□ 中国高等职业技术教育研究会推荐

高职高专计算机专业规划教材

C++程序设计教程

主　编　崔志磊
副主编　苏　涛
主　审　杨俊清

西安电子科技大学出版社
2008

内 容 简 介

本书全面系统地介绍了C++面向对象程序设计的基本概念和方法。全书共分八章，主要包括面向对象的基本概念、类与对象、函数重载与引用、模板、运算符重载、继承与组合、多态和多继承以及流与异常处理等内容。另外，本书各章还配有丰富的例题和练习，可帮助读者加深理解并达到灵活应用的目的。

本书内容丰富，由浅入深，讲练结合，通俗易懂。

本书可作为高职高专计算机专业学生的教材，也可供相关人员参考使用。

★ 本书配有电子教案，有需要的老师可登录出版社网站，免费下载。

图书在版编目(CIP)数据

C++程序设计教程/崔志磊主编. —西安：西安电子科技大学出版社，2008.2

中国高等职业技术教育研究会推荐. 高职高专计算机专业规划教材

ISBN 978–7–5606–1951–4

Ⅰ. C… Ⅱ. 崔… 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(2007)第192837号

策　　划　臧延新

责任编辑　阎　彬　臧延新

出版发行　西安电子科技大学出版社(西安市太白南路2号)

电　　话　(029)88242885　88201467　邮　编　710071

http://www.xduph.com　E-mail: xdupfxb@pub.xaonline.com

经　　销　新华书店

印刷单位　陕西华沐印刷科技有限责任公司

版　　次　2008年2月第1版　2008年2月第1次印刷

开　　本　787毫米×1092毫米　1/16　印张21.25

字　　数　494千字

印　　数　1～4000册

定　　价　28.00元

ISBN 978 – 7 – 5606 – 1951 – 4 / TP • 1008

XDUP 2243001-1

* * * 如有印装问题可调换 * * *

本社图书封面为激光防伪覆膜，谨防盗版。

前　言

欢迎进入 C++的世界！C++所导入的面向对象技术引起了程序设计的一场变革，它使程序设计者体会到了编程语言的易用性与强大功能的完美结合。对于有志成为一名软件工程师的人士,学好并用好 C++就是其目标之一。

C++是目前使用最广泛的编程语言之一。它主要分为两大部分：面向过程与面向对象。面向过程部分的语法和结构在本质上与 C 一致，面向对象部分是 C++的精华所在。

本书以面向对象为切入点，着重介绍用面向对象的思想分析问题、解决问题的方法。全书共分八章，作者在章节和内容安排上作了详细的规划。各章通过提出问题、分析和讲解相结合的方式展开，将概念与实例有机结合，处处体现作者的用心。在每个实例之后，作者对结果进行了分析、解惑和辨义，帮助读者正确把握精髓。每章最后都进行了要点归纳，并配有不同难度的练习题。本书附录中提供了练习题参考答案，以便于读者自学。书中的所有例题和练习题都已在 VC 6.0 中调试通过。

本书由作者的讲稿整理而成，结合作者的教学经验，内容安排由浅入深。对于一些难点、要点和学习中的薄弱环节，书中进行了层层介绍和分析。本书以介绍方法为主，结合实例，引导读者逐步建立分析和解决问题的方法，掌握程序设计思想，进而最终掌握一个大型项目开发的方法与要领。因此，本书非常适合高职高专计算机专业学生使用，也适合程序设计人员和广大软件设计爱好者参考使用。

本书由崔志磊任主编，苏涛任副主编。崔志磊编写了第 1～4 章及第 8 章，并对全书进行了统稿。苏涛编写了第 5～7 章。

在此笔者特别要感谢西安电子科技大学出版社的臧延新编辑，他对本书的结构安排提出了建设性的意见。也要感谢陶文林老师，他为书中练习的编写做了大量的工作。

由于时间仓促和作者水平有限，书中难免存在错漏之处，敬请广大读者批评指正。

作者联系方式：cuiszxy@126.com。

作 者

2007 年 11 月

目　　录

第 1 章　面向对象的基本概念

面向对象程序设计简称 OOP(Object-Oriented Programming)，它从 20 世纪 80 年代开始逐步发展，现在已成为程序设计方法的主流。它将客观世界中描述事物的方法应用到了程序设计中，使程序语言成为现实世界的真实反映。

1.1　概　　述

1．软件设计方法的发展

软件开发早期采用结构化程序设计方法，程序被理解为：程序=(算法)+(数据结构)。即算法和数据结构是各个独立的整体，分别进行设计。

随后，软件工程师们开始逐步注重系统整体关系的表示和数据模型技术，他们把数据结构与算法看做一个整体的功能模块，于是程序被重新认识为：程序=(算法+数据结构)。即算法与数据结构是一个整体，算法总是离不开数据结构，算法含有对数据结构的访问。一个算法只能适用于特定的数据结构，想设计一个适合于访问多个数据结构的算法是不明智的，而且数据结构由多个算法来对其进行同种操作也是多余的。

此后，面向对象的程序设计方法被提出并获得了发展。

2．面向对象语言的特征

在面向对象程序设计中，算法与数据结构被捆绑成一个“类”，类进一步具体化为实物，即对象。于是程序被进一步理解为：程序=(对象+对象+…)，对象=(算法+数据结构)。

现实世界本身就是一个对象的世界，任何对象都通过它的属性和行为体现出来。即程序就是许多对象在计算机中相继表现的体现。从这样的角度看问题，我们在开发程序时就不用为程序功能如何实现而费尽心机了，而只要通过对象的组合就可以轻易实现。面向对象程序设计思想突破了软件思想的障碍，使得程序规模迅速扩大，软件产业得以飞速发展。面向对象程序设计是软件设计方法发展的必然结果。

C++是一种流行的面向对象的程序设计语言。它通过类机制，最有效地实现了面向对象的原理。它使得我们能从真实客观的角度来理解和开发应用程序。因此，可以说 C++是类机制比较完善的一种高级语言。

3．C++和 C

C++是一种高效实用的程序设计语言，它同时兼有面向过程和面向对象程序设计，因而得到了广泛应用。C++从 C 进化而来，是 C 语言的超集，C 中的语法规则在 C++中都适用。

C++也可以说是C的增强版，所以在名字中使用了C语言中的自增量运算符++，而形成C++。但是，C++并不是C语言的简单扩充，而是一个本质性的变革。它在继承C语言优点的基础上，提供了面向对象程序设计的能力。C++是一门优秀的、高效实用的、具有发展前途的程序设计语言。学好C++，很容易触类旁通地学好其他程序设计软件。

1.2 C++ 入门

1．输入与输出

C++有许多自己的特性，其中之一是将输入/输出改进为I/O流。例如，cout<<2就是将2这个数据送给显示器，cin>>a就是从键盘读取数据赋给a。cout和cin都是关键字。标识符cout代表屏幕，运算符<<用于把内容放进输出流，即将数据送到显示器。若有语句cout<<"I learn C++";，执行该语句之后就能在显示器上看到“I learn C++”。输出后若要换行可以使用语句cout<<"I learn C++"<<endl;，endl的作用是换行并刷新输出流。用cout<<的方式输出数据时，对于基本数据类型，它能自动匹配一种合适的类型输出，所以输出时不必再指出数据类型。标识符cin代表键盘，提取运算符>>用于把内容放进输入流，即把从键盘敲入的数据送到给定的单元中。

C++规定，凡使用输出流和输入流的程序，要用#include<iostream.h>包含头文件。这部分内容详见第8章。

2．标识符

在C++中，将由程序员命名的各名称，如变量名a等都称为标识符。在命名标识符时要注意几点：第一，标识符要以字母或下划线开头，后接由字母、数字或下划线组成的字符序列；第二，命名标识符时，不能使用已被系统定义的关键字；第三，可用大小写字符来区别不同的标识符。

表1-1列出了C++中的关键字。

表1-1 C++中的关键字

C和C++公用部分							
auto	brack	case	char	const	continue	default	do
else	enum	extern	float	for	goto	if	int
register	return	short	signed	sizeof	staic	struct	switch
double	long	typedef	union	unsigned	void	volatile	while
C++中新增部分							
asm	catch	class	delete	friend	inline	new	operator
private	protected	public	template	this	throw	try	virtual

1.3　类的基本概念

1.3.1　从结构到类

有一个学生的部分信息如图 1-1 所示，在 C 中可以用结构体类型描述如下：

```
struct student
{long num;
   char name[20];
   int score;
 };
struct student stu;
```

学号	姓名	学分
长整型	字符数组	整型

图 1-1　结构组成

上面的描述在 C 中定义了一个 student 的结构体，并定义了一个该结构体的变量 stu。在 C++中也有结构的概念，如：

```
struct student
{long num;
   char name[20];
   int score;
 };
student stu;
```

比较这两种方法，对于同样的定义，C++的描述只不过少了关键字 struct。在 C++中我们认为该描述定义了一个 student 的结构体，并定义了一个该结构体的对象 stu。stu 就是一个学生对象，学生是人，当然会有一些行为。为了表现行为，在 C++中的结构体内还可以定义函数，用它可以实现某一功能。

接上题描述一个学生的情况，并增加学习行为，在 C++中用结构体的方法可描述如下：

```
struct student
{long num;
   char name[20];
   int score;
   void Study();
 };
student Jonas;
```

该描述定义了一个名叫 student 的结构体，并定义了一个该结构体的对象 Jonas。于是就可用 Jonas.score=5;进行赋值，用 Jonas.Study()来调用成员函数表现行为。这两个语句的语意可描述为：Jonas 得到了 5 个学分，Jonas 去学习。

在 C++的结构体中可以定义函数。我们将结构内定义的函数称为成员函数，通过它可以实现某种功能来表现行为。

在 C 中，结构体内的成员是公开的，在结构体外部通过变量名引导就可以对成员进行访问，这样就很难保证数据的安全。为了能对结构体内成员的访问有所限制，C++中引入了“类”的概念。所谓“类”，是把事物的数据和与之相关的功能封装在一起，形成一种特殊的结构体，用以表示真实事物的一种抽象概念。在 C++中定义类的关键字用 class 而不用 struct。C++中的类是 struct 的一种形式。

类是 C++语言中的一种数据类型，它既可以包含描述事物特征的数据，又可以包含表现这些特征的函数。在类中的成员允许有 public(公有的)、private (私有的)和 protected(保护的)。

在面向对象程序设计方法中，程序是通过对象的表现来实现的，而对象的所有行为和特征都是由类来决定的。因此要建立对象，首先必须定义类。

定义类的一般方法如下：

```
class 类名
{private:                                    //标识成员的性质
    私有成员
 public:                                     //标识成员的性质
    公有成员
};
```

关键字 class 和类名组成类头，类名的命名要符合标识符的规定，通常第一个字母要大写。类的定义体用“{”和“}”括住，可将它称为类体。类体中是类成员表。右括号后加分号“;”，作为类定义的结束标记。一个类就是由用户定义的一个新增类型，这个类型并不“抽象”，我们对它的使用就像使用 int 和 float 一样简单。在上面的定义方法中，“//”之后的文字是注释，对程序的执行不起作用，但可以帮助理解程序。

【例 1-1】 定义一个学生类的方法。

```
class Student                                //定义学生类
{private:                                    //私有成员
    long num;
    char name[20];
    int score;
 public:                                     //公有成员
    void Study(int x)                        //成员函数
    {score=x;}                               //函数体，赋值统计
};
```

在上面的描述中，类名 Student 代表的是一个新增数据类型，可以通过它来定义对象。对象的定义方法与我们用内部类型定义变量的方法一样。例如：

```
void main()
{Student Jonas; }                    //定义了一个 Student 类的对象 Jonas
```

引入类的概念之后，我们可以把学生信息重新描述为：定义了一个学生类 Student，并在主函数中定义了一个该类的对象 Jonas。

1.3.2　属性与行为

类的描述主要由属性和行为两部分组成。属性用于描写个性，由数据成员来完成。行为用于实现某一功能，由成员函数来完成。

在 C++中定义类时并不分配内存单元，只有定义了类的对象之后才分配内存单元。

在例 1-1 中定义了一个类 Student，并通过该类定义了一个对象 Jonas，这时就在内存中开辟了一片空间，如图 1-2 所示。有了内存空间，就可以对该空间的数据进行存取等操作了。

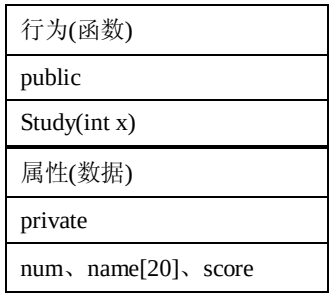


图 1-2　属性和行为

用 private 和 public 可将类的成员分为私有和公有两个区。private 区的成员具有私有性，类的外部不能访问它们；public 区的成员具有公有性，类的外部可以对它们进行访问。我们将 private、public 等称为访问控制说明。

类中的所有成员只能是成员函数或数据成员。类的行为由成员函数来实现，类的特征由数据成员来表现。在上面的类定义中，Student 类有学号、姓名和成绩三个特征，有学习课程的行为，并通过学习累积学分。

在主函数中定义了一个 Student 类的对象 Jonas。整个学生是一个群体，是一个学生类，而 Jonas 是学生中的一个对象，是学生中的一员。Jonas 有统计学分的行为，该行为是公有的，同时他还有学号、姓名、成绩等私有属性。

1.3.3　封装与隐藏

1．类的封装

类的封装就是标识成员的访问控制说明，即指定某成员具有 public(公有的)、private(私有的)或 protected(保护的)性质。类的这一要求称为类的封装性。在完成了封装后，对于说明为 public 的成员，类的外部可以对它们进行访问和操作；对于私有或保护成员，类的外部不能对它们进行访问，只有该类的成员函数才有权对它们进行访问。对于不指定访问控制说明的成员，默认其封装性为 private。

2．封装的作用

类有许多特性和方法，用户不需要全部了解，也能很好地实现其功能，这就如我们并不需要了解电视机的内部结构，只需通过一些按钮便可使它发挥作用一样。生产厂将内部实现技术与外部接口自成于一体，这种自成一体性，称为封装。经过封装，用户通过接口使类发挥作用，既保护了内部元器件，又有利于技术保密。因此，封装的作用就是有选择地隐藏类中的特性和方法。

通过封装有选择地隐藏类中的特性和方法，将类的一些数据成员声明为私有以及通过公有成员函数对它们进行访问，有助于保护它们的完整性。

C++定义的类具有封装性。正确的类定义一旦建立，就可以将类看做是完成封装的实体，可以作为一个整体单元被使用。通过公有接口也就是成员函数，对类属性实现有效的和有限的访问，既使类发挥了功效，又使类得到了可靠的保护，这就是封装的作用。

在 C++中，成员函数的行为是由程序员规定好的，只能完成预定的功能，其他想通过成员函数来对被封装的数据成员进行破坏的行为都不被理睬。这一封装特性正是 C++符合软件工程理论的又一精妙之处。

3．封装的目的

(1) 相对于外部函数(普通函数或其他类的成员函数)而言，使类内部的保护和私有成员不被肆意更改。

(2) 只有类的成员函数才能访问保护或私有成员，使类具有自身的维护功能。

(3) 限制类的外部接口，把类分成两部分，一部分是公有的，一部分是保护的。

(4) 减少类与其他代码的关联程度。

可以类比一下我们家中的电脑，它由接口、键盘和主机箱等组成。接口和键盘是 public 的，允许我们使用，而主机箱内是 protected 或 private 的，除维修人员以外是不能打开的。这样做的好处是主机箱内部的元器件可以不被肆意拆卸，减少了许多麻烦。通过接口、键盘和鼠标等，我们可以正确地使用电脑的功能。而接口、键盘和鼠标的功能是有限的，对它们的使用一般不会造成主机箱内元件的损坏。

1.4 现实生活中的类与对象

通常我们所说的对象就是现实世界中的实体。比如我们每一个人就是一个对象。人有大脑、眼睛、鼻子、耳朵和手等。人还会有一些行为，如行走、工作和思考等。除了具有上面列举的共同特性之外，每一个人还有自己的姓名、年龄和身高等。

把具有相同特性的对象进行抽象，归并在一起，就可以用“类”的概念来描述。比如在描述人时，先要描述人类。首先说明人类是什么，然后说一个具体的人就是其中的一员，即一个对象。这种方法用程序设计的语言来描述就是先定义类，然后由类来定义对象。

1.4.1 类的描述

C++使得我们能从真实客观的角度来理解、开发应用程序，这就要求程序设计应当是现实世界的体现。

比如描述汽车类，首先区分哪些是汽车的行为，哪些是它的特征。汽车的型号、生产厂和发动机等是特征，汽车的载客、行驶和制动等是行为。因为我们不希望汽车用户改变型号、生产厂等，所以可将它们作为汽车类的私有数据成员。汽车的价值是供人使用，因此要将汽车的载客、行驶等作为汽车类的公有成员函数。类的外部可以对这些公有成员函数进行访问，以便能表现它的行为。

1.4.2　类与对象

现实世界中的对象不仅有共性也有个性。我们如何将现实世界中的对象转化为程序中的对象呢？即对于现实世界中的对象，如何在程序中表现出来呢？在本节中就要解决这个问题。程序中的对象也就是编程对象，它就是将对象的特性与行为组合在一起得到的结果。

1．编程中的类

程序设计的目的就是要解决实际问题。下面描述现实生活中的对象——人，并将他转换为编程对象。

现实生活中的人

类名

行为：

　　行走

　　工作

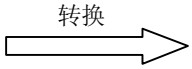
　　思考

特征：

　　身高

　　四肢

　　体重



程序中的人(即编程中的类)

```
class human
{public:
      void Run();
      void Work();
      void Think();
 private:
      int   length;
      int   limb;
      float weight;
};
```

类是一个抽象的概念，对象才是一个客观的实体。例如，人类就是一个类，它可以有具体实物，就是人，这个人就是人类的对象。人有身高体重，并占有一定的空间。

2．编程中的对象

现实生活中的对象是生产、制造或生养出来的，程序中的对象要由类来定义。在程序中要定义对象，先要定义类，再通过类定义对象。类对所定义对象的数量并没有限制。

现在用程序设计语言来描述一个学生对象。

【例 1-2】 说明定义对象的方法。

```
#include<iostream.h>
class Student                          //定义学生类
{private:                              //私有成员
      long num;                        //学号
      char name[20];                   //姓名
      int score;                       //成绩
 public:                               //公有成员
      int no;
      void Study(int x)                //获得学分
      {score=x;}
      void Display()                   //输出学分
      {cout<<"score="<<score<<endl;}
```