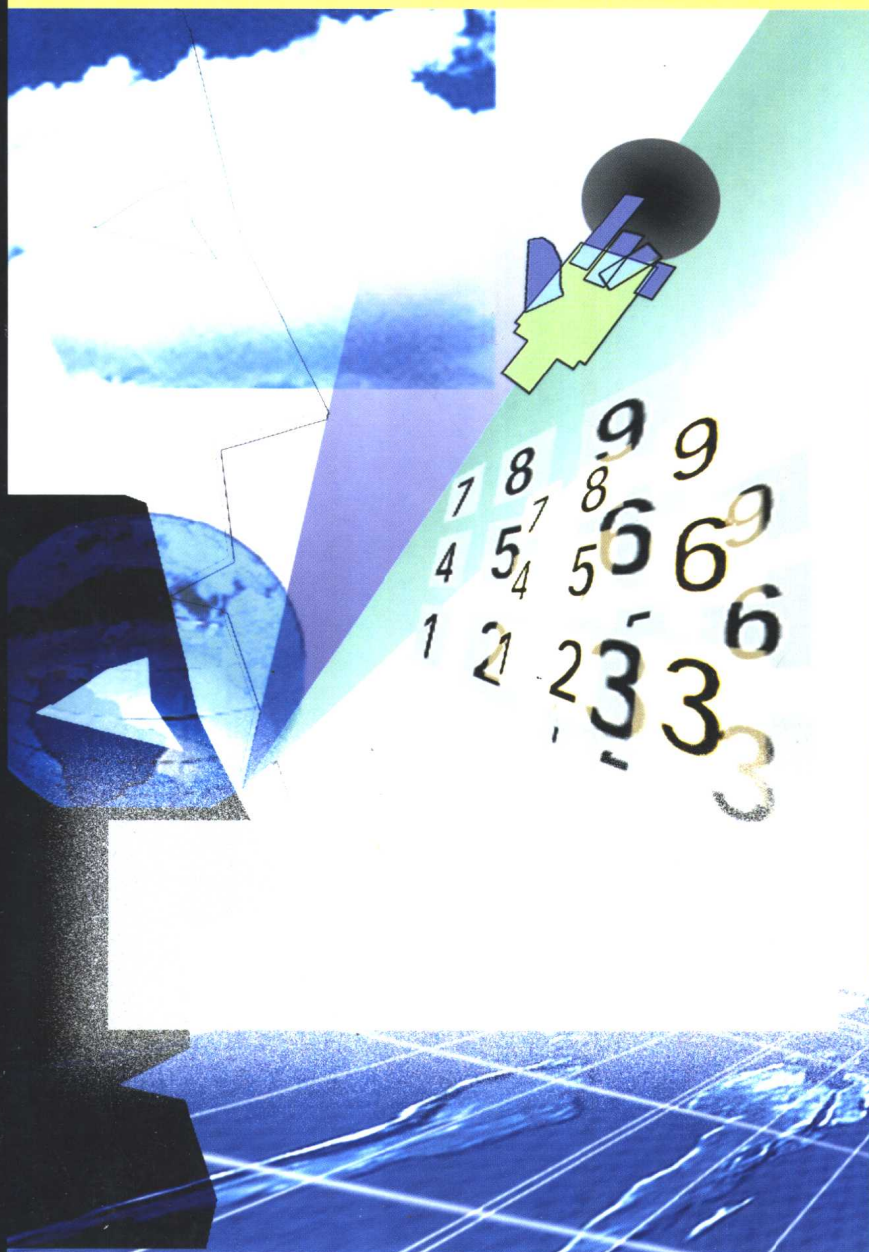


C++ 面向对象程序设计 习题解析与上机指导

陈维兴
陈
昕
编著



清华大学出版社

内 容 简 介

本书是高等学校电子信息类规划教材《C++面向对象程序设计教程》的配套用书,也适合单独作为学习 C++ 语言的学习辅导教材。书中内容分为三部分:第一部分“习题与解析”,提供了作者多年教学中积累、收集与验证的习题,每道题都给出了详细的解答与运行结果;第二部分“上机实验指导”给出了精心设计的 7 套上机实验题,每套实验都给出了实验目的、实验基本要求和实验步骤;第三部分给出了《C++面向对象程序设计教程》各章习题的参考答案。书后的附录中给出了上机实验的参考源程序。

本书可作为理工科大学学生学习 C++ 语言的参考书和自学辅导书。

版权所有,翻印必究。

本书封面贴有清华大学出版社激光防伪标签,无标签者不得销售。

图书在版编目(CIP)数据

C++面向对象程序设计习题解析与上机指导/陈维兴,陈昕编著. —北京:清华大学出版社,2003

ISBN 7-302-06406-7

I. C… II. ①陈… ②陈… III. C 语言—程序设计—高等学校—教学参考资料
IV. TP312

中国版本图书馆 CIP 数据核字(2003)第 018773 号

出 版 者: 清华大学出版社(北京清华大学学研大厦,邮编 100084)

<http://www.tup.com.cn>

责任编辑:柳 萍

版式设计:刘祎森

印 刷 者: 北京四季青印刷厂

发 行 者: 新华书店总店北京发行所

开 本: 787×1092 1/16 印张: 19.5 字数: 439 千字

版 次: 2003 年 5 月第 1 版 2003 年 5 月第 1 次印刷

书 号: ISBN 7-302-06406-7/TP·4828

印 数: 0001~5000

定 价: 24.00 元

前 言

C++ 面向对象程序设计习题解析与上机指导

对于学习 C++ 语言程序设计的学生或为提高面向对象程序设计能力的初学者来说,最重要的是提高自己的学习效果和实际编程能力。除了看书理解外,学习程序设计最关键的两个环节一个是多做习题多编程,另一个是多上机。为此,我们编写了这本习题解析与上机指导。本书从比较典型的习题解析着手,对一些基本的程序算法和规则进行详细的分析,以期帮助学习者掌握 C++ 语言程序设计的基本规则与编程规律,并能够将这些语言规则与技巧同实际问题相结合,编制出具有良好风格的应用程序,最终能够顺利地通过上机调试。为提高学习者的学习效果和实际应用能力,本书既注重编程基本素质的培养,也注重上机实验的训练,从两个方面促进学习者掌握 C++ 面向对象程序设计方法与相关基本概念,并能够熟练运用 C++ 程序设计语言进行工程开发。

本书的主要内容分为三部分:第一部分“习题与解析”,提供了多道 C++ 的典型实例;第二部分“上机实验指导”,介绍了 C++ 程序设计调试环境,并安排了 7 套精心设计的实验,每个实验题目都给出了详细的实验目的、实验基本要求和实验步骤,帮助学习者掌握 C++ 程序设计方法,并进一步加深对课程相关内容的理解与掌握;第三部分是“《C++ 面向对象程序设计教程》习题参考答案”,给出了《C++ 面向对象程序设计教程》每章习题的参考答案,各个习题可能有不止一种答案,本书提供了较为详尽的参考答案,这样有助于学习者自学;附录部分是“上机实验指导”中的各个实验的参考源程序,提供实验参考源程序的主要目的是给学习者一个参考和借鉴,帮助学习者掌握实验内容和理解具体实现步骤,以更好地掌握 C++ 面向对象程序设计基本方法和概念。本书中所有程序都经作者在 Visual C++ 6.0 或 Turbo C++ 3.0 上调试通过。

本书第一部分和第三部分由陈维兴编写,第二部分和附录部分由陈昕编写。全书由陈维兴主编并统稿。

在本书的编写与出版过程中得到了郑玉明教授、陈宝福教授、高明博士以及杨硕和杨光同学的帮助和支持,在此表示最诚挚的感谢。

本书内容是作者多年教学实践的总结,虽然得到了学生的肯定,但由于编者水平有限,错误和不当之处在所难免,在此恳请广大读者批评指正。

编 者

2003 年 2 月

目 录

C++ 面向对象程序设计习题解析与上机指导

第一部分 习题与解析

第1章 面向对象程序设计概述·····	3
第2章 C++ 语言基础 ·····	6
第3章 类和对象 ·····	23
第4章 继承与派生 ·····	77
第5章 多态性与虚函数·····	103
第6章 模板及其使用·····	153
第7章 基于 C++ 流库的输入与输出 ·····	172

第二部分 上机实验指导

C++ 程序设计语言实验环境介绍 ·····	203
实验1 C++ 程序设计基础练习·····	211
实验2 类与对象 ·····	215
实验3 对象数组与对象指针 ·····	217
实验4 派生类与继承 ·····	219
实验5 虚函数与多态性 ·····	221
实验6 函数模板与类模板 ·····	224
实验7 I/O 流类库 ·····	227

第三部分 《C++ 面向对象程序设计教程》习题参考答案

第1章 面向对象程序设计概述·····	233
第2章 C++ 概述 ·····	236
第3章 类和对象·····	241
第4章 派生类与继承·····	246

第 5 章 多态性..... 253

第 6 章 模板..... 262

第 7 章 C++ 的 I/O 流类库 267

附录 上机实验题参考源程序

实验 1 的参考源程序..... 277

实验 2 的参考源程序..... 280

实验 3 的参考源程序..... 283

实验 4 的参考源程序..... 290

实验 5 的参考源程序..... 295

实验 6 的参考源程序..... 301

实验 7 的参考源程序..... 304

第一部分

习题与解析

第 1 章 面向对象程序设计概述

【1-1】 什么是类和对象？在现实世界中，对象有哪些特点？

解

类是一组具有共同的属性特征和行为特征的对象抽象。对象就是系统中用来描述客观事物的一个实体，它是构成系统的一个基本单位，由一组属性和一组行为构成。类和对象之间的关系是抽象和具体的关系，类是对多个对象进行抽象的结果，对象是类的具体实现。

对象是现实世界中的一个实体，其具有以下一些特点：

- (1) 每一个对象必须有一个名字以区别于其他对象；
- (2) 需要用属性来描述它的某些特征；
- (3) 有一组操作，每一个操作决定了对象的一种行为。

【1-2】 什么是消息？消息具有什么性质？

解

在面向对象程序设计中，一个对象向另一个对象发出的请求被称为“消息”。消息是一个对象要求另一个对象执行某个功能操作的规格说明，通过消息传递才能完成对象之间的相互请求或相互协作。

消息具有三个性质：

- (1) 同一个对象可以接收不同形式的多个消息，做出不同的响应；
- (2) 相同形式的消息可以传递给不同的对象，所做出的响应可以是不同的；
- (3) 消息的发送可以不考虑具体的接受者，对象可以响应消息，也可以不响应。

【1-3】 什么是封装和抽象？请举例说明。

解

所谓数据的封装，就是将数据结构和作用于数据结构上的操作组成一个实体，数据的表示方式和对数据的操作细节被隐藏起来，用户通过操作接口对数据进行操作。对于用户来说，只知道如何通过操作接口对该数据进行操作，而并不知道是如何操作的，也不知道数据是如何表示的，这就是数据的封装。对我们经常使用的操作系统，如 Windows XP，每当我们通过可视化的用户界面进行操作时，其实操作系统在后台做了大量的工作。比如我们进入 D 盘，打开一个 RM 格式的电影，进行娱乐。这时计算机需要控制硬盘机

械手进入指定盘道进行读,然后写到内存,通过 CPU 等等。我们不需要详细了解,就是因为有了封装的存在。

所谓抽象则是通过对数据实例的分析,抽取其共同性质的结果。我们在定义一个类时,只表示那些主要特征,而忽略次要的。数据的封装隐藏了抽象的内部实现细节,封装时将数据抽象的外部接口与内部实现细节清楚地分离开,抽象和封装是互补的,好的抽象有利于封装,封装的实现则帮助维护抽象的完整性。

【1-4】 什么是继承性? 请举例说明。

解

继承所表达的是对象类之间的相关关系,这种关系使得某类对象可以继承另外一类对象的特征和能力。现实生活中,继承是很普遍和容易理解的。例如我们继承了父母的一些特征,如种族、血型、眼睛的颜色等,父母是我们所具有的属性的基础。

图 1 所示是一个继承的典型例子:汽车继承的层次。

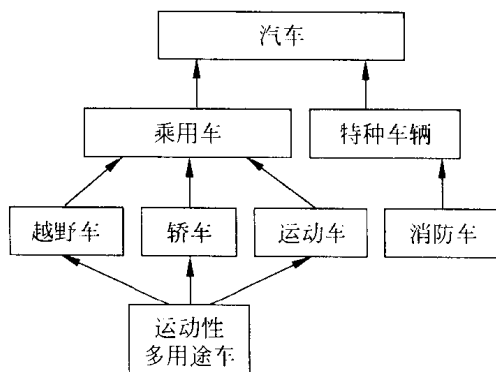


图 1 汽车继承的层次图

【1-5】 具有继承关系的类之间具有什么特征?

解

若类之间具有继承关系,则它们之间具有以下几个特征:

- (1) 类间具有共享特征,包括数据和程序代码的共享,如每种类型的汽车都有发动机、驱动系统等;
- (2) 类间具有差别或新增部分,包括非共享的数据和程序代码,比如每一层次都增加了一些新的功能;
- (3) 类间具有层次结构。

【1-6】 继承有哪几种分类方法?

解

继承有两种分类方法,一种从继承源上划分,另一种从继承内容上划分。

从继承源上划分,继承分为单继承和多继承。上面的例子中,消防车对特种车辆的继承就是单继承。而运动性多用途车(SUV)对越野车、轿车和运动车的继承就是多继承。

从继承内容上,分为取代继承、包含继承、受限继承和特化继承。取代继承:比如超级跑车从运动车上继承到了所有特点,凡是可以使用运动车的地方当然对超级跑车也没有

问题。包含继承:无论什么类型的车辆,它们都继承了汽车的所有特征,任何一种车辆都是一辆“汽车”,这就属于包含继承。受限继承:尽管我们继承了父母的一些特征,但是并不意味着我们就具有了父母的所有特长、性格以及爱好。特化继承:例如运动性多用途车是一类特殊的汽车,比一般的汽车具有更多的功能,使用范围更广,这就属于特化继承。

【1-7】 什么是多态性?请举例说明。

解

多态性也是面向对象程序的重要特征。它是指不同的对象收到相同的消息时产生不同的行为方式。例如我们同样双击 Windows 系统桌面上的图标时,有的是打开媒体播放器,有的是打开 D 盘,而有的是关机。

利用多态性,用户只需发送一般形式的消息,而将所有的实现留给接收消息的对象。对象根据所收到的消息而做出相应的动作。C++ 语言支持两种多态性,即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来实现的;而运行时的多态性是通过虚函数来实现的。

【1-8】 何为函数重载和运算符重载?为什么要使用重载?

解

重载一般包括函数重载和运算符重载。函数重载是指一个标识符可同时用于为多个函数命名,而运算符重载是指一个运算符可同时用于多种运算。也就是说,相同名字的函数或运算符在不同的场合可以表现出不同的行为。

为什么要使用重载?使用重载的目的是为了更好地表达行为共享,这种行为共享就像将相似的操作划分在一起。使用重载可以使程序员在只知道操作的一般含义,而不知道操作的具体细节的情况下能正确地对某个对象使用一个操作。

另外,使用重载的直接益处是减少了程序员记忆操作名字的负担。

第 2 章 C++ 语言基础

【2-1】 由用户输入一个整数,要求在屏幕上依次列出 10 个数及其二次方、三次方。

解

实现本题功能的程序如下:

```
#include<iostream. h>
main()
{
    int n;
    cout<<"Enter a number: ";
    cin>>n;
    for (int i=n; i<=n+9; i++)
    {
        cout<<i;
        cout<<" "<<long(i) * i<<" "<<long(i) * i * i;
        cout<<endl;
    }
    return 0;
}
```

本程序的运行结果如下:

```
Enter a number: 11
11      121      1331
12      144      1728
13      169      2197
14      196      1744
15      225      3375
16      156      4096
17      289      4913
18      324      5832
19      361      6859
20      400      8000
```

说明:

(1) 为尽可能容纳数值的三次方,程序中使用了 long 型整数,例如:

```
cout<<long(i)*i*i;
```

或

```
cout<<long(i*i*i);
```

(2) long 长整型数容纳的数值范围为：-2147483648 到 2147483647。

【2-2】 由用户从键盘输入一个数,计算该数的阶乘。

解

实现本题功能的程序如下:

```
#include <iostream.h>
main()
{
    int n;
    unsigned long factor=1;
    cout<<"Enter a number: ";
    cin>> n;
    for (int i=n;i>1;i--)
        factor=factor*i;
    cout<<n<<"!="<<factor<<endl;
    return 0;
}
```

本程序的运行结果如下:

```
Enter a number: 6
6!=720
Enter a number: 12
12!=479001600
Enter a number: 8
8!=40320
```

注意:

unsigned long 型整数可容纳的数值范围为 0 到 4294967295,故用户最多只能计算到 12!。

【2-3】 由用户从键盘输入一个数 n,计算第 n 个 fibonacci 数。

解

实现本题功能的程序如下:

```
#include <iostream.h>
main()
{
    int n;
    unsigned long fib1=0, fib2=1;
    cout<<"Enter a number: ";
    cin>>n;
```

```

// 计算第 n 个 fibonacci 数
for (int i=2; i<=n;i++)
{
    long x = fib1;
    fib1 = fib2;
    fib2 = x+fib1;
}
cout<<"fibonacci: "<<fib2<<endl;
return 0;
}

```

本程序的运行结果如下：

```

Enter a number: 8
fibonacci: 21
Enter a number: 14
fibonacci: 377
Enter a number: 16
fibonacci: 987

```

说明：

fibonacci 数列为：1,1,2,3,5,8,13,32,34,55...

定义：

$\text{fib}(n)=1$ (当 $n=1$ 或 $n=2$ 时)

$\text{fib}(n)=\text{fib}(n-1)+\text{fib}(n-2)$ (当 $n>2$ 时)

【2-4】 要求用户从键盘输入一段文字，由程序返回该段文字所包含的字符和单词个数。单词定义为以空白符隔开的文本字符串。

解

实现本题功能的程序如下：

```

#include <iostream. h>
#include <conio. h>
main()
{
    int char_count=0;
    int word_count=0;
    char ch;
    int num=0;
    cout<<"Please enter a string: \n";
    while ((ch=getche())!='\r')
    {
        char_count++;
        if (ch==' '||ch=='\t')
        { num++;}
        else if (num>0)
        {

```

```

        word_count++;
        num=0;
    }
}
cout << " \n This string contains " << char_count
    << " character " << endl;
cout << " This string consists of " << word_count-1
    << " words " << endl;
return 0;
}

```

本程序的运行结果如下：

```

Please enter a string:
This is a test.
This string contains 15 character
This string consists of 4 words

```

说明：

- (1) 键盘上输入的任何字符(包括退格键)均列入字符总数内。
- (2) 按下 Enter 键时 getch()函数返回值为字符'\r',退出循环。

【2-5】 从键盘上输入一个长整型数,判别该数是否为素数。

解

实现本题功能的程序如下：

```

#include <iostream. h>
#include <process. h>
void main()
{
    long x;
    cout<<"Please enter a number: \n";
    cin>>x;
    for (long n=2;n<x/2; n++)
        if (x % n==0)
        {
            cout <<"It's not prime, can be divided by "
                <<n<<endl;
            exit(0);
        }
    cout<<x<<" is a prime !" <<endl;
}

```

本程序的运行结果如下：

```

Please enter a number:
115
It's not a prime,can be divided by 5

```

```
Please enter a number;
```

```
11113
```

```
11113 is a prime !
```

说明:

(1) 素数的定义是: 除了 1 及本身外, 不能被其他数整除的数。

(2) 判断某个整数是否为素数, 可将该整数自 2 除起, 如被整除表示该数不是素数。

为提高程序的运行效率, 除数的最大值应小于该数的 $1/2$ 。

【2-6】 利用交换式气泡排序方法, 由小到大显示出数组元素的值。

解

实现本题功能的程序如下:

```
#include <iostream.h>
#include <iomanip.h>
const MAXSIZE=20;
void sort(int list[], int size);
void main()
{
    static int list[MAXSIZE];
    int size=0;
    do
    {
        cout<<"Type number <-999 to exit>: ";
        cin>>list[size];
    }
    while (list[size++]!=-999);
    sort(list, size-1);
    cout<<"After sorting ... \n";
    for (int dex=0; dex<size-1; dex++)
        cout<<setw(5)<< list[dex];
}

void sort(int list[], int size)
{
    int x;
    for (int y=0; y<size-1; y++)
        for (x=y+1; x<size; x++)
            if (list[y]>list[x])
            {
                int temp=list[x];
                list[x]=list[y];
                list[y]=temp;
            }
}
```

本程序的运行结果如下:

```
Type number <-999 to exit>: 134
Type number <-999 to exit>: 120
Type number <-999 to exit>: -334
Type number <-999 to exit>: 55
Type number <-999 to exit>: 77
Type number <-999 to exit>: 567
Type number <-999 to exit>: -999
After sorting ...
-334 55 77 120 134 567
```

说明:

气泡排序(bubble sorting)是一种用来将数据重新排列的最简单、但效率较低的方法。交换式气泡排序使用双层循环,外层循环由 list[0]开始依次和内层循环 list[1]…list[size-1]比较,较小的数存放在 list[0]处。接着外层循环的 list[1]和内层循环 list[2]…list[size-1]比较,较小的数存放在 list[1]中。就这样依次比较,最后,外层循环的list[size-2]和内层循环的 list[size-1]比较,较小的数存放在 list[size-2]中。

本例设定最多可以容纳 20 个元素的整型数组。由用户自键盘上输入数值,输入 -999 后结束。

【2-7】 试编写一程序,其中定义一个联合类型和一个结构类型,这两个类型有相同的成员,它们是整型变量 i 和双精度型变量 d。用这两个类型各声明一个变量,随后对这两个变量赋值,并分别打印出这两种类型变量的值。

解

实现本题功能的程序如下:

```
#include <iostream. h>
void main()
{
    int size;
    union u_tag
    {
        int i;
        double d;
    } u={88};
    struct s_tag
    {
        int i;
        double d;
    } s = {66,1.234};
    size = sizeof(union u_tag);
    cout<<"sizeof(union u_tag)="<<size<<endl;
    u.i = 100;
    cout<<"u.i="<<u.i<<endl;
    u.d = 1.2345;
    cout<<"u.d="<<u.d<<endl;
```

```

size = sizeof(u.d);
cout<<"sizeof(u.d)="<<size<<endl;
cout<<"s.i="<<s.i<<endl;
cout<<"s.d="<<s.d<<endl;
size = sizeof(struct s_tag);
cout<<"sizeof(struct s_tag)="<<size<<endl;
}

```

本程序的执行结果如下：

```

sizeof(union u_tag)=8
u.i=100
u.d=1.2345
sizeof(u.d)=8
s.i=66
s.d=1.234
sizeof(struct s_tag)=16

```

说明：

- (1) 本例说明联合变量和结构变量的应用方式相同,但使用方法不同。
- (2) 联合类型和联合变量的空间大小与其成员中占空间最大的成员相同。例如,本例中的联合类型 `u_tag` 的空间大小为 8 个字节,与其成员 `d` 的大小相同。

【2-8】 编写一程序,其中声明了一个枚举类型,并定义了两个枚举变量,考查对枚举变量的操作。

解

实现本题功能的程序如下：

```

#include <iostream.h>
void main()
{
    enum color{
        RED=1,
        GREEN=6,
        BLUE=9
    };
    enum color v1=RED,v2;
    cout<<"RED="<<RED<<endl;
    cout<<"GREEN="<<GREEN<<endl;
    cout<<"BLUE="<<BLUE<<endl;
    cout<<"v1="<<v1<<endl;
    v2=GREEN;
    cout<<"(v1<v2)="<<(v1<v2)<<endl;
}

```

程序的执行结果如下：

```
RED=1
```