高 职 高 专 计 算 机 系 列 教 材

C++程序设计

白伟青 主编 刘春秋 韩亮 宾晟 张昊 编著

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

高职高专计算机系列教材

C++程序设计

白伟青　主编

刘春秋　韩亮　宾晟　张昊　编著

中国铁道出版社
CHINA RAILWAY PUBLISHING HOUSE

内 容 简 介

本书详细介绍了C++语言和面向对象的程序设计。全书共分10章，分别介绍了基本的C++语言语法、C++中的类和对象、运算符重载、继承、虚函数和多态性、C++输入/输出流，以及程序调试方法等。每章后都附有适量的习题，便于读者巩固已学的知识。

本书是针对没有学过C语言的读者而编写的入门级教材，内容深入浅出，循序渐进，便于自学。可作为计算机及相关专业面向对象程序设计课程的教材，也可作为广大计算机爱好者的自学参考书。

图书在版编目（CIP）数据

C++程序设计/白伟青主编；刘春秋，韩亮，宾晟编著. —北京：中国铁道出版社，2006.1
（高职高专计算机系列教材）
ISBN 7-113-06930-4

Ⅰ.C… Ⅱ.①白… ②刘…③韩…④宾…
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（2006）第007149号

书　　名：C++程序设计
作　　者：白伟青　刘春秋　韩　亮　宾　晟　张　昊
出版发行：中国铁道出版社（100054，北京市宣武区右安门西街8号）
策划编辑：严晓舟　杨东晓
责任编辑：严　力　李晶璞　李　旸
封面制作：白　雪
印　　刷：河北省遵化市胶印厂
开　　本：787×1092　1/16　印张：16　字数：383千
版　　本：2006年2月第1版　2006年2月第1次印刷
印　　数：1～5 000册
书　　号：ISBN 7-113-06930-4/TP·1741
定　　价：22.00元

前　言

随着面向对象程序设计方法的不断普及和应用，学习和掌握C++语言已成为许多计算机专业工作者和广大计算机应用人员的迫切需要。学好C++，可以很容易地触类旁通其他语言，如Java和C#等。C++架起了通向强大、易用、真正的软件开发应用的桥梁。

全书共分10章，另外还有附录部分，第1章讲述了面向对象程序设计的基本概念、C++程序的结构、基于Visual C++ 6.0集成开发环境的C++程序的编写、编译、连接和运行步骤。第2章～第5章介绍基本的C++语言语法，包括基本数据类型和构造数据类型、函数、运算符、表达式以及顺序、选择和循环3种基本程序结构等，是C++程序设计语言的基础。第6章～第10章围绕面向对象程序设计的封装性、继承性和多态性3个基本特性，讲述类与对象、构造函数和析构函数、静态成员、友元、继承、运算符重载、虚函数与多态性以及输入/输出流等内容。最后的附录给出了程序调试的基本方法。

本书的结构是：首先介绍给读者一个面向对象的C++程序，通过讲解这个程序的结构，让读者先对面向对象的程序结构有一个整体的初步认识。然后以类的数据成员和成员函数为线索展开，介绍定义数据成员时所涉及到的数据类型及怎样定义成员函数等相关知识，最后介绍在C++中怎样实现面向对象的三大特性：封装性、继承性和多态性。本书直接介绍面向对象的程序设计，并贯穿始终，力求让读者尽快地建立起面向对象编程的思想。使读者不仅学会一门程序设计语言，还能初步掌握面向对象的程序设计方法。

在本书的编写中，编者结合自己的教学和编程实践经验，力图用生动，通俗易懂的语言并结合编程实例来讲解各个知识点，便于读者理解和掌握。每章后都附有适量的习题，读者可通过习题巩固已学的知识。书中所有实例已由编者在Visual C++ 6.0环境下调试通过。

本书是针对没有学过C语言的读者而编写的入门级教材，内容深入浅出，循序渐进，便于自学。可作为计算机及相关专业面向对象程序设计课程的教材，也可作为广大计算机爱好者的自学参考书。

本书由白伟青主编，第1、3、4、6章由白伟青编写，第2章由张昊编写，第5章由刘春秋编写，第7、8章由宾晟编写，第9、10章及附录由韩亮编写，参加编写的还有魏爱敏、杨金龙。全书由白伟青和刘春秋统稿并审校。

由于编者水平有限，加之时间仓促，书中难免有疏漏之处，敬请广大读者批评指正，以使本书质量得到进一步提高。

编　者

2005年11月

目　录

第 1 章　C++语言概述

1-1　C++的起源

由于 C 语言能在绝大多数系统程序设计任务中替代了汇编语言，且具有高效、灵活和良好的可移植性，所以已经成为广泛应用的程序设计语言之一。随着 C 语言的应用，它的缺点也逐渐显露出来，主要有以下几个：

（1）C 语言的类型检查机制较弱，使得程序开发过程中的一些错误不能在编译过程中被发现。

（2）C 语言本身是面向过程的语言，没有支持代码重用的机制，使得一个程序员编制的程序很难再被其他程序使用。

（3）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。

1980 年美国 AT&T 贝尔实验室的 Bjarne Stroustrup 对 C 语言进行了扩充，把 Simular 67 中类的概念引入到 C 中，当时称为“带类的 C”。1983 年，由 Rick Maseitti 提议正式命名为“C++”，并于同年 7 月首次向外界发表。在经历了 3 次修订后，于 1994 年制定了 ANSI C++标准的草案。以后又经过不断完善，成为目前的 C++。

C++语言改进了 C 的不足之处，增加了对面向对象的程序设计的支持。在改进的同时，保持了 C 的简洁性和高效性。在解决一些小型的简单程序设计问题时，C 是非常有效的，而 C++是解决大型的复杂问题时所选择的语言。开发一个大型项目时，使用面向对象的工具减少了程序的开发时间，并且可以提高程序的可维护性。

1-2　面向对象的基本概念

1-2-1　对象

从一般意义上讲，对象是现实世界中一个实际存在的事物，它可以是有形的（如一辆汽车），也可以是无形的（如一项计划）。对象是构成世界的一个独立单位。在面向对象程序设计中，对象是客观世界中事物在计算机领域中的抽象，是一组数据和施加于该组数据上的一组操作（行为）组成的集合体。

对象是面向对象语言中构成整个程序的主要成员。对象所具备的特征是：

（1）对象必须有一个名字以区别于其他对象。

（2）用属性描述对象的特征（状态）。

（3）用一组操作描述对象的行为。

例如，张三这个对象，对象的属性为：

身高：175 cm

年龄：20

性别：男
国籍：中国
对象的操作可能是：
回答身高
回答年龄
回答性别
回答国籍

C++语言中的对象由描述对象状态的数据结构和作用于这个数据结构上的方法（或称为操作）构成，它们都可以分为私有的和公有的两个部分，私有部分不可从对象的外部直接访问，而公有部分可以从对象的外部直接访问。C++语言中对象之间的相互联系和作用通过对公有数据和方法（操作）的访问来实现。

1-2-2 类

类是对一组具有相同特征的对象的抽象描述，所有这些对象都是这个类的实例。例如，学生是一个类，而一个具体的学生则是学生类的一个实例（对象）。类和对象之间的关系是抽象和具体的关系，类是对多个相似对象进行综合抽象（抽取共同点）的结果，对象又是类的一个具体实例。在程序设计语言中，类是一种数据类型，而对象是该类型的变量，变量名即是某个具体对象的标识。

一个类的定义至少包含以下两个方面的描述。

（1）该类所有实例的属性定义或结构定义。

（2）该类所有实例的操作（或行为）的定义。

在C++语言中，一个类的定义包含数据成员和成员函数两部分内容。数据成员定义该类对象的属性，不同的对象属性的值可以不同；成员函数定义了该类对象的操作。

下面是C++中一个类及其对象的定义的例子。

```
//定义一个圆类
class Circle
{
  private:
    float Radius;                    //描述半径的数据成员
  public:
    Circle(float radius)             //构造函数
    {
      Radius=radius;
    }
    ~Circle() {  };                  //析构函数
    float GetArea()                  //计算面积的成员函数
    {
      return 3.14*Radius*Radius;
    }
};
//定义一个具体的圆对象
Circle  MyCircle(2.0);
```

在面向对象程序设计中，类的确定与划分是非常重要的，是软件开发的基础，划分的好坏直接影响软件开发的后续工作和软件产品的质量。类的确定没有统一的标准和固定的

方法，基本上依赖设计人员的经验、技巧以及对实际问题的理解程度。所以，对于同一个问题，不同的人有不同的分类方法。

1-2-3　消息

软件系统的对象之间也存在着依存关系，一个对象除了通过对外提供服务来发挥自己的作用外，还需要请求其他对象为它服务。当一个对象 A 需要对象 B 为它服务时，它就发送一个消息给对象 B，对象 B 接收到这个消息后，就启动相应的服务程序来响应这个消息。

消息是向对象发出的服务请求，是面向对象系统中对象之间交互的途径。对象之间通过消息联系，彼此共同协作，才能形成一个有机的系统。消息有几个关键要素：消息的发送者、消息的接收者、消息所要求的具体服务、消息所要求服务的一些参数以及消息的应答。

在具体的程序设计语言中，消息表现为对象在其操作过程中对另一个对象的服务程序的调用，也就是函数的调用。调用时，需要给出被调用对象、被调用的函数和适当的函数参数。函数的执行过程就是对消息的响应过程，函数的返回值就是对消息的应答。

以上面给出的圆对象为例，给该对象发送一个消息：求其圆面积。发送的方式为：

```
MyCircle.GetArea();
```

这里 MyCircle 是消息的接收者，也是服务的提供者。消息的名称为 GetArea，消息的响应过程就是求圆的面积的过程，消息的应答就是返回的圆面积值。

对于纯粹的面向对象的程序设计语言，所有的函数调用都可以看成是一种消息，所有的消息发送者都是对象，但是，有些非纯粹的面向对象语言中，还保留了结构化程序设计的语言成分，并不是所有函数都是某一对象提供的服务。因而不是所有的函数调用都是消息，也不是所有的消息发送者都是对象，比如 C++中的 main()函数。

1-2-4　封装

从字面上理解，封装就是把某个事物包起来，使外界不知道该事物的具体内容。

在面向对象的程序设计中，把数据和实现操作的代码集中起来放在对象内部。一个对象好像是一个不透明的黑盒子，对象中的一些成员被封装在黑盒子里面，从外面是看不见的，更不能从外面直接访问或修改这些数据及代码。它们被有效地屏蔽，以防外界的干扰和误操作。另一些成员是公共的，它们作为接口提供给外界使用。

图 1-1 是一个圆类的描述。

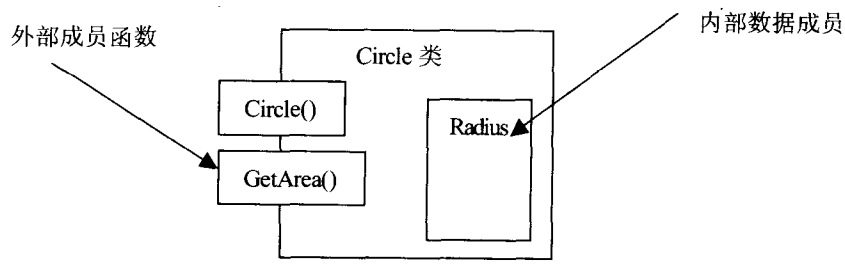

图 1-1　一个圆类的描述

使用一个对象时，使用者只需知道它向外界提供的接口形式就可以使用它，而不需要知道它的数据结构细节和实现操作的算法。用户向对象发送消息，对象根据收到的消息，调用内部方法做出响应。

封装的目的在于将对象的使用者和对象的设计者分开，使用者无须知道对象内部实现的细节，只需要知道对象接收的消息即可。

封装性也就是信息隐藏，把对象的实现细节对外界隐藏起来了。

C++通过建立类来支持封装性。类一旦被建立，就可看成是完全封装的实体，可以作为一个整体单元被使用。类的内部工作被隐藏起来，用户使用类时不需要知道类是如何工作的，只要知道如何使用它就行了。

1-2-5 继承

在现实生活中，继承是非常普遍和容易理解的。例如 PC 是一种计算机，它具有计算机的基本功能，如能计算、存储数据等，但它又有自己的特点，如体积比较小。

在面向对象的程序设计中，继承表现为可以在已经定义的类的基础上生成新的类。

在 C++语言中，通过类的派生机制来实现类的继承，可以从一个类中派生出一个新的类，这个类称为派生类的基类或父类，派生出的新类称为基类的派生类或子类。派生类的对象具有基类对象的特征，同时又有其自身特有的特征。例如，一个教师类的对象与一个学生类的对象都具有类人所描述的特征，同时又具有各自所属教师类和学生类特有的特征。

利用类之间的继承关系，可以简化类的描述。在基类中描述共性，在派生类中描述各自的个性。利用继承机制可以提高软件代码的可重用性。在设计一个新类时，不必从头设计编写全部的代码，可以通过从已有的具有类似特性的类中派生出一个类，继承原有类中的部分特性，再加上所需的新的特性。

1-2-6 多态性

简单地讲，多态性是指不同的对象收到相同的消息产生不同的行为。例如，一个经理要到外地开会，当他把这个消息告诉不同的人：他的夫人、秘书、下属，这些人听到这个消息后，会有不同的反应：夫人会为他准备行装，秘书会为他安排机票和住宿，下属会为他准备需要的材料。

C++语言支持两种多态性：编译时的多态性和运行时的多态性。编译时的多态性是通过重载实现的，重载的函数在编译时就可以决定选用哪个函数；而运行时的多态性是通过虚函数实现的，需要借助抽象类和动态联编机制。动态联编是系统在运行时动态判断对象类型，根据对象实际的类型来动态决定调用哪个对象的函数的机制。C++的虚函数的机制就提供了动态联编的能力。

1-3 一个面向对象的 C++程序

1-3-1 C++程序的基本结构

C 语言是支持面向过程的程序设计语言，而 C++则是既支持面向过程的程序设计又支持面向对象的程序设计的语言。学习 C++程序设计有两种途径：一种是先学习面向过程的程序设计方法，然后逐渐过渡到面向对象的程序设计；另一种是直接学习面向对象的程序设计。本书所采用的方法是直接学习面向对象的程序设计。所以，本节将给出一个面向对象的 C++程序，目的是使读者对面向对象的 C++程序有一个初步的整体概念。

【例 1.1】 声明一个矩形类，有长、宽两个属性，由成员函数计算矩形的面积。

```
#include<iostream.h>                          //包含 iostream.h 文件
class Rectangle                               /*定义 Rectangle 类*/
{
//数据成员
private:
  float Length;                               //矩形的长
  float Width;                                //矩形的宽
//成员函数
public:
  Rectangle(float len, float width)           //构造函数
  {
    Length=len;
    Width=width;
  }
  ~ Rectangle(){  };                          //析构函数
  float GetArea()                             //求矩形面积函数
    {
      return Length*Width;
    }
};
void SayGoodBye();
void main()
{
  float length, width;
  cout<<"请输入矩形的长度: ";
  cin>>length;
  cout<<"请输入矩形的宽度: ";
  cin>>width;
  Rectangle  r(length, width);
  cout<<"长为"<<length<<"宽为"<<width<<"的矩形的面积为: "<<r.GetArea()<<endl;
  SayGoodBye();
}
void SayGoodBye()
{
  cout<<"Bye! "<<endl;
}
```

运行结果为：

```
请输入矩形的长度: 5
请输入矩形的宽度: 6
长为 5 宽为 6 的矩形的面积为: 30
Bye!
```

程序中定义了一个 Rectangle 类和两个函数，其中一个名为 SayGoodBye 的函数，另一个名为 main 的函数，main()函数是程序的主函数。C++程序的执行总是从 main()函数开始。

```
class Rectangle
{
  //以下说明成员函数
      …
  //以下说明数据成员
      …
};
```

定义了 Rectangle 类，其中 class 是关键字，说明现在要定义一个类。class 之后写出类的名字，在类名后面的一对花括号“{”和“}”之间，说明类的属性和操作。C++语言把类的属性称为数据成员，把类的操作称为成员函数。最后用分号“;”结束对一个类的定义。从双斜线“//”开始至该行的结束为止的内容是注释，说明要在该处定义成员函数或数据成员。有关类的定义的详细内容将在后面的章节介绍。

一个 C++程序有且仅有一个 main()函数，main()函数可以放在程序的任何位置，但 main()函数的名称是不能更改的。因为编译器在编译程序时，会先从 main()函数的位置开始编译，如果没有这个函数，就无法完成编译的工作。

在 main()函数定义中的 void 表示该函数不返回任何值；放在一对花括号“{”和“}”之间的部分称为函数体，函数体可以为空，也可以由一系列的 C++语句构成：一组说明数据对象的语句，实现运算的表达式和赋值语句，程序结构控制语句，以及输入/输出语句。每条语句以“;”结束，在主函数中通过消息传递完成对象之间的通信。也可以说，一个 C++程序就是一组相互通信的对象。

本程序的 main()函数中，第 1 行声明了两个浮点型变量 length 和 width；第 2 行在屏幕上输出提示信息：“请输入矩形的长度：”；第 3 行输入矩形的长给变量 length；第 4 行在屏幕上输出提示信息：“请输入矩形的宽度：”；第 5 行输入矩形的宽给变量 width；第 6 行定义一个矩形类的对象 r；第 7 行通过调用矩形对象 r 的成员函数 GetArea()向矩形 r 发出计算其面积的消息，并将其计算结果显示出来；第 8 行调用 SayGoodBye()函数，在屏幕上显示“Bye!”。

在 C++中执行输入/输出操作时，通常使用两个名为 cin 和 cout 的标准对象。这两个对象是在 iostream 头文件中定义的，所以为了在程序中使用 cin 和 cout 进行输入/输出，要在程序的开头加上这样一条语句：#include <iostream.h>。该语句以“#”开头，是一条编译预处理命令，它的作用是在编译之前将文件“iostream.h”的内容插入（包含）到本程序中，作为程序中的一部分。iostream.h 是系统定义的一个头文件，它设置了 C++的 I/O 相关环境，定义输入/输出流对象 cout 与 cin 等。cin 是标准输入设备，通常指键盘；cout 是标准输出设备，通常指显示器。

C++使用下列两个运算符执行 I/O 操作：

流插入运算符“<<”，常与 cout 连用；

流析取运算符“>>”，常与 cin 连用。

流运算符“<<”和“>>”的定义在 iostream.h 文件中。流插入运算符“<<”把数据插入到 cout 对象中，然后再送到屏幕上进行显示。例如本程序的 main()函数中的第 2 行语句，指示计算机将“<<”右侧的字符串输出到屏幕上显示。

流析取运算符“>>”从 cin 对象中析取数据，并把它存储到操作符“>>”右侧的对象的内存地址中。例如 main()函数中的第 3 行语句，指示计算机从 cin 对象（通常是指用来输入数据的键盘）析取一个值并把该值存储到保存变量 length 的内存空间中。

一个 cout<<语句可以用来输出不止一个值，例如 main()函数中的第 7 行语句。它按下列顺序将数据输出到屏幕上：首先是字符串“长为”和变量 length 的值；然后是字符串“宽为”和变量 width 的值；最后是字符串“的矩形的面积为：”和对象 r 的成员函数 GetArea()的返回值。endl 是 C++在 iostream.h 文件中定义的控制符（Manipulator），它代表回车，因此本行信息输出后会自动换行。

一个 cin>>语句也可以用来输入不止一个值，例如可以把变量 length 和 width 的值的输

入改为用下面的语句来实现：

```
cout<<"请输入矩形的长度和宽度: ";
cin>>length>>width;
```

在输入两个值之间，用户应该输入一个空格，或按【Enter】键。输入的两个值分别被放至变量 length 和 width 中。

程序的注释是程序的编写过程中非常重要的工作。如果没有程序注释，就很难读懂别人编写的程序，即使是程序设计者本人日后也很难理解程序的设计流程。C++的程序注释方式如下：

```
// 注释
```

或

```
/* 注释 */
```

从两个左斜线“//”后的文字开始直到该行的结束，都被编译器视为注释。

在“/*”和“*/”之间的文字，也是程序的注释。在“/*”之前和“*/”之后的文字则会被编译器当成程序代码执行。

注释多用于以下几种情况。

（1）程序头。

在每个程序的开头写上注释，提供该程序的有关信息，如程序的名字，程序功能的简短描述，作者的名字，以及编写程序的目的、用途等。

（2）函数的前面加上注释，提供有关函数的信息。

函数的注释信息一般包含以下内容：函数的功能；函数的参数，说明函数各个参数的含义、参数允许的取值范围等；函数的返回值的含义；函数的副作用：函数运行产生的影响，如改变了系统的状态、创建了一个文件，或者申请了一些内存空间等。

（3）说明所定义的变量的含义。

（4）在函数体中，用注释解释算法的思路。

（5）为类加上注释，用注释将成员分类。

为类写注释说明类提供的功能、使用方式（如何构造对象，如何改变类的状态）、使用限制等。一个稍微复杂的类会有很多的成员，如果不对这些成员做适当的区分，会使类定义显得杂乱无章。划分类成员的方式是多种多样的，而且不同的类其划分的方式也会有所不同。首先应该将类的接口与实现分开，然后将接口根据其功能进行分类。

（6）其他需要说明的地方。

通过上面的例子可以看出，C++源程序的基本组成部分有：

（1）预处理命令。如文件包含命令，放在程序开头。

（2）类的定义。在程序开头定义一个或多个类，在类体内包含数据成员和成员函数。

（3）函数。C++源程序中除了 main()函数外，在类体内可定义成员函数，如 GetArea()函数，还可在类体外定义一般函数，如 SayGoodBye()函数。在 main()函数中定义类的对象，通过向对象发送消息来实现程序中的一切操作。对象接收到消息后，调用有关对象的成员函数完成相应的操作。

（4）注释。通过使用注释可以增加程序的可读性。

类和函数是 C++源程序的主体。

1-3-2 C++程序的开发过程

程序员编写的C++程序叫源程序，以.cpp为扩展名，不能直接被计算机执行。源程序被编译器编译后会生成一个以.obj为扩展名的目标文件，该文件是源程序的目标代码，即机器语言指令。但现在仍不能被计算机执行，因为目标代码只是一个个的程序块，需要进行相互连接，生成一个完整的程序。

C++可执行程序通常是通过同时连接一个或几个目标文件与一个或多个库而生成的。库（.lib）是一组由机器指令构成的程序代码，是可连接文件。库分为标准库和用户生成的库，标准库是由C++提供的，用户生成的库是由软件开发商或程序员提供的。

在编译过程中，编译器将会检查程序的语法是否符合规定。如果编译器检测到错误，会显示程序的错误信息。这时程序员需要对源程序进行修改，然后再重新编译，直到编译时不再提示错误为止。同样，在连接和运行过程中也会遇到错误，这时程序员也需要对源程序进行修改，然后再重新编译、连接、运行，直到连接、运行时不再提示错误为止。整个开发过程如图1-2所示。

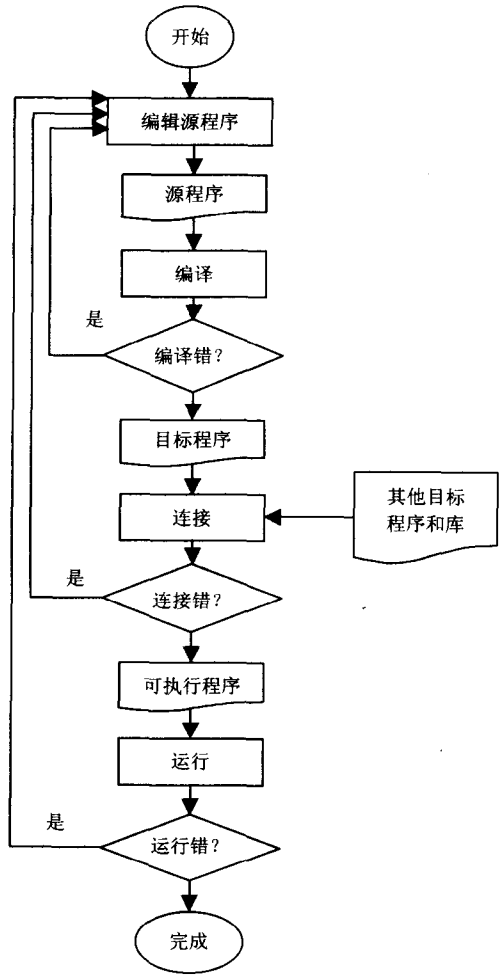

图1-2 C++程序开发过程

1-4　Visual C++ 6.0 集成开发环境

Visual C++ 6.0 提供了良好的可视化编程环境，集项目建立、打开、浏览、编辑、保存、编译、连接和调试等功能于一体。Visual C++ 6.0 可运行于 Windows 95/98/2000 及 Windows NT 环境。将 Visual C++ 6.0 正确安装到 Windows 系统之后，选择“开始\程序\Microsoft Visual Studio 6.0\Microsoft Visual C++ 6.0”命令，即可启动，进入集成开发环境（Developer Studio），如图 1-3 所示。

图 1-3　Visual C++ 6.0 集成开发环境

集成开发环境的主窗口包括标题栏、菜单栏、工具栏、项目工作区、正文窗口、输出窗口和状态栏。标题栏用于显示应用程序名和打开的文件名；菜单栏完成 Developer Studio 中的所有功能；工具栏对应于某些菜单或命令的功能，简化用户操作；项目工作区（Workspace）窗口用于组织文件、项目和项目配置；正文窗口用于各种程序文件、资源文件、文档文件等显示或编辑；输出窗口用于显示项目建立过程中所产生的各种信息；状态栏给出当前操作或所选择的命令的提示信息。

1-4-1　项目工作区

Developer Studio 使用项目工作区来组织文件、项目和项目配置。在一个项目工作区中，可以处理：

- 一个项目和它包含的文件
- 一个项目的子项目
- 多个相互独立的项目
- 多个相互依赖的项目