C++程序设计

和克智　编著

西安交通大学出版社

C++程序设计

和克智　编著

西安交通大学出版社

内 容 简 介

C++是80年代出现的一种新型的程序设计语言。它除了保持C语言的简洁高效等特点外，还支持面向对象的程序设计。

本书共分13章和两个附录，全面、系统地介绍了C++语言的语言要素及C++程序设计的方法与步骤。特别是在面向对象的程序设计方面，书中通过逐步完善几个较大型的例子，从应用角度予以详细的介绍，以求读者能迅速地掌握这一新型的程序设计方法。

本书是作者集多年教学科研和实际应用的心得体会，精心撰写而成的。书中的许多内容是其它资料中所没有提到而在实用中又经常会遇到的。相信读者能从本书中得到较大的收获。

本书可用作大专院校C++语言教材，计算机软件开发和应用人员的参考资料，尤其适合自学C++语言的读者使用。

C++程序设计

[illegible]智 编著

责任编辑 白居宪

*

西安交通大学出版社出版

(西安市咸宁西路28号 邮政编码710049)

西安理工大学印刷厂印装

陕西省新华书店经销

*

开本：787×1092 1/16 印张：19 字数：456千字

1995年7月第1版 1995年7月第1次印刷

印数：1——8 000

ISBN7-5605-0764-6/TP·104 定价：18.00元

前　言

目前风靡计算机世界的面向对象的程序设计方法是软件开发上的一场革命。C++语言是一个优秀的面向对象的程序设计语言，它保持了C语言的简洁、高效和接近汇编语言的特点，消除了C语言中的一些不安全因素，更重要的是它将高科技研究中的面向对象的思维方式引进到程序设计语言中，增加了类这种语言成分，以支持面向对象的程序设计。这不仅使得程序的可靠性、数据的安全性得到了保证，而且提高了程序代码的复用性，从而极大地改善了程序设计的效率。

C++语言除支持面向对象的程序设计外，也同时支持传统的非面向对象的程序设计。实际上，由于C++语言的变量说明非常灵活，输入/输出操作极为方便，并提供有引用数据类型，因此，用C++语言书写非面向对象的程序时也会比传统程序设计语言方便得多。

本书共分13章和两个附录。全书以翔实的内容系统而全面地介绍了C++语言的各个语言成分及其特性，以及用C++语言进行程序设计的基本方法与设计要点。第1章介绍C++语言的发展及其特点、面向对象程序设计的基本概念、C++程序的基本结构、基本的输入/输出方法和开发C++程序的基本步骤。第2章介绍C++语言的词法符号、基本数据类型、变量、运算符和表达式。第3章介绍C++语言的程序控制语句。第4章介绍C++语言的函数、作用域和存储类以及编译预处理指令。其中函数的重载从一个方面体现了面向对象程序设计的多态性。本章内容为C++语言的重点之一。第5章介绍C++语言的数组类型和字符串。第6章介绍C++语言的指针、引用和动态内存分配。本章内容也是C++语言的重点与难点之一。第7章介绍C++语言的结构、联合和枚举型数据类型。第8章介绍C++语言的类类型和对象。从本章起所介绍的内容均与面向对象的程序设计密切相关。类体现了面向对象程序设计语言的封装性，因此本章的内容为C++语言的重点之一。第9章介绍C++语言中类的构造函数和析构函数。第10章介绍C++语言的继承和派生类，它们体现了面向对象程序设计的继承性。本章亦为C++语言的重点之一。第11章介绍C++语言中的友元、虚函数、类的静态成员以及const和volatile成员。其中，虚函数体现了面向对象程序设计的运行时的多态性。第12章介绍C++语言的运算符重载。运算符重载从另一个方面体现了面向对象程序设计的多态性。本章内容是C++语言的又一重点。第13章介绍C++语言的输入/输出流类，包括C++的基本I/O流类、文件流类和RAM流类。本章内容是C++语言的又一个重点和难点。附录A介绍了C++语言的关键字asm及其简单用法。附录B为ASCII码表。

书中每章后面都附有习题。应当说明的是：这些习题不仅是用来使读者巩固各章的内容，它们同时又是相应章节内容的补充。由于篇幅所限，一些较大型的例子不便于在正文中给出，而是放在习题中由读者对它们进一步地完善。这一现象在第8章以后尤为明显。所以，希望读者尽量完成所有的习题，以便全面地掌握C++语言。

本书是作者集多年对本科生、研究生的C语言和C++语言的教学经验，并参阅有关中外资料的前提下撰写而成的。对学习C++语言时常犯的错误和易混淆的概念进行了详细地介

绍与提示。为使读者能更好地掌握C++语言，作者根据自己多年在科研和教学中使用C++语言的心得体会，精心编写了书中所有的例题并全部上机通过，另外还编写了几个较大型的程序，其内容贯穿C++语言的主要语言成分，并在相应章节逐步出现、逐步完善，以期为读者掌握向面对象的程序设计方法提供一个思路。

作者力求本书具有通俗性和实用性。尽量避免过多的理论描述，而用较多的篇幅介绍实用方法。本书中的许多内容出自于作者的经验和体会，是其它资料所没有提到而在实用中又常常遇到的。作者将这些内容奉献给读者，期望读者通过本书能很快地掌握C++语言，打消"C++语言难学"或"C++语言好学难用"的不正确看法。

本书将C++语言作为一个独立的程序设计语言，而不是作为C语言的扩展来进行介绍的，因此，不要求读者先行掌握C语言。

本书可供大专院校作C++语言教材使用，也可以作为从事计算机软件开发和应用的人员的重要参考资料，尤其适合作为广大计算机爱好者学习C++语言的自学材料。

对于本书存在的错误和不足之处，作者殷切希望广大读者批评指正。

和克智

一九九五年四月于西安理工大学

目　录

第1章 绪论

本章介绍C++的起源、发展及其特点；面向对象程序设计的基本概念；C++程序的基本结构；C++的基本输入/输出功能。

1.1 C++语言的发展和特点

自1946年第一台电子计算机ENIAC问世已来，随着计算机应用领域的迅速扩大，硬件和软件都取得了高速的发展。作为计算机软件的基础，程序设计语言也得到不断的充实和完善。功能全面、使用便利的程序设计语言相继问世。

60年代，Martin Richards为计算机软件人员在开发系统软件时作为记述语言使用而开发了BCPL语言。1970年，Ken Thompson在继承BCPL的许多优点的基础上发明了比较实用的B语言。美国DEC公司的PDP-7型机中的UNIX操作系统就是用B语言记述和开发的。到了1972年，在美国贝尔研究所进行的对小型机PDP-11 UNIX操作系统的开发工作中，Dennis Ritchie和Brian Kernighan对B语言作了进一步的充实和完善，推出了更加通用的C语言。C语言具有以下的主要特点：

C语言是结构化程序设计语言 结构化程序设计要求程序的逻辑结构只能由顺序、选择和循环三种基本结构组成。C语言提供了编写结构化程序所需要的语句，十分便于采用自顶向下、逐步求精的结构化程序设计技术。

C语言是模块化程序设计语言 C语言程序的函数结构，非常便于把一个程序的整体分割成若干相对独立的功能模块，并且为程序模块间的相互调用以及数据传递提供了便利。

C语言程序高效、灵活、功能强 C语言是为了能够完成系统程序设计的要求而开发的，具有很强的表达能力，能够用于描述系统软件各方面的特性。因此，用C语言编写的程序表述灵活，功能强大。另外，C语言编写的程序生成的代码质量极优，所以C程序具有很高的运行效率。

C语言可以部分地取代汇编语言 C语言比较靠近硬件与系统，可以像汇编语言那样直接访问硬件。比如，C语言的第一个应用就是开发UNIX操作系统，而该系统软件仅用了很少的汇编指令，绝大部分代码是用C语言写成的。

C语言具有较高的移植性 在C语言中，没有依赖于硬件的输入/输出语句。在程序中，输入/输出操作是通过调用由系统提供的、独立于C语言的库函数来完成的。因此，C语言本身并不依赖于计算机的硬件系统，从而便于在硬件结构不同的机器间实现程序的移植。

C语言具有很强的数据处理能力 C语言提供了丰富的运算符，除了可以对数据进行四则运算和逻辑运算外，还可以实现以二进制位(bit)为单位的诸多运算。另外，C语言还提供了丰富的数据类型。因此，C语言具有很强的数据处理能力。

由于C语言具有以上众多的特点以及UNIX的成功和广泛应用，使得C语言迅速得到普及，成为一种广泛使用的程序设计语言。目前的各种机型和各种操作系统上都运行有C编译器。

然而，C语言也存在着一些缺陷，比如：类型检查机制相对较弱；缺少支持代码重用的语言结构；不适合开发大型程序等。随着软件工程规模的扩大，C语言的这些缺陷逐渐显露出来。比如，当程序超过50 000行、开发人数达数十人时，系统维护工作量将变得相当大，而且系统的整体性难以保证。

为了克服C语言存在的缺点，并保持C语言简洁、高效和接近汇编语言的特点，1980年，贝尔实验室的Bjarne Stroustrup博士及其同事开始对C语言进行改进和扩充，把Simula 67中类的概念引入到C中，并将改进后的C语言称为"带类的C"(C with class)。1983年夏，由Rick Maseitti提议正式命名为"C++"，并于同年7月首次对外发表。在此之后，Stroustrup博士与他的同事们在使用该语言所积累的经验的基础上，于1985年和1989年进行了两次修订，先后引进了运算符重载、引用、虚函数等许多特性，并使之更加精炼，于1989年底推出最新的AT&T C++ 2.0版本。现在，AT&T C++ 2.0事实上已经成为各种版本C++语言的标准。

C++对C语言的扩充和改进是具有革命性的，这是由于C++支持面向对象的程序设计。同时，C++又是C语言的一个超集，这就使得许多C代码不经修改就可以为C++所用。另外，用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。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。

C++已经被应用于程序设计的众多应用领域，它尤其适合用于中等和大型的程序开发项目。有报告表明，C++已应用于C曾经使用过的所有场合，且其效果要比C语言好得多。从开发时间、费用到形成的软件的可重用性、可扩充性、可维护性和可靠性等方面，都显示出C++的优越性。

1.2 面向对象的程序设计

面向对象的程序设计(Object-Oriented Programing，简称OOP)是继承和发展了结构化程序设计而产生的一种新的程序设计思想，它是一种通过摹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)进行软件开发的思想体系。面向对象的程序设计语言必须具有四种对象化属性：抽象性、封装性、继承性和多态性。

1.2.1 抽象和封装——对象

对象(Object)是OOP最重要的性质之一。从概念上讲，对象就是既含数据又含对数据进行操作的代码(叫做方法 Method)的一个逻辑实体。在对象中，有些数据和代码是为该对象所特有的，亦即不能为对象之外的任何方法直接访问。这样，可以有效地防止程序中其他不相关部分无意地修改或不正确地使用这些数据和代码。

对象是类(Class)的实例(Instance)。类用于描述对象的群体特性，它同时又是进行抽象和封装的基石。

抽象(Abstraction)是指将具体事物一般化的过程。对具有特定属性及行为特征的对象进

行概括，从中提取这一类对象的共性，并从通用性的角度描述共有的属性及行为特征。抽象包括两方面的内容：一是数据抽象，即描述某类对象的公共属性；二是代码抽象，即描述某类对象的行为特征。通过类，可以很方便地实现数据抽象和代码抽象，这种结合方式就是所谓的封装(Encapsulation)机制。抽象和封装可以提高软件的模块化程度，增强代码的重用性。

访问一个方法的过程称为向一个对象发送一个消息(Message)。对象的工作是靠消息来激发的，对象之间也是通过消息发生联系的，即请求其他对象做什么或响应其他对象的请求是通过发送和接收消息来实现的。

1.2.2 派生和继承

派生(Deriving)是指由基本类导出子类的过程，该子类就叫做派生类。派生是分层次等级进行的。继承(Inheritance)是指一对象获取另一对象之性质的过程。它与按层次分类的思想是面向对象的主要性质。在有类这一概念之前，必须针对每个对象定义其所有性质；由于使用了类，就只需在派生类中定义对象的区别于基本类中其他对象之性质。由于派生类的存在，那些与基本类共享的性质就能很方便地继承下来。

1.2.3 多态性

同一个消息为不同的对象所接收时，可导致完全不同的行为，这种现象称为多态性(Polymophism)。多态性的重要性在于允许一个类体系的对象各自以不同的方式响应同一消息，即所谓的“同一接口，多种方法”。多态性机制不仅提高了程序设计的灵活性，而且大大减轻了类体系使用者的记忆负担。

1.3 C++程序的结构

学习任何一种程序设计语言，首先应了解用该语言所编写的程序的基本结构，这里，通过一个简单的C++程序来说明C++程序的结构及其特点。

例1.1 一个简单的C++程序

```
// EX1_1.CPP
/* This program demonstrats the construction of C++
   program file. */
#include <iostream.h>

int main()
{
    cout << "i = ";          // Display a prompt
    int i;                   // Declear variable i
    cin >> i;                /* Get input for i */
    cout << Your number is " << i << '\n';
    return 0;
}
```

当运行这个程序时，屏幕上将显示：

i =

并等待用户输入一个整数。设用户的输入为5，则屏幕上将显示：

Your number is 5

并返回操作系统。

程序中第一至三行是C++语言中的注释语句。与汇编语言相比，C++语言所编写的程序的可读性要好得多，然而，它和人类的自然语言仍有很大的距离。为了增强程序的可读性，在书写源程序时，应养成用注释来说明程序的功能、语句的作用的良好习惯。C++编译器在编译一个程序时，将跳过注释语句，不对它进行处理。因此，无论源程序中有多少注释，均不会增加可执行文件的长度。

C++语言提供了两种注释语句形式：一种是由符号"//"引出、"回车"结束的单行注释语句；一种是由符号对"/*"和"*/"括起来的多行注释语句。前者主要用于较短小、内容不超过一行的注释，而后者主要用于较长的多行注释。当然，可以根据个人的爱好与习惯只使用其中一种注释形式或将两种注释形式混合使用，如例1.1中第八至十行所示。

程序中第四行是一条编译指令。关于编译指令的祥细内容将在4.10节中介绍。这里，暂时理解为：为了能进行输入/输出操作，程序中必须有这一行。

程序中以 main 开始的部分定义了一个函数，该函数规定了程序的功能。main 是函数名，其后紧跟一对圆括号。所有的C++程序必须有且仅能有一个 main 函数，通常称该函数为主函数。一个C++程序总是从主函数中的第一条语句开始执行，在正常情况下，总是在执行完主函数中的最后一条语句后结束整个程序并返回操作系统。

一个C++函数中的所有内容均被括在一对花括号("{"和"}")中，其中，左花括号表示"该函数从这里开始"，而右花括号表示"该函数到这里结束"。被括起来的内容叫作函数体。函数体由一系列C++语句所组成，这些语句描述了这个函数是怎样实现它的功能的。

在例1.1中，采用了比较整齐美观的所谓"缩进"格式来书写程序。事实上，C++程序的书写格式是非常自由的，比如，上例完全可以写成：

```
// EX1_1.CPP
/* This program demonstrats the construction of C++
   program file. */
#include <iostream.h>
int main(){cout << "i = ";// Display a prompt
int i;// Declear variable i
cin >> i;/* Get input for i */cout << Your number is " << i << "\n";
return 0;}
```

若主函数中没有单行注释的话，甚至可以将整个函数体全部连成一行。编译器是完全可以正确理解这个程序的。但对人来说，阅读这个程序实在是太困难了。

1.4 基本的输入/输出

一个程序通常会要求用户向它提供一些信息，程序接收外部信息的操作就叫作程序的输入。一个程序通常总是要向用户发出一些信息，程序向外部发送信息的操作就叫作程序的输出。C++程序输入操作可由流 cin 来完成，而输出操作则可由流 cout 来完成。