第 1 章　面向对象程序设计概述

本章简要介绍编程语言的发展、面向对象程序设计语言的起源与特点和面向对象方法的由来及其基本概念，然后介绍面向对象的软件开发，最后介绍 C++程序在 VC 7.0 中的编译运行。

1.1　程序设计语言的发展

1.1.1　编程语言的发展

自从 1946 年 2 月世界上第一台数字电子计算机 ENIAC 诞生以来，在这短短的 50 多年间，计算机科学迅猛发展，计算机及其应用已渗透到社会的各个领域，有力地推动了整个信息化社会的发展，计算机已成为信息化社会中必不可少的工具。

计算机系统包括硬件系统和软件系统。计算机之所以有如此强大的功能，不仅因为它具有强大的硬件系统，而且也依赖于软件系统。软件包括了使计算机运行所需的各种程序及有关的文档资料。计算机的工作是用程序来控制的，离开了程序，计算机将一事无成。

机器语言是一种最原始的编程语言，这种语言是计算机可以直接识别的语言。这种语言使用 0 和 1 两种代码，编写出的程序难以理解和记忆，因为它远不同于人们的习惯思维方式。

汇编语言是一种使用助记符号来替代代码 0 和 1，是一种低级语言，它比机器语言稍有提高，符合人们的形象思维，它是低层次的抽像，对于汇编语言，计算机不能直接识别，需要编译后才可识别。这种语言虽然效率较高，但是由于难以记忆，使用较少。

高级语言的出现是计算机编程语言的一大进步。它屏蔽了机器的细节，提高了语言的抽像层次，程序中可以采用具有一定含义的数据命名和容易理解的执行语句。这使得在书写程序时可以联系到程序所描述的具体事物。

20 世纪 60 年代末开始出现的结构化编程语言进一步提高了语言的层次。结构化数据、结构化语句、数据抽像、过程抽像等概念，使程序更便于体现客观事物的结构和逻辑含义。这使得编程语言与人类的自然语言更接近。但是二者之间仍有不少差距。主要问题是程序中的数据和操作分离，不能够有效地组成与自然界中的具体事物紧密对应的程序成分。

目前应用比较广泛的几种高级语言有 FORTRAN、BASIC、PASCAL、C 等。当然本书介绍的 C++语言也是高级语言，但它与其他面向过程的高级语言有着根本的区别。

面向对象的编程语言与以往各种编程语言的根本不同点在于，它设计的出发点就是为了能更直接地描述客观世界中存在的事物（即对象）及其之间的关系。

开发了一个软件是为了解决某些问题，这些问题所涉及的业务范围称为该软件的问题域。面向对象的编程语言将客观事物看作具有属性和行为的对象，通过抽像找出同一类对象的共同属性（静态特征）和行为（动态特征），形成类。通过类的继承与多态可以很方便地实现代码重用，大大缩短了软件开发周期，并使得软件风格统一。因此，面向对象的编程语言使程序能够比较直接地反映问题域的本来面目，软件开发人员能够利用人类认识事物所采用的一般思维方法来进行软件开发。

面向对象的程序设计语言经历了一个很长的发展阶段。例如， LISP 家族的面向对象语言， Simula 67 语言， Smalltalk语言，以及 CLU、Ada、Modula－2等语言，或多或少地都引入了面向对象的概念，其中 Smalltalk 语言是第一个真正的面向对象的程序语言。

然而，应用最广的面向对象程序语言是在 C语言基础上扩充出来的 C++语言。由于 C++对 C 兼容，而 C 语言又早已被广大程序员所熟知，所以， C++语言也就理所当然地成为应用最广的面向对象程序语言。

1.1.2 C++语言的起源与特点

C++语言是在 C 语言的基础上为支持面向对象的程序设计而研制的一个通用目的的程序设计语言，它的开发者是 AT&T 贝尔实验室的 Bjarne Stroustrup 博士。

C 语言是 1972 年由 Dennis Richie 在 AT&T 贝尔实验室设计的一个通用目的的程序设计语言，它的前身是 B 语言，而 B 语言又是在继承和发展了 BCPL 语言的基础上设计的。C 最初用作 UNIX 操作系统的记述语言，由于 UNIX 的成功和广泛使用，也使 C 成为一种普遍使用的程序设计语言，目前在各种机型和各种操作系统上都运行有 C 编译器。C 有广泛的应用基础，有众多的程序员在使用 C 语言 ,并且 ,有许多 C 语言的库代码和开发环境。

研制 C++的一个首要目标是使 C++首先是一个更好的 C,所以，C++要根除 C 中的问题。同时，在 C++中引入了类等机制来支持面向对象的程序设计。所研制的这个语言最初被称为“带类的 C”, 1983 年取名为 C++，以后又经过不断完善和发展，成为目前的 C++. 除了个别的例外情况，它将 C 作为它的子集，并引入了其他语言中的一些概念和机制。它确实达到了使 C++成为更好的 C 的目标，并且还保持了 C 的简洁、高效和接近汇编语言的特点。C++对 C 的类型系统进行了改进和扩充，因此，C++比 C 更安全，C++编译器能检查出更多的类型错误。研制 C++ 的另一个主要目标是支持面向对象的程序设计。面向对象的程序设计是一种更好的程序设计技术，它通过使程序员在程序中能够定义更多更强有力的类型，使 C++语言对更高级的抽像有更好的支持，这种支持使在计算机上解决问题的方式更加类似于人类的活动。

C++保持与 C 兼容，这就使许多 C代码不经修改就可以为 C++所用，用 C 编写的众多的库函数和实用软件可以用于 C++中；更重要的是，C 程序员仅需学习 C++语言的新特征，就可以很快地用 C++编写程序。另外，C++编写的程序可读性更好，代码结构更为合理，可以更直接地在程序中映射问题空间的结构。

C++对 C 的兼容性对 C++的快速普及是功劳卓著的，C++极大地促进了对面向对象技术的广泛应用。在 20 世纪 80 年代出现了许多面向对象的语言和工具，但 C++是比较高效

的，并且有广泛的程序员使用基础。

但需要指出的是，C++对 C 的兼容使 C++不是一个纯正的面向对象的语言。因为 C 语言不是面向对象的语言，而是面向过程的语言，所以， C++也必须支持 C 的面向过程的程序设计。由于两种不同风格的程序设计技术融于同一个语言中，使初学 C++的程序员感到有些困难。C++的初学者不仅需要记忆一些本不需要记忆的东西，还需要掌握用面向对象的方法去构造程序。

所以，在学习 C++时，我们必须首先明确 C++是一门完整的语言，它提供特定的机制去支持特定的程序设计技术——面向对象的程序设计。保持与 C 的兼容是为了保护在 C 上已做的大量的投资，同时，使 C 程序员在转向使用新技术时，对于新语言的语法和语言中的一些机制的学习不致于感到太困难。在新的技术——面向对象的程序设计面前，C 程序员也是新手。从 C++面向对象的程序设计需要的角度去考察 C++中的 C 成份，我们会发现，C 中的大部分成份在 C++中被降到次要的地位，其原因在于，面向对象的程序设计从一个不同于面向过程的程序设计的角度去观察程序和构造程序。虽然与 C 兼容部分不是 C++的主要部分，但仍然不能超越它，像数据类型、算法的控制结构、函数等，不仅是面向过程程序设计的基本成分，也是面向对象编程的基础。

1.1.3 C++源程序的构成

C++是 C 的一个超集，它几乎保留了 C 的所有的特性，下面给出一个简单的 C++程序，以便读者对 C++程序的格式有一个初步的了解。

【例 1.1】 输出一行字符。

```
#include <stdio.h>
#include<iostream.h>
/*本程序的作用是输出一行字符*/
void main( )
 {
printf("This is a C++ program.\n");
cout<<"This is a C++ program.\n";/   /本行输出一行字符
}
```

程序运行结果：

```
This is a C++ program.
This is a C++ program.
```

分析：

（1）在 C++程序中一般习惯在主函数 main 前面加了一个类型声明符 void，表示 main 函数没有返回值。

（2）除了可以用/* */形式的注释行外，还允许使用以//开头的注释。从程序最后一行中可以看到：以//开头的注释可以不单独占一行，它出现在语句之后。编译系统将//以后到本行末尾的所有字符都作为注释。应注意：它是单行注释，不能跨行。 C++的程序设计

人员多愿意用这种注释方式，它比较灵活方便。

（3）除了可以用 printf 函数输出信息外，还可以用 cout 进行输出。cout 要与运算符<<配合使用，程序中 cout 的作用是将<<运算符右侧的内容送到输出设备中输出（有关 cout 的使用，将在 2.3 节中详细介绍）。

（4）使用 cout 需要用到头文件 iostream.h，在程序的第一行用 #include 命令将该头文件“包含”进来。

程序运行时输出：

```
This is a C++ program.
This is a C++ program.
```

可以看到程序中最后两个语句的作用相同，都是输出 This is a C++ program。

1.2 面向对象的方法

程序设计语言是编写程序的工具，因此程序设计语言的发展恰好反映了程序设计方法的演变过程。这里我们首先初步介绍一下面向对象方法的基本概念和基本思想，当您学习完本书之后，对面向对象的方法会有一个深入、完整的认识。

1.2.1 面向对象方法的由来

在面向对象的方法出现以前，我们都是采用面向过程的程序设计方法。早期计算机是用于数学计算的工具，例如，用于计算炮弹的飞行轨迹。为了完成计算，就必须设计出一个计算方法，或解决问题的过程。因此，软件设计的主要工作就是设计求解问题的过程。

随着计算机硬件系统的高速发展，计算机的性能越来越强，用途也更加广泛，不再仅限于数学计算。由于所处理的问题日益复杂，程序也就越来越复杂和庞大。20 世纪 60 年代产生的结构化程序设计思想，为使用面向过程的方法解决复杂问题提供了有力的手段。因而，在 70 年代到 80 年代，结构化程序设计方法成了所有软件开发设计领域及每个程序员都采用的方法。结构化程序设计的思路是：自顶向下、逐步求精；其程序结构是按功能划分为若干个基本模块，这些模块形成一个树状结构；各模块之间的关系尽可能简单，在功能上相对独立；每一模块内部均是由顺序、选择和循环三种基本结构组成；其模块化实现的具体方法是使用子程序。结构化程序设计由于采用了模块分解与功能抽像，自顶向下、分而治之的方法，从而有效地将一个较复杂的程序系统设计任务分解成许多易于控制和处理的子任务，便于开发和维护。

虽然结构化程序设计方法具有很多的优点，但它仍是一种面向过程的程序设计方法。它把数据和处理数据的过程分离为相互独立的实体，当数据结构改变时，所有相关的处理过程都要进行相应的修改，每一种相对于老问题的新方法都要带来额外的开销，程序的可重用性差。另外，由于图形用户界面的应用，使得软件使用起来越来越方便，但开发却越来越困难。一个好的软件，应该随时响应用户的任何操作，而不是请用户按照既定的步骤

循规蹈矩地使用。例如，我们都熟悉文字处理程序的使用，一个好的文字处理程序使用起来非常方便，几乎可以随心所欲，软件说明书中决不会规定任何固定的操作顺序，因此对这种软件的功能很难用过程来描述和实现，如果仍使用面向过程的方法，其开发和维护都将很困难。

那么，面向对象的方法是什么呢？首先，它将数据及对数据的操作方法放在一起，作为一个相互依存、不可分离的整体——对象。对同类型对象抽像出其共性，形成类。类中的大多数数据，只能用本类的方法进行通讯。这样，程序模块间的关系更为简单，程序模块的独立性、数据的安全性就有了良好的保障。另外，通过后续章节中将介绍的继承与多态性，还可以大大提高程序的可重用性，使得软件的开发和维护都更为方便。

面向对象的方法有如此的优点，然而对于初学程序设计的人来说，是否容易理解、容易掌握呢？回答是肯定的。面向对象方法的出现，实际上是程序设计方法发展的一个返朴归真过程。软件开发从本质上讲，就是对软件所要处理的问题域进行正确的认识，并把这种认识正确地描述出来。面向对象方法所强调的基本原则，就是直接面对客观存在的事物来进行软件开发，将人们在日常生活中习惯的思维方式和表达方式应用在软件开发中，使软件开发从过分专业化的规则和技巧中回到客观世界，回到人们通常的思维方式。

1.2.2 面向对象的基本概念

现在，我们简单介绍面向对象方法中的几个基本概念。当然我们不能期望通过几句话的简单介绍就能让您完全理解这些概念，在本书的后续章节中，我们会不断帮助读者加深对这些概念的理解，以达到熟练运用。

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，它可从是有形的（比如一辆汽车），也可以是无形的（比如一项计划）。对象是构成世界的一个独立单位，它具有自己的静态特征（可以用某种数据来描述）和动态特征（对象所表现的行为或具有的功能）。

面向对象方法中的对象，是系统中用来描述客观事物的一个实体，它是用来构成系统的一个基本单位。对象由一组属性和一组行为构成。属性是用来描述对象静态特征的数据项，行为是用来描述对象动态特征的操作序列。

2. 类

把众多的事物归纳、划分成一些类，是人类在认识客观世界时经常采用的思维方法。分类所依据的原则是抽像，即忽略事物的非本质特征，只注意那些与当前目标有关的本质特征，从而找出事物的共性，把具有共同性质的事物划分为一类，得出一个抽像的概念。例如，石头、树木、汽车、房屋等都是人们在长期的生产和生活实践中抽像的概念。

面向对象方法中的“类”，是具有相同属性和行为的一组对象的集合。它为属于该类的全部对象提供了抽像的描述，其内部包括属性和行为两个主要部分。类与对象的关系犹如模具与铸件之间的关系，一个属于某类的对象称为该类的一个实例。

3. 封装

封装是面向对象方法的一个重要原则，就是把对象的属性和行为结合成一个独立的系统单位，并尽可能隐蔽对象的内部细节。这里有两个含义：第一个含义是把对象的全部属性和全部行为结合在一起，形成一个不可分割的独立单位。第二个含义也称作“信息隐蔽”，即尽可能隐蔽对象的内部细节，对外形成一个边界（或者说一道屏障），只保留有限的对外接口使之与外部发生联系。

4. 继承

继承是面向对象技术能够提高软件开发效率的重要原因之一，其定义是：特殊类的对象拥有其一般类的全部属性与行为，称作特殊类对一般类的继承。

继承具有重要的实际意义，它简化了人们对事物的认识和描述。比如我们认识了轮船的特征之后，再考虑客轮时，因为知道客轮也是轮船，于是可以认为它理所当然地具有轮船的全部一般特征，只需要把精力用于发现和描述客轮独有的那些特征。

继承对于软件复用有着重要意义，使特殊类继承一般类，本身就是软件复用。而且不仅于此，如果将开发好的类作为构件放到构件库中，便可以在开发新系统时直接使用或继承使用。

5. 多态性

多态性是指在一般类中定义的属性或行为，被特殊类继承之后，可以具有不同的数据类型或表现出不同的行为。这使得同一个属性或行为在一般类及其各个特殊类中具有不同的语义。

1.3 面向对象的软件开发

在整个软件开发过程中，编写程序只是相对较小的一个部分。软件开发的真正决定性因素来自前期概念问题的提出，而非后期的实现问题。只有识别、理解和正确表达了应用问题的内在实质，才能做出好的设计，然后，才能具体的编程实现。

面向对象的软件工程是面向对象方法在软件工程领域的全面应用。它包括面向对象的分析、面向对象的设计、面向对象的编程、面向对象的测试和面向对象的软件维护等内容。

1.3.1 面向对象的分析

从问题的陈述着手，建立一个说明系统重要特性的真实情况模型。为理解问题，系统分析员需要与客户一起工作。系统分析阶段应该扼要精确地抽像出系统必须做什么，而不必关心如何去实现。

面向对象的系统分析，直接用问题域中客观存在的事物建立模型中的对象，无论是对单个事物还是对事物之间的关系，都保留它们的原貌，不做转换，也不打破原有界限而重新组合，因此能够很好地映射客观事物。

1.3.2 面向对象的设计

面向对象的分析模型是针对问题域运用面向对象方法产生一个具体实现。针对这个具体的实现，面向对象的设计运用面向对象的方法来完成：

（1）把分析模型直接搬到设计中去，作为设计的一部分；

（2）根据具体实现中的用户、用户界面存储等因素补充一些与实现有关的部分，而这些部分与分析模型采用相同的表示方法和模型结构。

由于分析模型与设计方案采用了一致的表示法，不存在转换，只是局部地调整和补充，这就大大地降低了由分析模型设计方案过渡的难度，同时也减少了工作量和出错率。

1.3.3 面向对象的编程

面向对象的编程工作变得很简单，即将面向对象的设计方案中的每个成员使用面向对象的语言写出来就可以了。所需做的工作如下：

- 用具体数据结构来定义对象的属性；
- 用具体的语句来实现程序流程图中所表示的算法。

使用面向对象的方法会使得：

- 问题域的事物对应于面向对象分析模型中的全部类及对象；
- 面向对象设计模型中一部分对象类对应于面向对象分析模型，另一部分对象类对应于实现有关的因素；
- 面向对象的编程对应于面向对象的设计。

1.3.4 面向对象程序的调试

在使用面向对象的程序设计语言编写的程序中，使对象成为一个独立的程序单位，只通过有限的接口与外部发生关系，从而减少了错误影响的范围。

面向对象的调试是以对象的类作为基本调试单位，查错范围是类定义之内的属性和行为以及接口所涉及的部分。

继承性也给调试带来了极大的方便，当调试好了父类，子类的调试重点就放在它所新定义的属性及行为上。

1.3.5 面向对象的维护

由于程序与问题域的一致性，各阶段的表示也是一致的。这就减少了编程人员的理解难度。发现问题可以直接追溯到问题域，其道路比较平坦。由于系统中最容易变化的因素被封装在对象的内部，并且修改某个对象对其他对象的影响又较小，这将给调试与维护带来极大方便。

综上所述，由于面向对象方法分析问题的出发点与人们对客观事物认识的一致性，使

得软件开发工作变得更容易、更方便。

1.4 C++ 语言的词法和词法规则

1.4.1 C++语言的字符集

C++语言的字符集同于 C 语言的字符集。

C++语言的字符集由下列字符组成：

（1）大小写英文字母。

a~z 和 A~Z

（2）数字字符。

0~9

（3）其他字符。

空格 ! # % ^ & * _(下划线) — + = ~ < > / \

| . , : ; ? ’ ”( ) [ ] { }

1.4.2 单词及词法规则

单词是一种词法记号，它是由若干个字符组成的具有一定意义的最小词法单元。

下面分别讲述 C++语言的 6 种单词。

1. 标识符

标识符是程序员用来命名程序中一些实体的一种单词。使用标识符来定义函数名、类名、对象名、变量名、常量名、类型名和语句标号名等。 C++规定，标识符是由大小写字母、数字字符和下划线符组成的，并以字母或下划线开头的字符集合。

定义标识符应注意如下几点：

（1）标识符中的大小写字母是有区别的。例如， ABC、Abc、abc、ABc 等都是不同的标识符。

（2）标识符的长度，即组成一个标识符的字符个数，是不受限制的。但是，有的编译系统所能识别的标识符长度是有限的，例如，有的系统只识别前 32 个字符。

（3）在实际应用中，尽量使用有意义的单词作标识符。但是，不得用系统中已预定义的标识符，如关键字和设备字。

2. 关键字

关键字是系统中已预定义的单词， 它们在程序中表达特定的含义。下面列举出 C++语言中常用的关键字：

auto break case char class const continue

default	delete	do	double	else	enum	explicit
extern	float	for	friend	goto	if	inline
int	long	new	operato	private	protected	public
register	return	short	signed	sizeof	static	struct
switch	this	typedef	union	undigned	virtual	void
while						

以上这些关键字的含意将在本书逐渐讲述。

3. 运算符

运算符是一些用来进行某种操作的单词，这实际上是系统预定义的函数名，这些函数作用于被操作的对象上将获得一个结果值。运算符是由 1 个或多个字符组成的单词。

C++语言的运算符除了包含 C 语言中的运算符外，还增加了一些新的运算符。

C++ 语言的运算符可以重载。

4. 分隔符

分隔符被称为程序中的标点符号，它是用来分隔单词与程序正文的，它用来表示某个程序实体的结束和另一个程序实体的开始。

C++常用的分隔符包括有：

（1）空格符：用作单词之间的分隔符。

（2）逗号：用作变量之间或对象之间的分隔符；或者用作函数的多个参数之间的分隔符。

（3）分号：用于 for 循环语句中，作为关键字 for 后面的括号内的三个表达式之间的分隔符。

（4）冒号：用作语句标号与语句间的分隔符以及 switch 语句中 case （整数型表达式）与语句序列之间的分隔符。

（5）花括号：用来为函数体、复合语句等定界。

6. 注释符

注释在程序中起到对程序的注解和说明的作用，其目的是为了便于对程序的阅读和分析。C++语言中注释方法有如下：

（1）使用“ /* ”和“*／”括起来进行注释，在‘ /* ”和“ */ ”之间的所有字符都为注释符。这种注释方法适用于多行注释信息的情况。

（2）使用‘ // ”，从“ // ”后面字符开始，直到它所在行的行尾所有字符都被作为注释信息。这种方法适用于注释一行信息的情况。

这两种注释方法都可以放在程序的任一位置：程序开头、结尾及中间任何位置都可以。前一种方法可以放在某一程序中的语句行的前边或后边，甚至中间；而后一种方法可以放在某一语句行的后边，而不能放在前边或中间。

1.5 C++程序在 Visual C++.NET 中的编译运行

Visual C++.NET 版本编译系统是当前国际上比较流行的最新一种 C++编译系统。该系统功能较强，使用方便。在这里仅介绍如何使用该编译系统来运行一个 C++程序。关于该系统的更多功能可参阅本书后面的介绍及 Visual C++ 的操作说明书或在线帮助。

1. 启动 Visual C++.NET 开发环境

从“开始”菜单中选择“程序” |Microsoft Visual Studio 7.0| Microsoft Visual C++ 7.0，显示 Visual C++.NET 开发环境窗口。

2. 创建一个项目

（1）单击“文件”菜单中的“新建|项目”选项，显示“新建项目”对话框（如图 1.1 所示）。

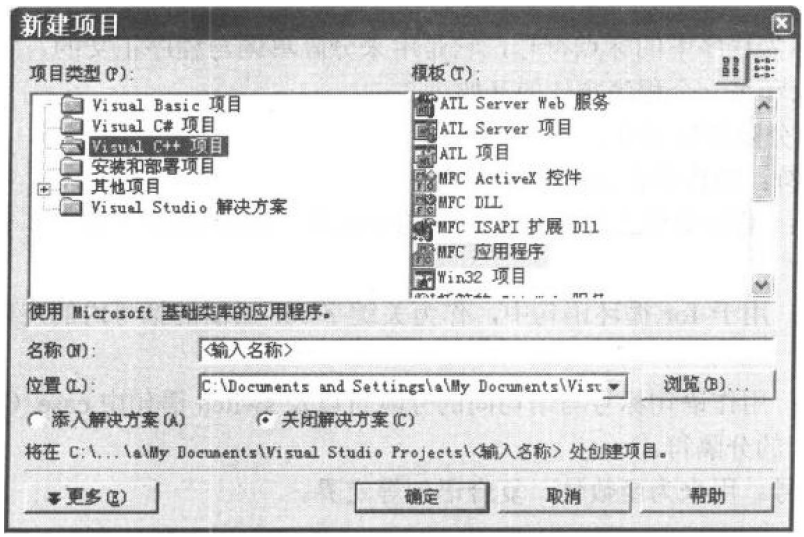

图 1.1 “新建项目”对话框

（2）单击左边的 Visual C++项目，在右边列表框中，选择“ Win32 项目 ”,在项目名称编辑框中输入：1，在位置编辑框中输入或选择好路径，然后按“确定”按钮。

（3）在接下来的对话框中，选择左边“应用程序设置”选项卡，然后选择右边的“控制台应用程序 ”(如图 1.2 所示）。单击“完成”按钮。

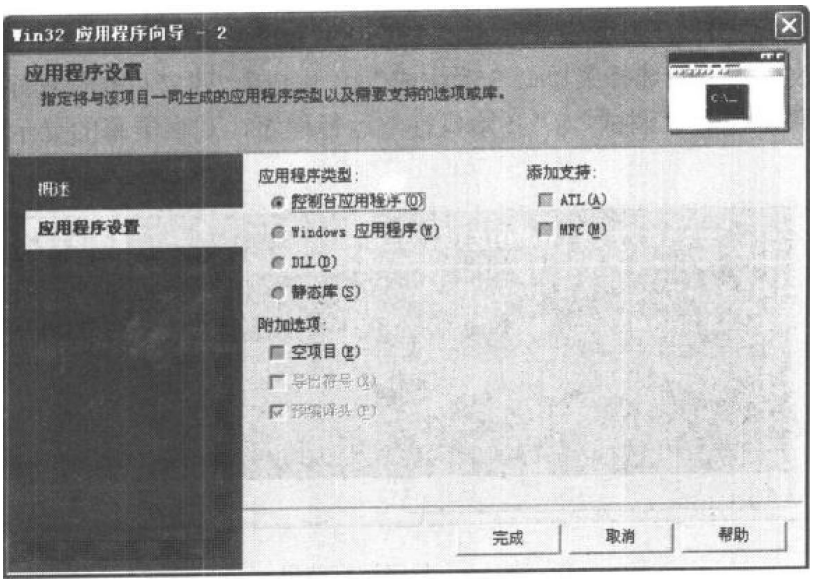

图 1.2 创建控制台应用程序

3. 建立 C++源程序文件

（1）在屏幕右边的“解决方案管理器”中双击“1.cpp”。

（2）在工作区窗口中显示了系统为你做好的函数框架。在“return 0;”语句之前加入下列语句：

```
cout<<"This is a C++ program.\n";
```

在已有#include 语句之后，添加下列语句：

```
#include "iostream.h"
```

完成后的程序如下所示：

```
// 1.cpp : Defines the entry point for the   console application.
//
#include "stdafx.h"
#include "iostream.h"
int  tmain(int argc,  TCHAR* argv[])
{
    cout<<"This is a C++ program.\n";
    return
}
```

（3）保存这个文件。

4. 建立并运行可执行程序

（1）选择菜单命令“生成”|“生成”建立可执行程序。

如果你正确输入了源程序，此时便成功地生成了可执行程序 1.exe。

如果程序有语法错误，则屏幕下方的状态窗口中会显示错误信息，根据这些错误信息对源程序进行修改后，重新选择菜单命令“生成”|“生成”,建立可执行程序 1.exe。

（2）选择菜单命令“调试”|“开始执行”运行程序，观察屏幕的显示内容（如图 1.3 所示）。

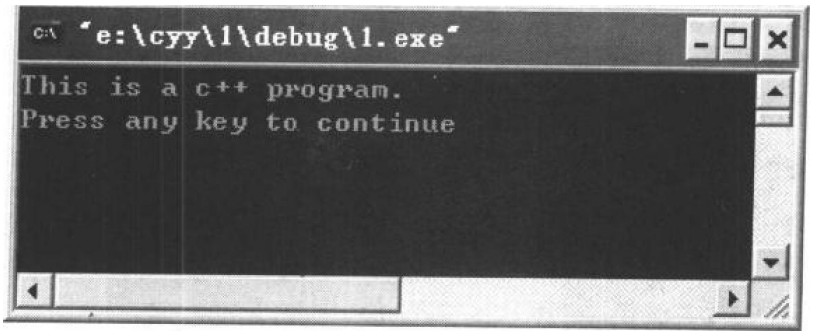

图 1.3 程序运行结果

5. 关闭解决方案

选择菜单命令“文件”|“关闭解决方案”来关闭解决方案。

1.6 小 结

C++语言是在 C 语言的基础上支持面向对象的程序设计语言，它的开发者是 AT&T 贝尔实验室的 Bjarne Stroustrup 博士。

C++中引入了类等机制来支持面向对象的程序设计。它将 C 作为它的子集，对 C 的类型系统进行了改进和扩充，并保持与 C 兼容。因此，C++比 C 更安全，功能更强。

C++面向对象方法中介绍了对象、类、封装、继承、多态性等基本概念。

面向对象的软件工程是面向对象方法在软件工程领域的全面应用。它包括面向对象的分析、面向对象的设计、面向对象的编程、面向对象的测试和面向对象的软件维护等内容。

Visual C++.NET 版本编译系统是当前国际上比较流行的最新一种 C++编译系统。该系统功能较强，使用方便。

习题一

1. 面向对象的编程语言有哪些特点？
2. 面向对象的软件工程包括哪些主要内容？
3. C++有哪些词法记号？
4. 注释的作用是什么？在程序的编译阶段，注释是被怎样处理的？

5. **什么是面向对象？如何理解面向对象中的一些基本概念：对象、类、封装、继承、多态性等？**
6. **什么是编程语言？从编程语言的发展史中给我们什么启示？**
7. **C 语言和 C++语言有什么关系？**
8. **简述源程序实现的三个步骤？**
9. **分析下面程序的输出结果。**

```
#include<iostream.h>
void main()
{
    cout<<"Beijing"<<" ";
  cout<<"Shanghai"<<" ";
  cout<<"Tianjin";
}
```

第 2 章　数据类型、运算符和控制结构

2.1 基本数据类型

数据是程序处理的对象，数据可以依其本身的特点进行分类。我们知道在数学中有整数、实数等概念，在日常生活中需要用字符串来表示人的姓名和地址，有些问题的回答只能是“是”或“否”（即逻辑“真”或“假”）。不同类型的数据有不同的处理方法，例如，整数和实数可以参加算术运算，但实数的表示又不同于整数，要保留一定的小数位；逻辑数据可以参加“与”“或”“非”等逻辑运算。

在 C++语言中提供了丰富的数据类型和运算，下面我们将对此一一做介绍。

2.1.1　基本数据类型

C++的基本数据类型有布尔型（以 bool 表示）、字符型（以 char 表示）、整型（以 int 表示）、实型（单精度浮点型以 float 表示、双精度浮点型以 double 表示）。除了 bool 型外，主要有两大类：整数和浮点数。因为 char 型从本质上说也是整数类型，它是长度为 1 个字节的整数，通常用来存放字符的 ASCII 码。其中关键字 signed 和 unsigned，以及关键字 short 和 long 被称为修饰符。

用 short 修饰 int 时，short int 表示短整型，占 2 字节。Long 可以用来修饰 int 和 double。用 long 修饰 int 时，long int 表示长整型，占 4 字节。int 型所占的字节数在不同的系统中有可能不一样，在 VC++ 6.0 编译环境中的情况（也是目前大多数编译环境中的情况），int 型和 long int 所占的字节数是一样的，都是占 4 字节。short 型和 long 型的字节是固定的，任何支持标准 C++的编译系统中都是如此。所以如果需要编写可移植性好的程序，应将整型数据定义为 short 型或 long 型。

signed 和 unsigned 可以用来修饰 char 型和 int 型（包括 long int），signed 表示有符号数，unsigned 表示无符号数。有符号整数在计算机内是以二进制补码形式存储的，其最高位为符号位，“0”表示“正”，“1”表示“负”。无符号整数只能是正数，在计算机内是以绝对值形式存放的。char 型和 int 型（包括 long int）在缺省（不加修饰）情况下是有符号（signed）的。

bool 型数据的取值只能是 false（假）或 true（真）。bool 型数据所占的字节数在不同的编译系统中有可能不一样，在 VC 7.0 编译环境中 bool 型数据占 1 字节。

程序所处理的数据不仅分为不同的类型，而且每种类型的数据还有常量与变量之分。下面我们将详细介绍各种基本数据类型的数据。

2.1.2 常量和符号常量

所谓常量是指在程序运行的整个过程中其值始终不可改变的量，也就是直接使用符号表示的值。例如，123，3.5，‘A’都是常量。

1. 整型常量

整型常量即整常数，包括正整数、负整数和零。C++整常数可用以下三种形式表示。

（1）十进制整型常量，表示形式为：[±]若干个 0～9 的数字，但数字部分不能以 0 开头。如 123，－145，0 等。

（2）八进制整型常量，以 0 开头的数是八进制数，其表示形式为：

[±]0 再加上若干个 0～7 的数字，如 0123 表示八进制数 123，即$(123)_8$。

（3）十六进制整型常量，以 0x 开头的数是十六进制数，其表示形式为：

[±]0x 再加上若干个 0～9 的数字及 A～F 的字母（大小写均可），如 0x123 表示十六进制数 123，即 $(123)_{16}$。

整型常量可以用后缀字母 L（或 l）表示长整型。

2. 实型常量

实型常量即以文字形式出现的实数，实数有两种表示形式：十进制小数形式和指数形式。

（1）十进制小数形式：例如，123.5，－123.5 等。

（2）指数形式：例如，0.345E+2 表示 $0.345*10^2$，－34.4E－3 表示$-34.4*10^{-3}$，其中，字母 E 可以大写或小写。

实型常量缺省为 double 型，如果后缀为 F（或 f）则为 float 型。

（3）字符常量

字符常量是单引号括起来的一个字符，如‘a’，‘x’，‘?’等都是字符常量。

除了以上形式的字符常量外，C++还允许用一种特殊形式的字符常量，即以“＼”开头的字符序列。例如换行、制表符、回车等等。这些字符是不可显示字符，也无法通过键盘输入，因此 C++提供一种称为转义序列的表示方法来表示这些字符，表 2.1 列出了 C++预定义的转义序列。

表 2.1 C++的转义序列及其作用

字符形式	含义	ASCII 码	字符形式	含义	ASCII 码
\n	换行	10	\f	换页	12
\t	水平制表	9	\\	反斜杠字符“\”	92
\b	退格	8	\'	单引号字符	39
\r	回车	13	\"	双引号字符	34

表 2.1 中列出的字符称为“转义序列”，意思是将反斜杠（\）后面的字符转换成另外的意义。如‘\n’中的‘n’不代表字母 n 而作为“换行”符。

无论是不可显示字符还是一般字符， 都可以用十六进制或八进制 ASCII 码来表示，表示形式是：

\hhh 八进制形式

\xhhh 十六进制或十六进制数

例如，‘a,的十六进制 ASCII 码是 61，于是‘ a’ 也可以表示为‘\x61’。

3. 字符串常量

字符串常量简称字符串，是用一对双引号括起来的字符序列，例如， “Hello”，“CHINA”，都是字符串常量。

4. 布尔常量

布尔型常量只有两个：true（真）和 false（假）。

5. 符号常量

符号常量是用一个标识符代表一个常量，即标识符形式的常量。符号常量在使用之前一定要首先声明，这一点与变量很相似。常量声明语句的形式为：

const 数据类型说明符 常量名=常量值：

或：

数据类型说明符 const 常量名 =常量值；

例如，我们可以定义一个代表圆周率的符号常量：

```
const float pi=3.1415926;
```

注意，符号常量在声明时一定要赋初值，而在程序中间不能改变其值。例如，下列语句是错误的：

```
const float pi;
pi=3.1415926;//错！常量不能被赋值
```

使用符号常量，由于只在定义时赋以初值，修改起来十分简单，从而可以避免因修改常量值带来的不一致性。

2.1.3 变量

在程序的执行过程中其值可以变化的量称为变量，变量是需要用名字来标识的。

1. 变量的数据类型和初值

就像常量具有各种类型一样，变量也具有相应的类型。变量在使用之前需要首先声明其类型和名称。变量名也是一种标识符，因而给变量命名时，应该遵守标识符构成规则。在同一语句中可以声明同一类型的多个变量，变量声明语句的形式如下：

数据类型 变量名 1,变量名 2…，变量名 n;

例如，下列两条语句声明了两个 int 型变量和三个 float 型变量；

```
int num,total;
```

```
float v,r,h;
```

在程序运行时，系统会给每一个声明过的变量分配内存空间，用于存放对应类型的数据，因而变量名也就是对相应内存单元的命名。我们在声明一个变量的同时，也可以给它赋以初值，而这实质上就是给对应的内存单元赋值，例如：

```
int  a=123;
```

在声明变量的同时赋初值还有另外一种形式，例如： `int  a(123);`

2. 变量的存储类型

变量除了具有数据类型外，还具有存储类型：

auto 存储类：采用堆栈方式分配内存空间，属于暂时性存储，其存储空间可以被若干变量多次覆盖使用。

register 存储类：存放在通用寄存器中。

extern 存储类：在所有函数和程序段中都可引用。

static 存储类：在内存中是以固定地址存放的，在整个程序运行期间都有效。

3. 变量的引用

“引用”（reference）是 C++的一种新的变量类型，是对 C 的一个重要扩充。它的作用是为一个变量起一个别名。假如有一个变量 a，想给它起一个别名 b，可以这样写：

```
int a;
int &b=a;
```

这就声明了 b 是 a 的“引用”，即 a 的别名。经过这样的声明后，使用 a 或 b 的作用相同，都代表同一个变量。

声明引用并不另开辟内存单元，b 和 a 都代表同一变量单元。在声明一个引用型变量时，必须同时使之初始化，即声明它代表哪一个变量。在声明一个变量的引用后，在本函数执行期间，该引用一直与其代表的变量相联系，不能再作为其他变量的别名。下面的用法是不合法的：

```
int a1, a2;
int &b=a1;
b=a2;  //企图使 b 变成 a2 的引用是非法的。
```

通过下面的例子可以了解引用的简单使用。

```
#include<iostream.h>
#include<iomanip.h>
void main( )
{
int a=10;
int &b=a;
a=a*a;
```