普通高等教育“十二五”规划教材

C++语言程序设计

张 鸿 主编

中国电力出版社
CHINA ELECTRIC POWER PRESS

普通高等教育“十二五”规划教材

C++语言程序设计

主　编　张　鸿
副主编　张玉宏
编　写　王　黎　张　蕾
主　审　苏锦祥

中国电力出版社
CHINA ELECTRIC POWER PRESS

内 容 提 要

本书为普通高等教育“十二五”规划教材。C++是一种高效实用的程序设计语言。本书是作者总结多年教学和编程的实践经验，并借鉴国内外大量资料精心编写而成的。书中以浅显的语言深入讲解语法规则，并配以大量的图解和例题，力争做到深入浅出，详略得当，便于读者在学习时有的放矢。本书共 9 章，包括面向对象程序设计概述、C++基础、类与对象、继承性、多态性、名称空间、模板、输入/输出流和文件处理、异常处理。本书以应用为目的，注重培养应用能力。除特殊说明外，本书中的所有例程均可在 Windows 环境下的 Visual C++ 6.0 及以上版本和 Linux 环境下的 GCC 编译通过。

本书可作为普通高等院校程序设计基础课程的本、专科教材（可以根据本科、专科教学要求的不同进行适当取舍），也可作为 C++语言自学者的参考书。

图书在版编目（CIP）数据

C++语言程序设计 / 张鸿主编. —北京：中国电力出版社，2014.8

普通高等教育“十二五”规划教材

ISBN 978-7-5123-6111-9

Ⅰ. ①C… Ⅱ. ①张… 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（2014）第 144728 号

中国电力出版社出版、发行

（北京市东城区北京站西街 19 号 100005 http://www.cepp.sgcc.com.cn）

汇鑫印务有限公司印刷

各地新华书店经售

*

2014 年 8 月第一版 2014 年 8 月北京第一次印刷

787 毫米×1092 毫米 16 开本 19 印张 461 千字

定价 **38.00** 元

敬 告 读 者

本书封底贴有防伪标签，刮开涂层可查询真伪

本书如有印装质量问题，我社发行部负责退换

版 权 专 有 翻 印 必 究

序　言

用计算机解决问题，就需要通过一种计算机能接受的形式，即程序语言，来描述"计算"的步骤，即算法；而如何更好地用程序语言描述算法，即更好地进行程序设计，就需要一整套程序设计的理论、方法和技术。因此，计算机诞生的半个多世纪以来，对程序语言与程序设计理论、方法和技术的研究始终成为计算机科学技术发展的主流之一。

C++作为面向对象的程序语言是在C语言的基础上扩充了面向对象的机制而形成的。C++完全支持面向对象的程序设计方法和技术，支持面向对象的四个特性，即抽象性、封装性、继承性和多态性。

面向对象的程序设计是程序设计方法、技术发展等阶段中最先进的。面向对象方法的基本思想就是直接面向客观实际进行软件开发，将人们在日常生活和工作中习惯的思维方式和表达方式应用于软件开发之中。

C++出现30多年来仍在不断演化。1994年2月标准模板库（STL）正式成为C++的一部分，促使C++程序员的思维方式更朝向泛型编程（generic program）发展。1997年，C++语言成为美国国家标准（ANSI）。1998年，C++语言又成为了国际标准（ISO）。目前，C++语言已成为使用最广泛的面向对象程序设计语言之一。正像C++的奠基者Bjarne Stroustrup博士所说的："C++是一个通用的程序设计语言，其核心应用领域是最广泛意义上的系统程序设计。此外，C++还被成功地用到许多无法称为系统程序设计的应用领域中。从最现代的小型计算机到最大的超级计算机上，以及几乎所有的操作系统上都有C++的实现。"

作为计算机专业的大学生掌握一门乃至几门程序设计语言的语法、语义和语用的基本知识以及受到程序设计方法、技术的基本训练是必要的，也是至关重要的。

本书是作者在多次教授"C++程序设计"课程后，并对教材组织和教学方法潜心钻研的基础上编写而成的，是一本有特色的计算机专业及相关专业的教材和参考书，相信必将受到广大读者的青睐。

苏锦祥

2014年5月1日于郑州大学

前　言

C++语言是在 C 语言的基础上扩充了面向对象机制而形成的一种面向对象程序设计语言。20 世纪 90 年代以来，随着面向对象程序设计迅速地在全世界流行，并成为程序设计的主流技术，近年来推出的新一代程序设计语言、软件开发工具与环境以及操作系统，几乎都是面向对象的。如果说 C 语言作为编程入门课程，介绍结构化程序设计方法的话，C++语言则提供了面向对象程序设计风格，学会了 C++面向对象程序设计，对别的面向对象语言和工具环境也就不难触类旁通了。因此，C++成为各高等院校介绍面向对象程序设计的首选语言，是计算机科学及相关专业的重要基础课程。

本书共 9 章，分别介绍面向对象程序设计概述、C++基础、类与对象、继承性、多态性、名称空间、模板、输入/输出流和文件处理、异常处理。附录中给出了 ASCⅡ码字符集，C++关键字，Visual C++常用库函数，常用运算符的功能、优先级和结合性。各章开头处提出了本章的学习目标，后面附有习题，便于读者学习、练习和复习。

本书由郑州升达经贸管理学院张鸿副教授主编，河南工业大学张玉宏博士副主编，河南工业大学王黎副教授和郑州升达经贸管理学院张蕾讲师编写。其中张鸿编写第 1、3、5 章，张玉宏编写第 4、6、7 章，王黎编写第 2、8 章，张蕾编写第 9 章及附录。全书由张鸿统稿。

在本书编写期间，河南省计算机理事会名誉会长苏锦祥教授给予了指导和把关，主审了本书，并为本书作了序，在此表示最衷心的感谢。

在本书的编写过程中，郑州升达经贸管理学院的甄姬娜、王丽娟等老师及河南工业大学的李冰凯、蔡耀、赵宏志、黄鹏飞、谢振强等同学参与了校正工作。此外，编者参阅了部分网络资源，在此对其作者一并表示感谢。

限于编者水平，疏漏与不足之处在所难免，敬请广大读者不吝赐教。

编　者

2014 年 4 月

此为试读，需要完整PDF请访问：www.ertongbook.com

目　录

第 1 章　面向对象程序设计概述

学 习 目 标

- 了解面向对象思想的产生背景，并熟悉面向对象程序和传统程序结构上的特点及本质区别。
- 了解类和对象的含义及二者之间的关系。
- 了解消息和方法的含义及二者之间的关系。
- 了解 OOP 的特点，进一步理解 OOP 的四个基本特性。

随着计算机应用的普及和深入，软件系统的规模和复杂性急剧扩大和增加，软件的质量、可靠性和可维护性等要求也越来越高。因此，更好的、适用的程序设计方法、设计语言和开发工具便成了软件设计人员不断追求的目标。

面向对象思想是当今计算机技术发展的重要成果和趋势之一，它应用于程序设计中形成的面向对象技术（Object-Oriented Technique，OOT）是程序设计领域中的一种新方法。这种方法可以使程序设计的整个过程更加贴近现实世界，从而能够更方便地进行分析、设计和实现，有利于系统的延伸和扩展，也可以提高软件的复用性。本章将主要介绍面向对象思想的产生背景和面向对象程序设计（Object-Oriented Programming，OOP）中涉及的基本概念等有关问题。

1.1　面向对象思想的产生背景

早期计算机中运行的程序大都是为特定的硬件系统专门设计的，称为面向机器的程序。这类程序的运行速度和效率相对都很高，但是程序的可读性和可移植性却很差。随着硬件的发展，相应的，软件要及时进行更新，软件开发的规模要扩大，这类面向机器的程序逐渐被以 Pascal、C 等为代表的面向过程语言所编写的面向过程程序所取代。

面向过程的程序采用面向过程的思想进行分析和设计。面向过程的思想主要是把程序看成是处理数据的一系列过程，通过调用过程得到问题的解。简单讲，程序由过程定义和过程调用组成，这是面向过程程序设计的范型（Paradigm，是指设计程序的规范、模型和风格），主要特点可使用以下等式表示：程序=过程+调用。

请看［例 1.1］，一个面向过程风格的 C++程序（暂不关注语法）。

【例 1.1】 输入一组年、月、日，并输出。

```
#include<iostream.h>
//定义一个 Date 结构体类型(只有年、月、日属性,隐含为 public 成员)
struct Date
{
      int year,month,day;
```

```
};
//定义输入过程
void Input(Date *pd)
{
     cout<<"Please input year,month,day:";
     cin>>pd->year>>pd->month>>pd->day;
}
//定义输出过程
void Output(Date *pd)
{
   cout<<pd->year<<"年"<<pd->month<<"月"<<pd->day<<"日"<<endl;
}
int main()
{
     Date d;                                    //定义一个结构体变量
     Input(&d);                                 //调用输入过程
     Output(&d);                                //调用输出过程
     return 0;
}
```

程 序 说 明

本例是一个面向过程风格的C++程序，功能非常简单：输入一个日期并输出。提供本例只是为了进一步说明面向过程风格的C++程序结构特点，使读者能够轻松过渡到面向对象风格的C++编程（参考［例1.2］）。

仔细分析本程序的结构，处理的日期数据（含有年、月、日三个公有成员的结构体）和作用在该日期之上的两个操作Input( )、Output( )在定义上是分开的，即数据和作用在其上的操作相分离。

进一步考虑下面的问题。

改写本例中的函数main( )，在调用语句“Input(&d);”之后增加一条“d.year=2006;”语句，其功能是修改Date变量d的成员year值为2006。那么在输出变量d值时，其year值就不再是Input( )函数中输入的值，而是2006。也就是说，日期d的值可以随意被修改。请各位读者积极开拓思维，延伸思考，如果这种情况出现在实际应用的管理信息系统中，就必将会引起极其严重的后果。那么该如何避免类似Date结构体中的成员不被外界随意或恶意修改呢？

另外，对于比较简单且单一的系统来讲，面向过程的问题求解方法可以精确地、完备地描述其具体的求解过程。但是，现实世界中的许多事物之间都有一定的关联。例如，“小学生”、“中学生”和“大学生”之间都有一定的共性，他们都属于“学生”，这四个集合之间的关系如何在程序设计中确切描述。“学生”和“人”这两个集合之间的关系又该如何确切描述……类似“学生”或“人”这样的包含了多个相互关联的复杂系统，面向过程的问题求解方法也是不能将其描述清楚的。

对于以上两个方面的问题，数据和作用在其上的操作相分离的程序结构是根本解决不了的。也就是说，采用面向过程的开发方法已经远远不能满足越来越多的实际需求了，我们迫切需要寻求一种新的开发方法以适应新的环境和要求。面向对象的思想就在这样的环境下应

运而生。

面向对象思想是一种新的程序设计思路、描述和处理问题的方法，与传统程序的开发方法不同。采用面向对象的思想求解问题时，考虑的不仅仅是孤立的单个过程，而且还有孕育所有这些过程的母体系统，包括系统的组成、系统中各种可能的状态及系统中可能产生的各种过程与过程引起的系统状态切换等。

通过下面的［例 1.2］，我们首先简单了解一下面向对象风格的程序。

【例 1.2】 改写［例 1.1］，使其具备面向对象风格。

```
#include<iostream>
using namespace std;
//定义 Date 类类型(属性+操作)
class Date
{
    int year,month,day;              //有三个属性:年、月、日,隐含为 private 成员
public:                              //定义两个操作,均为 public 成员
    //定义成员函数 Input(),实现输入操作
    void Input()
    {
       cout<<"Please input year,month,day:";
       cin>>year>>month>>day;
    }
    //定义成员函数 Output(),实现输出操作
    void Output()
    {
      cout<<year<<"年"<<month<<"月"<<day<<"日"<<endl;
    }
};
//使用 Date 类型
int main()
{
    Date d;                          //创建一个 Date 类型对象
    d.Input();                       //向对象 d 发送 Input 消息,即调用输入过程
    d.Output();                      //向对象 d 发送 Output 消息,即调用输出过程
    return 0;
}
```

程 序 说 明

面向对象程序的基本元素是对象，而对象是由描述对象属性或状态的数据和作用在数据之上的一些操作所构成的一个封装体（这正是现实世界中所有实体具备的特征）。程序中的一切操作都是通过向对象发送相应的消息来实现，对象接收到消息后，执行有关方法完成相应的操作。类的定义和类的使用两部分一起组成了面向对象风格的程序。

本例程中，年、月、日描述了 Date 类对象的属性或状态，作用在其上的操作 Input( )、Output( )描述的是 Date 类对象的操作或行为。语句“d.Input();”表示向对象 d 发送输入消息，类似地，“d.Output();”表示向对象 d 发送输出消息。因此，面向对象程序设计范型的主要特点可使用以下等式来表示：程序=对象+消息。

此为试读，需要完整PDF请访问：www.ertongbook.com

请各位读者认真比对本例程序和［例 1.1］程序的结构，可以看出：面向对象程序是将数据和作用在其上的操作封装在一起。

这样，在［例 1.1］中提出的两个方面的问题也就迎刃而解了。为了避免成员年、月、日不被外界随意或恶意修改，在 C++中可以使用 private 关键字来进行限制，那么这些成员就变成了私有的，从而保护了数据的安全。另外，使用继承（详见第 4 章）可以把“小学生”、“中学生”、“大学生”三个集合定义为“学生”集合的子集，而集合“人”又可以作为“学生”集合中一个成员。

另外，需要说明的是，C 语言在发展过程中建立了功能丰富的函数库，C++从 C 语言继承了这份宝贵的财富，在 C++程序中依然可以使用这些函数库。如果要使用函数库中的函数，就必须在程序文件中包含有关的头文件。在 C++中使用头文件有如下两种方法。

（1）用 C 语言的传统方法。C 语言标准中，头文件包括后缀.h，如 math.h、stdio.h。因此，C++程序文件中如果使用头文件，只需在文件中包含所用的头文件即可。例如，如果使用输入/输出流的操作，就可以使用包含指令“#include<iostream.h>”。

（2）用 C++的新方法。C++新标准要求系统提供的头文件不包括后缀.h，如 iostream。为了表示与 C 语言的头文件有联系又有区别，C++所用的头文件名是在 C 语言的相应头文件名（但不包括后缀．h）之前加一字母 c。例如，C 语言中实现输入与输出操作所使用的头文件名为 stdio.h，在 C++中相应的头文件名为 cstdio；C 语言中的头文件 string.h，在 C++中相应的头文件名为 cstring。但也有例外的，如 C 语言中的头文件 iostream.h，在 C++中相应的头文件名为 iostream。

由于 C++新标准中的库函数都是在名称空间（namespace，详见第 7 章）的 std 中声明的，因此在程序中要对名称空间 std 作声明。例如：

```
#include<cstdio>
#include<cmath>
using namespace std;
```

C++提供的名称空间（有些参考书中被称为命名空间）机制是为了解除无法使用同名常量（变量）或同名类的限制，来防止命名的冲突。

目前，所用的大多数 C++编译系统既保留了 C 语言的用法，又提供了 C++的新方法。上面两种用法在大部分功能上是等价的，如表 1.1 所示。但是用 C++的新方法纳入头文件更符合 C++的面向对象思想，也更有内涵。［例 1.2］就采用新标准纳入头文件，本书后续章节中的所有例程也均采用 C++新标准。

表 1.1　　C 语言传统风格和 C++风格头文件对比

C 语言传统风格 头文件	C++风格 头文件
#include<stdio.h>	#include<cstdio>
#include<math.h>	#include<cmath>
#include<string.h>	#include<cstring>
#include<stdlib.h>	#include<cstdlib>
#include<memory.h>	#include<memory>

1.2　面向对象程序设计的基本概念

在编写面向对象风格的 C++程序之前，首先必须了解以下几个概念。

1.2.1　对象和类

1. 对象

在现实世界中，任何可识别的实体都是对象（object），而这些对象总是具有一些静态的属性（或状态）和一些动态的行为（或操作、功能）。对象的属性表示了它所处于的状态，而对象的行为则用来改变对象的状态实现特定的目的。

例如，张三家的彩色电视机是一个具体存在的，有外形、尺寸、颜色等状态和开、关、频道设置等功能的实体。而这样的一个实体，在面向对象程序设计中，就可以表示成一个计算机可理解、可操纵并且具有一定属性和行为的电视机对象。

因此，OOP 中的对象就是由一组描述属性的数据和作用在其上的一组操作所构成的一个封装体。

2. 类

简单地讲，类（class）是同种对象的集合与抽象。

例如，我们日常接触的电视机有很多，张三家的彩色电视机、李四家的黑白电视机都属于电视机，这些实体在面向对象程序中将被表示成同类型中的两个对象。而这些代表不同电视机实体的对象之间存在着很多共性，如都可以接收并播放电视信号，调节音量、亮度、色彩……因此，为了描述和处理问题的方便，在 OOP 中定义了类的概念来表示同种对象的公共属性和特点。从这个意义上讲，类是一种抽象数据类型（Abstract Data Type，ADT），它是所有具有相同属性和相同操作的一组对象的集合与抽象。

3. 类和对象之间的关系

通过上面的分析，类和对象之间的关系就变得十分明了：抽象和具体。类是一个抽象的概念，而对象是类的实例（instance），是一个具体的概念。

类和对象的这种关系在现实世界中也很容易理解。如“电视机”“学生”“桌子”等都是抽象的概念，属于类；那么，“张三家的彩色电视机”“名字为张鸿的学生”“郑州升达经贸管理学院 94 级信息工程系信息管理专业甲班的讲桌”等都分别是“电视机”“学生”“桌子”等类实例化的结果，即对象。

在 OOP 中，总是先定义类，再用类创建对象。各位读者请思考，为什么会有那么多凉甜的冰淇淋，它们的形状是一模一样的呢？原因很简单，先制作一个冰淇淋的模板出来。

1.2.2　消息和方法

1. 消息

消息（message）就是要求对象进行某种活动（或执行某个操作）的信息。在 C++语言中，“消息”其实就是“函数调用”。也就是说，“消息”告诉对象做什么（What to do）。

现实世界中的实体不是孤立存在的，它们之间存在着各种各样的联系，也正是它们之间的相互作用和联系，才形成了世间各种不同的系统。那么，在 OOP 中，对象之间同样也需要联系和作用，这种机制就称为对象间的消息传递，而它也是对象间进行通信的唯一方式。

2. 方法

方法（method）是指对对象实施的各种具体操作。在C++语言中，“方法”就是“成员函数”的具体实现。换句话讲，“方法”告诉对象如何做（How to do）。

3. 消息和方法之间的关系

对象根据接收到的消息要调用相应的方法；反过来，只有有了方法，对象才能响应相应的消息。

例如，当你更换电视频道时，需要使用频道选择按钮来给电视机对象发送你要换台的消息；而电视机对象，通过选择和执行一个方法来响应消息。至于具体如何实现，那就是电视机的设计人员关心的问题啦。

1.3 面向对象程序设计的基本特性及其优点

OOP是通过模仿人类建立现实世界模型的方法，包括概括、分类、抽象和归纳等进行软件开发的一种设计方法，它具有以下基本特性：抽象性（abstraction）、封装性（encapsulation）、继承性（inheritance）和多态性（polymorphism）。

1.3.1 面向对象程序设计的基本特性

1. 抽象性

抽象是指从具体的实例中提取出其共同的特性并加以描述的过程。

抽象包括两个方面的内容：一是数据抽象，即描述某类对象的公共属性；二是行为抽象，即描述某类对象的共同行为特征或具有的共同操作。在C++语言中，通过类，可以很方便地实现抽象。

在面向对象方法中，抽象是通过对一个特定系统进行分析和归纳，强调系统中某些本质的特性，而忽略其不必要的特性，从而对系统进行的简化描述。即使对同一个研究对象，如果研究问题的侧重点不同，就可能产生不同的抽象结果。例如，在设计一个××学校教职员工工资管理的过程中，考察某个员工的工资时，一般只关心他的姓名、学历、职称等，而对他的年龄、肤色、家庭住址等信息就可以忽略。如果要对该校的教职员工设计一个健康信息档案，则关心的特征就有所不同了。因此，类的设计并没有统一的标准和方法，要根据题目的实际要求及依赖设计人员的经验和技巧。如果设计得当，既有利于软件的扩充，又能够提高软件的可复用性。

通过抽象建立的类具有一定的普遍性，从而可以使用在不同的应用领域中。

2. 封装性

封装是指将数据和作用在其上的操作集成在一起，形成一个实体，使外界不了解它的详细内情。

在C++程序设计中，对象就是实现封装的机制。它就好比是一个黑盒子，表示对象属性的数据和实现各个操作的代码都被封装在里面，外界是看不到的，更不能从外面直接访问或修改这些数据及操作。使用对象时，只需要了解它向外界提供的接口形式，而不必知道对象内部的数据描述和具体的功能实现。

通过封装，可以建立起具有严格内部结构的类，使得内部数据可以得到很好的保护，减少外界的干扰和影响，较好地实现信息隐藏。同时，封装也可以保证类保持自身的独立性，

可使用在不同应用环境中。

抽象性和封装性的具体应用详见第 3 章。

3. 继承性

继承提供了创建新类的一种方法。所谓继承，是指从已有类的基础上建立新类的过程。新类可以继承已有类的属性和操作，也可以对这些属性和操作进行修改和扩充，增添自己特有的一些性质和特征。

继承也是实现软件复用的一种方式。通过继承，可以将一个个原来彼此孤立但之间具有一定关联的类有效地组织起来，形成一个类的层次结构或类体系，从而很方便地实现应用的扩展和已有代码的重复使用。这在保证质量的前提下大大减轻了软件开发人员的工作量，提高了软件的开发效率。

有关继承性的具体应用详见第 4 章。

4. 多态性

同一个消息被一个类体系中的不同对象接收时，可能导致完全不同的行为，这种现象被称为多态。简单地讲，就是“同一接口，多种方法”。

例如，当你关闭一个 Word 文档时，Windows 系统会提供如图 1.1 所示的一个对话框。在这个对话框中，有三个命令按钮：“是（Y）”按钮、“否（N）”按钮和“取消”按钮。当单击“是”按钮时，该文档就被保存；而单击“否”按钮时，该文档就不会被保存。这两个命令按钮对象接收的是同一个 “单击”消息，却产生了完全不同的行为。

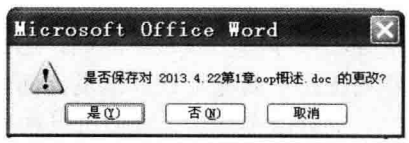

图 1.1　Windows 系统中的对话框

使用多态，一方面可以减轻程序设计人员的记忆负担；另一方面也可提高程序设计的灵活性。程序中可以通过极为简单的操作来访问同一类体系中的不同对象。

在后面章节中介绍的函数重载、运算符重载、虚函数和模板都体现了多态性。

1.3.2　面向对象程序设计的主要优点

面向对象程序设计方法的出现是计算机软件技术发展中的一个重大变革和飞跃。这种方法从根本上改变了人们以往的软件设计思维模式，力求符合人们日常自然的思维习惯。由 OOP 的特性可以看出，OOP 有如下优点。

1. 可控制程序的复杂性

OOP 中采用抽象和信息隐藏等技术，把表示事物属性或状态的数据和作用在数据之上的操作都封装在一个个类中，作为相互依存、不可分割的统一体来处理。这样，在程序中任何要访问这些数据的地方都只需要简单地通过传递消息或方法调用来实现，从而有效地控制了程序的复杂性。

2. 可增强程序的模块化

类是一种抽象数据类型（ADT），由于它的相对完整性，在程序中也可将其看成一个模块。这个模块要比面向过程风格程序中的函数模块或子程序的独立性强得多，从而能更好地支持大型程序设计。

3. 可提高软件的复用性

一个软件项目中所开发的模块不仅能够在本项目中使用，也可以重复使用在其他项目中，

从而在多个不同的系统中发挥作用，这种现象被称为“复用”。软件复用是提高软件开发效率的一个很重要的途径。类作为一个含有数据和操作的独立集成模块，既可以组合成一个新类，又可以派生出新的类，可以提高类的复用性。

4. 可改善程序的维护性

软件维护是软件生命周期中的最后一个且是非常重要的环节，而传统程序的结构又不能很好地适应需求变化，非常不利于程序维护。由于对对象的操作只能通过消息传递来实现，因此，只要方法的界面不变，对方法实现的任何修改都不会导致发送消息的主调模块的改变，这对程序的维护带来了极大的方便。另外，类的封装机制和信息隐藏使得外界不能非法访问其中的数据和操作，从而大大减少了程序的意外结果。

5. 使现实世界的复杂分类系统描述更加自然化

OOT 是把数据和作用在其上的操作集成在一起，看成一个封装体来处理，这与现实世界中实体的构成是一致的。因此，在 OOP 中使用类来描述现实世界中的实体和分类系统（如“小学生”“中学生”“大学生”等可作为“学生”的派生类来定义）就是很方便、很自然的事情了，从而延伸和扩展了计算机系统的描述和处理范围。

6. 能更好地适应新的硬件环境要求

OOP 中的对象、消息传递等思想和机制，恰好与并行处理、分布式系统、多机系统及网络等硬件环境相吻合。因此，进行面向对象的程序设计能更好地开发出适应这些新环境的软件系统，从而能更充分地发挥面向对象技术的优势。

综合以上优点，我们可以初步了解到：OOT 是在软件开发的需求刺激下发展起来的一种新技术、新方法和新思想，设计和开发面向对象的程序在很大程度上能够满足当前软件开发的迫切需求。

1.4 面向对象程序设计语言的发展

相对于硬件而言，计算机程序设计语言的发展是较缓慢的。尽管程序不是软件的全部，但在很大程度上决定着软件的质量。计算机程序设计语言的发展大致经历了以下几个阶段：①面向机器的语言；②面向过程的语言；③面向对象的语言。

下面简单介绍一下几个较有代表性的面向对象程序设计语言（Object-Oriented Programming Language，OOPL）。

1. Simula 语言

面向对象技术应用于程序设计语言可追溯到 20 世纪 60 年代推出的 Simula 语言。Simula 语言被公认为 OOPL 的鼻祖。在该语言中首次提出了模拟人类的思维方法，把数据和相关的操作集成在一起的思想。但是由于当时硬件条件的局限和面向对象方法本身不够成熟，这种思想和技术并没有得到推广和使用。随着软件危机的出现和过程化开发方法固有局限性的暴露，人们把目光重新转回到面向对象的方法上来。

Simula 语言中提供了对象、类及继承等重要概念和特性，奠定了面向对象语言的基础。

2. Smalltalk 语言

1972 年由美国的 Xerox 公司设计的 Smalltalk 语言是第一个真正的 OOPL，它的风格起源于 Simula。从 20 世纪 70 年代开始，经过 Smalltalk-72、Smalltalk-74、Smalltalk-76 和 Smalltalk-78

等几个版本的不断改进和完善，最后形成完整的 Smalltalk-80 版本。Smalltalk 在系统的设计中强调对象概念的统一，并引入和完善了对象、类、方法、继承和动态绑定等概念和机制。Smalltalk-80 被看成是一种纯粹的 OOPL，是程序设计语言领域中一个重要的里程碑。

3. C++语言

C++语言在 20 世纪 80 年代早期由贝尔实验室 B.Stroustrup 等人对传统 C 语言进行改进和扩充而形成的一种面向对象的语言，它在 C 语言基础上增加了面向对象的概念。由于 C++既支持传统的程序设计方法，又支持面向对象的程序设计方法，因此，它实际是一种混合型的程序设计语言。有丰富的应用基础和开发环境的支持，熟悉传统语言的程序员能很快学会使用这种混合型语言，因此 C++很快得到普及。

C++有很多版本，如 Microsoft C++、Turbo C++、Borland C++、Visual C++等。其中，Visual C++系列是 Microsoft 公司在 Microsoft C++的基础上推出的开发 Windows 应用程序的可视化开发工具。利用版本较高的 Visual C++提供的大量 MFC 类库和可视化编程工具，用户可以开发出规模更大、功能更加复杂的 Windows 应用程序，而编程人员付出的工作量将会大大减少。Visual C++是目前市场上使用最多的可视化编程语言之一，本教材中的所有示例都在 Visual C++6.0 开发环境中调试通过。

4. Java 语言

Java 语言是由 SUN 公司的 J.Gosling 和 B.Joe 等人在 20 世纪 90 年代开发出的一种 OOPL。Java 是一种广泛使用的网络编程语言，它不仅简单（没有指针）、不依赖于机器结构、采用面向对象思想；而且最大限度地利用了网络，Java 的应用程序（applet）可在网络上传输。另外，Java 也提供了丰富的类库，使编程者可以很方便地建立自己的系统。

1.5 重 点 与 难 点

1. 重点内容

（1）面向对象程序与传统程序的本质区别在于：前者是将数据和作用在其上的操作封装在一起；而后者是将二者分离的。

（2）对象和类的含义及关系；消息和方法的含义及关系。

（3）OOP 的四个基本特性。

2. 难点内容

（1）OOP 四个基本特性的应用。

（2）类的合理设计。

1.6 习　　题

1．面向对象程序与传统程序的区别是什么？

2．什么是类、对象？这二者之间的关系是什么？请举例说明。

3．什么是消息、方法？这二者之间的关系是什么？请举例说明。

4．OOP 的基本特性有哪些？请举例说明。

此为试读，需要完整PDF请访问：www.ertongbook.com