



21st CENTURY
规划教材

面向21世纪高等院校计算机系列规划教材
COMPUTER COURSES FOR UNDERGRADUATE EDUCATION

C++ 程序设计

刘加海 祁亨年 陈胤 主编



科学出版社
www.sciencep.com



面向21世纪高等院校计算机系列规划教材
COMPUTER COURSES FOR UNDERGRADUATE EDUCATION

C++程序设计

刘加海 祁亨年 陈 胤 主编

科学出版社

北 京

内 容 简 介

本书较为全面系统地讲述了 C++ 语言的基本概念、语法规则、编程方法以及初步应用,它是 C++ 程序设计的基础教材,全书共 9 章,主要内容涉及到面向对象的基本特征、C++ 程序设计的基本概念、类与对象、继承、运算符重载、虚拟函数与多态性、模板和异常处理、I/O 流与文件、用继承的方法应用 MFC 编写视窗程序。每章结尾都配备相当数量的习题。

本书内容通俗易懂、言简意赅、重点突出。内容循序渐进、深入浅出,以具体实例来分析和阐明 C++ 语言中的概念与原理,通过大量的实例示范,能使得学生在短时间内掌握 C++ 程序设计的原理及概念,并编写出高质量的 C++ 源程序。

本书适合作为高等院校计算机及相关专业的教材,也可作为广大电脑程序设计爱好者的自学参考书。

图书在版编目(CIP)数据

C++ 程序设计/刘加海、祁亨年,陈胤主编.—北京:科学出版社,2006

(面向 21 世纪高等院校计算机系列规划教材)

ISBN 7-03-017536-0

I. C… II. ①刘… ②祁… ③陈… III. C 语言—程序设计—高等学校—教材 IV. TP312

中国版本图书馆 CIP 数据核字(2006)第 070812 号

责任编辑:陈晓萍 许 进 / 责任校对:刘彦妮

责任印制:吕春珉 / 封面设计:飞天创意

科学出版社出版

北京东黄城根北街 16 号

邮政编码:100717

<http://www.sciencep.com>

双青印刷厂印刷

科学出版社发行

各地新华书店经销

*

2006 年 8 月第 一 版

开本:787×1092 1/16

2006 年 8 月第一次印刷

印张:18 1/4

印数:1—3 000

字数:432 000

定价:24.00 元

(如有印装质量问题,我社负责调换〈环伟〉)

销售部电话 010-62136131

编辑部电话 010-62138978-8003

前 言

本书是在多年教学实践的基础上编写而成的。在编写中总结了多所院校教师的教学经验,并针对学生在学习过程中遇到的要点与难点进行举例,充分反映了学生学习面向对象程序设计的思维过程。本书通俗易懂、由浅入深、概念清晰、语言简洁、例题丰富、内容全面、重点突出、适合自学、便于理解。

C++程序设计是计算机专业及理工科各专业非常重要的专业基础课,学习 C++不仅有利于深入理解面向对象程序设计的一些基本概念,而且非常有助于进一步学习其他的计算机语言,如 Visual C++程序设计等。

本书是 C++程序设计的基础教材。全书共分 9 章,较为全面系统地讲述了 C++语言的基本概念、语法规则、编程方法以及初步应用。

第 1 章主要讲述了有关面向对象的概念、特征。主要涉及到类、对象、继承、重载、多态性、事件驱动等概念。

第 2 章主要讲述了 C++的输入、输出、引用、函数重载等。

第 3 章是本书的重点,详细论述类、类对象、构造函数与析构函数,类的静态成员、类的友元及容器类。

第 4 章主要讲述了继承中的单继承与多继承、继承中的访问规则及继承中派生类的构造函数、虚基类等概念。

第 5 章主要讲述了运算符重载。

第 6 章主要讲述了动态联编与静态联编、虚函数与多态性、纯虚函数与抽象类等概念,它也是本课程的重点内容。

第 7 章主要讲述了模板和异常处理。

第 8 章主要讲述了 I/O 流与文件,详细介绍了用成员函数与格式化控制输入、输出以及不同格式文件的读写操作等。

第 9 章主要针对类、对象、继承、虚函数、事件驱动等面向对象的概念在视窗程序设计中进行综合应用,主要讲述了窗口程序、事件响应程序、菜单程序、文字显示程序、图像显示程序及一些常用控件程序的编写,学生通过学习,能很好地掌握面向对象程序设计思路与方法,初步了解视窗程序的设计。

本书在编写过程中从实际出发,兼顾最基本的理论知识,遵循深入浅出、通俗易懂的原则,概念清晰、逻辑性强,力求用大量的例子来阐明本课程的重点与难点,极大地减轻了读者学习 C++程序设计课程时的困难。本书不仅适合作为本科计算机专业、理工科各专业的 C++程序设计课程的教材,还适合作为计算机专业的专科生、高职生或夜大、电大等计算机专业的入门教材。

本书由浙江大学刘加海、陈胤,温州大学白宝钢,浙江大学宁波理工学院唐云廷,浙江林学院祁亨年,浙江科技学院罗朝盛,浙江工商大学徐云彪,浙江中医学院王海舜

等老师编写，全书由刘加海统稿。

由于作者水平有限，书中难免有疏漏和不妥之处，敬请读者批评指正。

作 者

2006 年 5 月于

浙江大学求是园

目 录

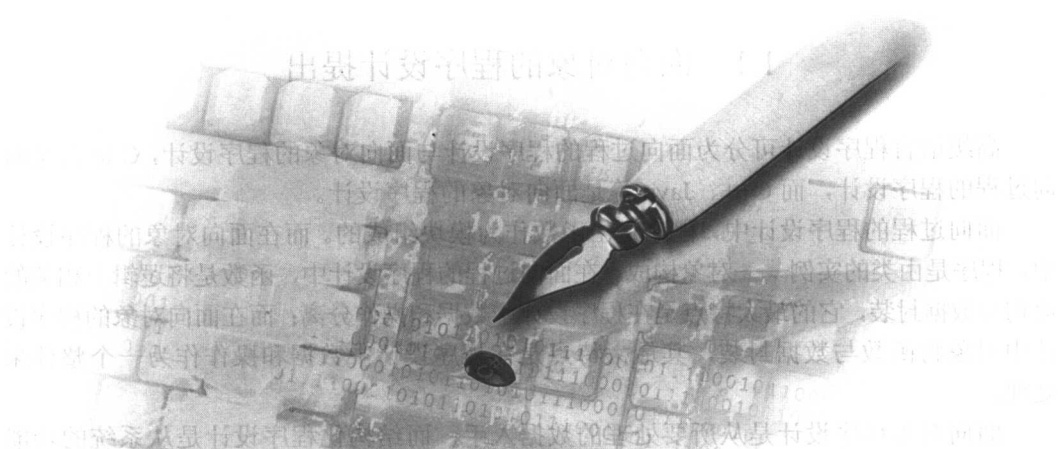
第 1 章 面向对象的特征概述.....	1
1.1 面向对象的程序设计提出.....	2
1.2 对象的概念.....	4
1.3 类的概念.....	6
1.4 消息.....	7
1.5 面向对象编程方法的基本特征.....	8
1.5.1 抽象.....	8
1.5.2 继承.....	8
1.5.3 封装.....	9
1.5.4 多态性.....	10
1.6 面向对象的软件设计思想.....	10
1.7 Visual C++ 6.0 环境下的程序调试.....	11
1.7.1 Visual C++ 6.0 源程序编辑、编译、连接、运行过程.....	11
1.7.2 打开已存在的文件.....	16
1.7.3 C++程序的调试.....	16
习题.....	21
第 2 章 C++的基本概念.....	22
2.1 引言.....	23
2.2 C++的单行注释.....	23
2.3 C++的输入 / 输出流.....	23
2.4 变量声明的位置.....	25
2.5 内联函数.....	26
2.6 默认的函数参数.....	28
2.7 引用参数.....	30
2.7.1 定义变量或对象的别名.....	30
2.7.2 定义函数的引用类型参数.....	31
2.7.3 定义函数的引用类型返回值.....	32
2.8 const 限定符.....	33
2.8.1 const 限定符的声明格式.....	33
2.8.2 带有指针的 const 限定符.....	34
2.9 域分辨操作符.....	35
2.10 运算符 new 和 delete.....	36
2.11 函数重载.....	39
习题.....	40

第3章 类与对象	45
3.1 类的基本概念	46
3.1.1 类的定义	46
3.1.2 类成员函数的定义方法	48
3.1.3 类对象的定义及引用方法	50
3.2 数据封装	54
3.2.1 类的私有成员	54
3.2.2 类的公有成员	55
3.3 类构造函数	56
3.3.1 构造函数的特点	56
3.3.2 构造函数的应用	58
3.3.3 缺省参数的构造函数	62
3.3.4 拷贝构造函数	63
3.4 类的析构函数	65
3.4.1 析构函数的特点	65
3.4.2 析构函数调用顺序举例	66
3.5 类静态成员	70
3.5.1 类的静态数据成员	70
3.5.2 类的静态成员函数	73
3.6 类的友元	76
3.6.1 友元函数	76
3.6.2 友元成员	78
3.6.3 友元类	79
3.7 容器类	82
3.8 类与结构	83
3.9 对象数组与对象指针	85
3.9.1 对象数组	85
3.9.2 指向类对象的指针	87
3.9.3 指向类成员的指针	88
3.9.4 this 指针	91
习题	92
第4章 继承	103
4.1 继承与派生	104
4.2 单继承	105
4.2.1 单继承	105
4.2.2 公有继承	106
4.2.3 私有继承	108
4.2.4 保护继承	110

4.3 派生类的构造函数和析构函数	111
4.3.1 派生类构造函数和析构函数的执行顺序	112
4.3.2 基类和派生类的赋值兼容规则	117
4.4 多继承	119
4.4.1 多继承中派生类的构造函数与析构函数	120
4.4.2 多继承中的二义性问题	123
4.5 虚基类	124
习题	128
第5章 运算符重载	134
5.1 运算符重载的基本概念	135
5.1.1 C++中可重载的运算符	135
5.1.2 运算符重载的定义形式	136
5.2 成员函数重载运算符	136
5.3 友元函数重载运算符	138
5.4 成员函数重载运算符与友元函数重载运算符比较	141
5.5 单目运算符的重载	142
5.6 赋值运算符的重载	146
5.7 二元运算符的重载	148
5.8 重载运算符	152
习题	153
第6章 虚拟函数与多态性	158
6.1 静态联编与动态联编	159
6.1.1 静态联编	159
6.1.2 动态联编	160
6.2 多态性的概念	161
6.3 虚拟函数	163
6.3.1 虚拟函数的定义	163
6.3.2 虚拟函数的调用	164
6.3.3 虚函数和重载函数的区别	168
6.4 构造函数和析构函数对虚函数的调用	170
6.5 虚拟函数的使用	171
6.5.1 虚拟函数的继承	171
6.5.2 多重继承与虚函数	174
6.5.3 派生类直接调用基类中的虚拟函数	175
6.6 虚拟函数的数据封装	175
6.7 纯虚函数与抽象类	177
6.7.1 纯虚函数	177
6.7.2 抽象类	177

6.8 多态性的异质单向链	180
习题	185
第7章 模板和异常处理	194
7.1 模板	195
7.1.1 函数模板	196
7.1.2 类模板	201
7.2 异常处理	208
7.2.1 异常处理的基本思想	209
7.2.2 异常处理的实现	209
7.2.3 异常生命周期	211
7.2.4 异常规格说明	212
7.2.5 异常处理中的构造与析构	212
习题	214
第8章 I/O 流与文件	215
8.1 C++流的概念	216
8.2 用 ios 类的成员函数实现格式化输入与输出	217
8.3 用操纵符实现格式化输入与输出	221
8.4 文件的操作	224
8.4.1 文件的操作过程	224
8.4.2 文件的打开与关闭	225
8.4.3 文件的操作方式	226
8.4.4 文本文件应用举例	227
8.4.5 二进制文件的操作	229
8.4.6 文件的随机读/写	230
8.5 用户自定义类型的输入/输出	232
8.5.1 输出运算符“<<”重载	233
8.5.2 输入运算符“>>”重载	234
习题	238
第9章 可视化程序设计初步	245
9.1 Windows 程序设计基本概念	246
9.1.1 Windows 消息	246
9.1.2 消息的种类	246
9.2 Windows 程序设计举例	248
9.2.1 CWinApp 类	249
9.2.2 CFrameWnd 类	249
9.2.3 应用程序举例 1	250
9.2.4 应用程序举例 2	252
9.2.5 应用程序举例 3	254

9.2.6 应用程序举例 4	259
9.2.7 应用程序举例 5	264
9.2.8 应用程序举例 6	267
习题	280
主要参考文献	281



第1章 面向对象的特征概述



本章要点

- ◇ 面向对象的特征
- ◇ 对象的概念
- ◇ 对象的属性与方法
- ◇ 类的概念
- ◇ Visual C++环境下程序调试方法



本章难点

- ◇ 对象的概念
- ◇ 类的概念
- ◇ 如何建立面向对象的思想

1.1 面向对象的程序设计提出

高级语言程序设计可分为面向过程的程序设计与面向对象的程序设计，C 语言是面向过程的程序设计，而 C++、Java 等是面向对象的程序设计。

面向过程的程序设计中，程序是由函数作为模块组成的。而在面向对象的程序设计中，程序是由类的实例——对象构成。在面向过程的程序设计中，函数是将逻辑上相关的语句与数据封装，它的最大特点是在程序设计中数据和操作分离；而在面向对象的程序设计中对象将函数与数据封装，其最大特点是在程序设计中数据和操作作为一个整体来处理。

面向对象程序设计是从所要处理的数据入手，而结构化程序设计是从系统的功能入手。

结构化程序设计从系统的功能入手，按照工程的标准和严格的规范将系统分解为若干功能模块，系统是实现模块功能的函数和过程的集合。由于用户的需求和软、硬件技术的不断发展变化，按照功能划分设计的系统模块必然是易变的和不稳定的。这样开发出来的模块可重用性不高。

面向对象程序设计从所处理的数据入手，以数据为中心而不是以服务（功能）为中心来描述系统。它把编程问题视为一个数据集合，数据相对于功能而言，具有更强的稳定性。面向对象程序设计技术的提出，主要是为了解决传统程序设计方法——结构化程序设计所不能解决的代码重用问题。

例如，求 a、b、c 三人的年收入。在程序中 i 表示月份，y 表示月收入，sum 表示年总收入，程序代码可写成：

```
#include<stdio.h>
struct penson
{
    int sum;
}a,b,c;
int yearsum ()
{
    int y,i,sum=0;
    for (i=1;i<=12;i++) {
        scanf ("%d",&y);
        sum=sum+y;}
    return sum;
}
void print (struct penson pe)
{
    printf ("The sum of year %d\n",pe.sum);
}

void main ()
{
    a.sum=yearsum ();
```

```

print (a) ;
b.sum=yearsum () ;
print (b) ;
c.sum=yearsum () ;
print (c) ;
}

```

在此程序中，程序设计是从程序的功能入手，把函数按功能划分，并且函数与三个对象 a、b、c 的数据相互独立。这个程序有三个方面的特点：①从功能入手；②代码或操作与数据分离；③代码的重用性差。

如果对以上程序进行修改，可写成：

```

#include<stdio.h>
struct penson
{
    int sum;
    void yearsum ()
    {
        int y,i,s=0;
        for (i=1;i<=3;i++) {
            scanf ("%d",&y) ;
            s=s+y;
        }
        sum=s;
    }
    void print ()
    {
        printf ("The sum of year %d\n",sum) ;
    }
};

void main ()
{
    penson a,b,c;
    a.yearsum () ;
    a.print () ;
    b.yearsum () ;
    b.print () ;
    c.yearsum () ;
    c.print () ;
}

```

在此程序中，程序设计从所处理的数据入手，把数据与对数据的操作视为一个数据集合，把数据与函数封装在一起，对象 a、b、c 共同具有这些数据与操作、它相对于功能而言，具有更强的稳定性。这个程序已基本具备面向对象程序设计的一些特征：

- 1) 从对象的属性与操作入手。
- 2) 操作与数据合并为一个不可分割的实体。
- 3) 实现了代码的重用性。

面向过程的程序设计中数据和操作分离，与自然界中具体的事物不一致。在面向对象的程序设计中，将动作和属性与每个实体相关联，即将代码与数据合并为一个不可分割的实体。例如，对人类的对象——一个具体人来说，人的特征或属性有：姓名、年龄、体重、身高；人具有的功能或行为有：走、跑、说、睡觉等。

- 对象：人，如图 1.1 所示。

特征（属性）：	姓名：张丽	年龄：18	体重：47	
行为（操作）：	走	跑	说	睡觉

图 1.1 人类的对象所具有的特征与行为

即对象具有的特征与行为，因而面向对象程序设计是一种围绕真实世界的概念来组织模型的程序设计方法，它采用对象来描述问题空间的实体。

1.2 对象的概念

在现实世界中，一切有形事物或无形事物、具体概念或抽象概念等都是对象。有形事物如一名教师、一件衣服、一本书、一个饭店、一座楼、一个学校等；无形事物如风、电磁场等；抽象概念如学校校规、企业规定等。不同的对象具有各自不同的特征和功能，例如，饭店具有饭店的特征和功能，学校具有学校的特征和功能。

一般认为，对象是现实世界中的一个实际存在的事物，它可以是有形的也可以是无形的，对象具有自己的静态特征与动态特征。静态特征反映了对象的属性，动态特征反映了对象的行为或方法。在程序设计领域，可以用“对象=数据+作用于这些数据上的操作”这一公式来表达。

例如物理对象：

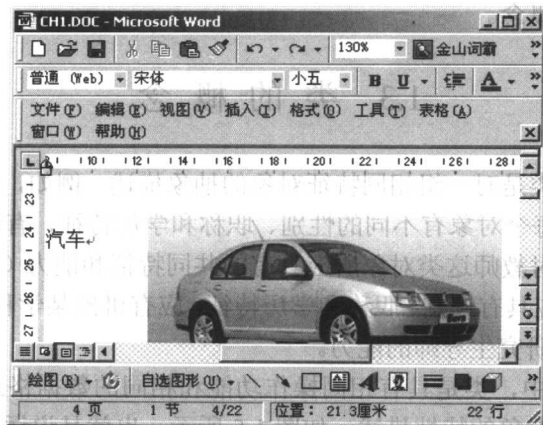
- 汽车，如图 1.2 所示。



特征（属性）：	名称：汽车	颜色：银灰	年份：2005	
行为（操作）：	启动	加速	行驶	停止

图 1.2 汽车对象的特征与行为

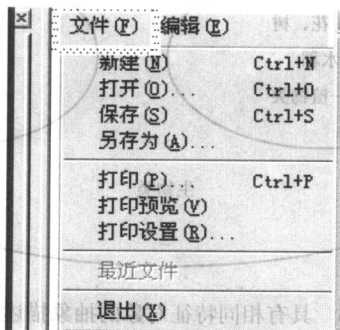
- 文档窗口，如图 1.3 所示。



特征(属性):	名称: 窗口	宽: 18	高: 7	
行为(操作):	最大化	最小化	关闭	编辑操作

图 1.3 文档对象的特征与行为

- 菜单，如图 1.4 所示。



特征(属性):	名称: 菜单	宽: 8	高: 16	位置: 30, 50
行为(操作):	打开操作	执行操作	关闭操作	

图 1.4 菜单对象的特征与行为

由此可见，现实世界中的对象具有如下特征：

- 1) 有一个名字用来惟一标识该对象。
- 2) 有一组状态用来描述其特征。
- 3) 有一组操作用来实现其功能。

在面向对象方法中，对象是既包括属性（特征、状态）数据，又包括作用于属性数据的一组操作（行为，方法、服务）的封装体。或者说，对象是属性数据和操作的封装体。在现实世界中，对象的能力通常称为操作或行为。在面向对象方法中，对象的能力通常称为方法或服务，对象的属性数据或状态数据通常称为属性。

在 C++ 语言中, 属性称作类数据成员, 行为称作类函数成员。对象是面向对象方法中最基本和最核心的概念。

1.3 类的概念

在现实世界中, 类是对一组相同特征对象的抽象描述。例如, 作为教师对象, 有张三、李四、王五等。每个对象有不同的性别、职称和学历特征, 有讲授不同学科课程的能力。而教师类则是对教师这类对象所应具有的共同特征和能力(功能)集合的抽象描述, 教师类的类对象应具有性别、职称和学历特征, 应有讲授某学科本科生课程的能力, 有些人还有做某学科研究生导师的能力。

在面向对象方法中, 类是具有相同操作功能和相同的数据格式或属性的对象的集合, 它是针对某一类对象的特性描述, 如图 1.5 所示, 也就是说运用抽象的方式描述这类对象。类为对象集合的抽象, 它规定了这些对象的公共属性和方法, 对象为类的一个实例。

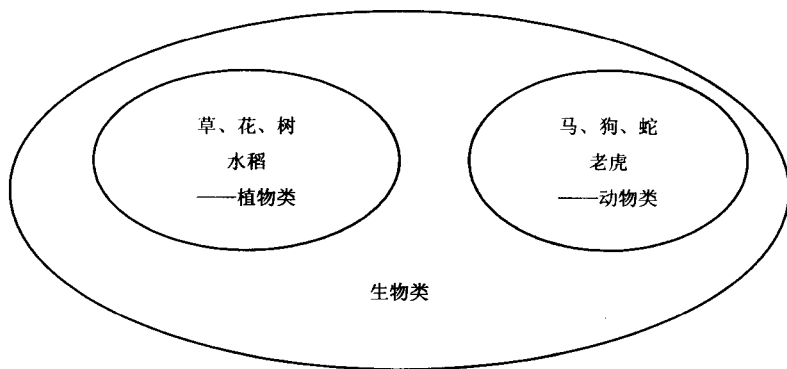


图 1.5 具有相同特征对象的抽象描述——类

例如, 你要在程序中描述一种动物: 猫。程序中一定有一种数据类型, 这种数据类型描述了猫的特性与方法, 比如: 猫有四只脚、一条尾巴、它的名字以及猫的叫声、猫抓老鼠等。当程序中要描述一只小花猫时, 可用这种猫的数据类型产生一只名叫小花猫的对象。

同理要描述汽车, 汽车有发动机、轮子、方向盘, 它能起动、行驶、停止等, 在程序设计中可以用抽象的方法定义有上述属性与方法的汽车数据类型, 再用这个数据类型产生一个汽车的对象。

苹果是一个类, 而放在桌上的那个苹果则是一个对象。对象和类的关系相当于一般的程序设计语言中变量和变量类型的关系。面向对象方法中程序设计的主体是类。类是对具有相同属性和服务的一个或多个对象的抽象描述, 因此类具有抽象性; 子类可以在继承父类所有属性和服务的基础上, 再增加自己特有的属性和服务, 或在某些操作中与父类有不同的服务, 因此类具有继承性。类的这种模块性和抽象性, 使面向对象方法

设计的模块较传统方法设计的模块有更高的重复使用率；类的这种继承性使面向对象方法设计的软件系统可维护性和系统升级能力大大提高。

在面向对象的程序设计中，首先定义一个类，类中包含对象的属性即类的数据成员、对象的方法即类的函数成员，类定义完成后，就可以定义类的对象。

1.4 消 息

在面向对象方法中，消息是向某个对象提出执行该对象具有的某个服务的申请。不同对象之间通过发送消息向对方提出服务要求，接受消息的对象主动完成指定服务。当一个消息发送给某个对象时，包含有要求接收对象去执行某个服务的信息。接收到消息的对象经过解释，然后予以响应，这种通信机制称作消息传递。发送消息的对象不需要知道接收消息的对象如何对消息予以响应。

通常，一个消息由以下三部分组成：

- 1) 接收消息的对象。
- 2) 消息名。
- 3) 零个或若干个参数。

例如，在 Windows 操作中，用鼠标右键点击某个文件或文件夹，在以点击点为左上角坐标出现一个弹出式快捷菜单。点击鼠标右键称为事件，文件或文件夹称为对象，发生事件后 Windows 产生一条消息 WM_RBUTTONDOWN，这条消息被对象文件或文件夹接受，文件或文件夹调用它的消息处理函数，产生快捷菜单，要注意快捷菜单中的命令项数、命令项的内容与所点击的文件或文件夹相关。

又如在 C++语言中，若已经定义了一个有具体圆心位置属性和半径属性及在显示器上绘制的图形的成员函数 Draw 的圆类，成员函数 Draw 要求给出一个显示颜色的参数，再用这个圆类产生圆对象 MyCircle，则语句：

```
MyCircle.Draw (Red) ;
```

是向对象 MyCircle 发送要求在显示器上绘制自身图形的消息 Draw，该消息携带的显示颜色的参数是 Red，即该消息是要求在显示器上，以该对象的圆心位置属性和半径属性为图形绘制数据，用红颜色绘制自身图形。

注意：① 同一对象可接收不同形式的多个消息，产生的响应各不相同。例如，在 Windows 系统中，存储的文件图标可以用鼠标左键单击、双击、鼠标右键单击，这些操作产生的消息不同，最后的结果也不相同。

② 相同形式的消息可以送给不同的对象，产生的响应各不相同。例如，在 Windows 系统中，鼠标右键单击不同的对象，相同的操作产生的消息相同，但结果不相同。