表示成员的公有、私有和保护等访问权限属性（可见性属性）。类在 UML 中的表示示例如图 1-1 所示。

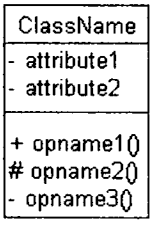


图 1-1　类在 UML 中的表示示例

1.1.2 类的定义格式

类的定义格式一般分为两大部分：说明部分和实现部分。说明部分用来说明该类中的若干成员，包括被说明的数据成员和成员函数。其中，数据成员也称为属性，成员函数又称为方法或操作，它们是用来对数据进行操作的。数据成员的说明包含数据成员的名字及其类型。成员函数可以在类体内定义，也可以只有函数原型说明。在类内定义的成员函数，按内联函数处理。类外实现部分主要用来对一些在类体内只有原型说明而没有定义的成员函数进行定义。如果一个类所有的成员函数都是在类体内定义的，则该类就没有类外实现部分。类定义的具体格式如下：

```
// 说明部分
class  <类名>
{ public:       <若干成员的说明或实现>
  private:      <若干成员的说明或实现>
  protected:    <若干成员的说明或实现>
};
// 实现部分
<若干函数成员的实现>
```

其中，class 为关键字；类名为标识符，类是具有唯一标识符的实体。一对花括号内包含的语句块是该类的类体，在类体中对该类的类成员进行定义和说明。

类成员包括数据成员和函数成员。数据成员通常是变量或对象的说明语句。函数成员是指函数的定义语句或说明语句。

public：关键字，指出其后的成员为公有成员。类外代码可以直接引用公有成员，它提供了类与外界的接口。通常将类的成员函数全部或部分定义为公有成员。

Private：关键字，可默认，指出其后的成员为私有成员。私有成员只能被本类定义的函数引用，通常将数据成员定义为私有成员。

类的成员属性还有一种称为保护的可见性属性（访问权限），用 protected 关键字修饰。它在不同的条件下具有公有成员或者私有成员的特性，在后面章节将详细论述。

【例 1-1】 定义一个类，描述平面上的一个点，要求操作包含获得该点坐标及移动该点。

```
//*********************************************************
//*  例 1-1  类的定义及其成员的表示方法                    *
//*********************************************************
class point                          // 定义一个名为 point 的类
{ private:                           // 定义私有成员
      float X,Y;                     // 定义点的 X 坐标和 Y 坐标
  public:                            // 定义公有成员
      void SetX(float x){ X=x;}
      void SetY(float y){ Y=y;}
      float GetX(void){return X;}
      float GetY(void) { return Y;}
      void Move (float x, float y) { X+=x; Y+=y; }
};
```

该类中有 7 个成员，其中有两个私有的数据成员，5 个公有的成员函数，成员函数都定义在类体内，为内联函数。通过该类中的成员函数可以分别设置和获得某个点的坐标值，并按规定的规则改变点的坐标。

定义类时要注意的事项如下：

1）在类中要指定成员的访问权限，如将一些数据成员和函数成员设为私有，就保证了程序运行的安全性。没有标明（默认）访问权限的成员为私有成员。

2）在类内说明访问控制属性的先后顺序没有规定，既可以先说明公有成员，也可以先说明私有成员。

3）在类体内进行数据成员定义时，不允许对数据成员进行赋值。

4）当在类体外定义函数时，在类体内应有函数原型说明。

5）类中的成员不能使用 auto、extern、register 等关键字进行修饰。

6）其他类的对象可以作类的成员，但自身类的对象不允许作该类的成员。

7）通常将类的定义部分存放在一个用户自定义的头文件中，可方便以后使用。

1.1.3 类的成员函数

类的成员函数描述的是类的行为和功能，是程序算法的具体实现。通常一个类的成员函数比较多，为了使类看起来简洁清晰，对成员函数的定义一般采用这样的策略：在类的定义中声明成员函数（函数原型），在类体外的实现部分定义该函数。在类体外定义成员函数时，必须使用“::”符号来限定，说明该函数是哪个类的函数，该符号为作用域运算符，用法见例 1-2。

【例 1-2】 类的成员函数的定义与实现。

```
//*********************************************************
//*  例 1-2  类的成员函数的定义与实现                      *
//*********************************************************
class string                                    // 定义一个串（string）类
{ private:
        int length;
  public:
        int getlengh(int);                      // 成员函数在类内进行原型声明
};
int string::getlengh(int n) {  return  n*length;  }   // 类的成员函数在类外实现格式
```

此例定义的 string 类只包含一个私有数据成员，一个公有成员函数，成员函数在类体内给出了原形说明，在类外实现定义。作用域运算符“::”指出函数 getlength( )是属于类 string 的成员函数。

在类体外定义成员函数时应注意以下几点：

1）在所定义的成员函数名之前应缀上类名，类名与函数名之间是作用域运算符“::”。

2）在定义成员函数时，对带参数的函数要给出参数的类型和参数名。而类内对应的该函数原型的形参可以只有类型，不写参数名。

3）函数的返回类型、形式参数的类型与类体内对应函数原型说明的类型要匹配。

4）当在类体外定义的成员函数较为短小时，可以定义为内联函数的形式，在定义时最前面用关键字 inline 修饰即可。如

```
inline int string::getlengh( ) {   return length;   }
```

1.1.4 类成员的访问控制

类成员访问权限的控制，实际上体现了类的隐藏和封装性，这种控制实际上是通过设置类成员的访问控制属性来实现的。由 1.1.1 节已知，访问控制属性有 3 种：public、private 和 protected。

公有类型定义了类的外部接口。对于类来讲，任何来自外部的访问都要通过外部接口进行。

私有成员不允许外部程序代码直接访问，这样私有成员就被隐藏在类中，对外界是不透明的。外部程序代码要访问私有成员，必须要通过类内定义的公有成员函数，这样就实现了访问控制和信息的安全性。

保护类型与私有类型有些相似，其差别是在类的继承和派生时对派生类的影响不同。这部分内容在学习类的继承和派生时将详细介绍。

定义一个类时，其成员函数可直接访问同类中的变量成员和调用同类中的成员函数。公有成员的作用域不仅包括类说明体，还包括它所属的对象的可视范围（静态成员除外）；私有成员或保护成员的作用域仅限于类的说明体和类的成员函数。

在 C++程序运行时，类是以其实例——对象之间的操作和交互访问完成系统的功能的。各个对象成员访问的原则是必须在成员名前加上对象名。因为类的每个对象都有自己成员的备份，成员只能存在于对象中，不能独立于对象直接给变量赋值。对于对象的成员访问，使用成员选择运算符“.”（用于一般对象）或指向符“->”（用于对象指针），将对象名与成员名联系起来。

1.1.5 对象的声明与使用

对象是类的一个实例。它属于某个类，在定义对象前必须先有定义好的类。一个类可以定义若干对象，各对象名字应具有唯一性，不同对象的数据成员值通常是不同的。

定义一个对象实例的格式如下：

```
<类名>  <对象名表>;
```

例如，point A;

其中，类名是指定义对象所属的类的名字，在上例中指 point 类，A 为 point 类定义的对象名。对象名表可以包含多个对象，即一个类可以定义多个同类对象，各对象之间用逗号分开。对象名可以是一般的对象名，也可以是指向对象的指针名或引用名，还可以是对象数组名等。

例如，point p1, p2, p[ ], &r=p1, *p3;

其中，p1 和 p2 是一般对象名，p[]是对象数组名。该数组的元素是类 point 的对象，r 是对象 p1 的引用名，p3 是对象指针。

引用名是 C++中为一个变量或对象起一个别名而设定的，引用名与对应的变量或对象使用同一个存储单元。在上面的描述中，引用对象 r 和一般对象 p1 指的是同一个对象，只不过该对象既称为 r，又称为 p1。

对象的成员表示方法：

<对象名> . <数据成员>　或　<对象名> -> <数据成员>

<对象名> . <函数成员>　或　<对象名> -> <函数成员>

在本例中，对象 p1 的成员就是例 1-1 中类 point 的成员，其成员函数可表示为

```
p1.SetX(10.5);          // 调用对象 p1 的成员函数 SetX( )，给私有成员 X 赋值
p1.SetY(6.8);           // 调用对象 p1 的成员函数 SetY( )，给私有成员 Y 赋值
p1.Move(2.6, 6.9);      // 调用对象 p1 的成员函数 Move( )，改变点的位置
p3->SetX(10.5);         // 调用对象指针 p3 的成员函数 SetX( )，给私有成员 X 赋值
p3->SetY(6.8);          // 调用对象指针 p3 的成员函数 SetY( )，给私有成员 Y 赋值
p3->Move(2.6, 6.9);     // 调用对象指针 p3 的成员函数 Move( )，改变点的位置
```

这里分别为对象 p1 和对象指针 p3 调用其公有成员函数，给私有成员 X、Y 赋值，并调用 Move( )函数改变点的位置。

一般来说，对象的使用有以下规则：

1）一般对象使用“.”操作符访问其成员，对象指针使用“->”操作符访问其成员。

2）同类对象之间可相互赋值，如 p1=p2。

3）对象可以作为函数的参数和返回值。

4）对象的成员也可以是对象。

5）可以说明对象数组、对象指针等，使用对象指针可以提高在函数间传递对象时程序运行的效率。

【例 1-3】 分析下面程序的输出结果，熟悉对象的定义及其成员的表示方法。

```
//*********************************************************
//*  例 1-3  对象的定义及其成员的表示方法                  *
//*********************************************************
#include<iostream.h>
class Tpoint
{ private:
     int X,Y;
public:
     void point(int x,int y);          // 只有函数原型，函数定义在类外进行
     int getX( ){ return X; }
     int getY( ){ return Y; }
     void Move(int,int);               // 只有函数原型，函数定义在类外进行
};
void Tpoint::point(int x,int y) { X=x; Y=y; }              // point(int x,int y) 函数的类外实现
void Tpoint::Move(int xset,int yset) {X+=xset; Y+=yset; }  // Move(int xset,int yset) 函数的类外实现
void main( )
{ Tpoint D1,*D2;                   // 说明 Tpoint 类的一般对象 D1 和对象指针*D2
  D1.point(10,12);                 // 一般对象的函数成员的引用方法
```