高等院校计算机精品教材系列

C++程序设计

免费提供
电子教案

郑 莉 李 超 编著

机械工业出版社
CHINA MACHINE PRESS

高等院校计算机精品教材系列

C++程序设计

郑莉　李超　编著

机械工业出版社

本书面向没有程序设计基础的初学者，立足基础，注重实践，讲解C++的基础语法和面向对象的程序设计方法，以及文件I/O、模板和异常处理机制，通过大量例题讲解程序设计思想，引导学习者在实践中掌握C++语言和面向对象的程序设计技术。每一章包括正文、复习思考、实验指导、自测练习，涵盖了全部教学环节。本书适于作为大专院校的教材，以及读者自学。

图书在版编目（CIP）数据

C++程序设计/郑莉，李超编著. —北京：机械工业出版社，2012.1
高等院校计算机精品教材系列
ISBN 978-7-111-36806-9

Ⅰ.①C… Ⅱ.①郑… ②李… 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（2011）第262504号

机械工业出版社（北京市百万庄大街22号 邮政编码100037）
责任编辑：郝建伟
责任印制：杨 曦
北京中兴印刷有限公司印刷
2012年3月第1版 · 第1次印刷
184mm×260mm · 21.25印张 · 527千字
0 001—3 000册
标准书号：ISBN 978-7-111-36806-9
定价：39.80元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

电话服务
社服务中心：(010)88361066
销售一部：(010)68326294
销售二部：(010)88379649
读者购书热线：(010)88379203

网络服务
门户网：http://www.cmpbook.com
教材网：http://www.cmpedu.com
封面无防伪标均为盗版

前　言

学习程序设计是一个充满新奇、不断探索的过程，在这个过程中学习者希望感到主动探究的乐趣，而不是被枯燥的语法规定所左右。因此，程序设计课程和教材的设计就需要以学习者为中心，通过实例引导学习者在实践中学会编写程序。

程序设计的目的，是对问题和解决问题的方法进行模拟，用计算机语言描述，进而利用计算机进行问题求解。C ++ 语言是一种面向对象的高级语言，而面向对象的观点正是人类看待自然的观点。不过人类以自然语言思维，而计算机程序设计语言的语法要比自然语言简单很多，词汇量也少很多，因此当读者以程序设计语言来编写程序时，就会感到困难。因此读者需要学习计算思维，也就是以计算机可以理解的方式思维，以程序设计语言能够表达的方式描述问题和解决问题的方法。

本书面向没有程序设计基础的初学者，立足基础，注重实践，介绍面向对象的程序设计方法和 C ++ 的基础语法，通过大量例题讲解程序设计思想，引导学习者在实践中掌握 C ++ 语言和面向对象的程序设计技术。本书每一章包括正文、复习思考、实验指导、自测练习，涵盖了全部教学环节。每章授课学时建议为 2 学时，实验和自测练习建议 4 学时。在附录中给出了自测练习的参考答案。

各章内容简介如下。

第 1 章程序与数据：作为全书的开始，首先介绍有关计算机程序设计的一些基本概念，并结合简单的 C ++ 程序简单介绍对面向对象的思想和 C ++ 程序的特点。然后介绍信息在计算机中的表示和存储。

第 2 章运算的模拟：模拟人脑思维活动是计算机的主要功能之一。人脑可以完成算数运算、比较运算、逻辑运算等多种运算。那么，计算机中如何完成这些数据运算？这一章介绍用 C ++ 语言编写程序进行计算的基础知识。

第 3 章流程控制：介绍 C ++ 中的选择结构和循环结构控制语句。

第 4 章类与对象：介绍类和对象的基本知识，包括对现实事物的模拟、类的设计、对象的定义和使用。

第 5 章共享与数据保护：介绍类的静态成员、友元类和共享数据的保护。

第 6 章全局函数：全局函数一般表示某项独立的功能，不属于任何一个类。这一章介绍函数的定义和调用、函数间的参数传递，以及函数的嵌套调用和递归调用。

第 7 章数组与字符串：数组是用于存储和处理大量同类型数据的数据结构。这一章介绍数组的定义及使用，包括一维数组和多维数组，以及用数组存放字符串。

第 8 章指针：介绍通过地址访问内存单元的方法，地址类型的变量——指针，指针的运算，以及指针访问数组和传递函数参数。

第 9 章动态内存分配：介绍实现动态内存分配和释放的 new 和 delete 运算，以及实现深拷贝的复制构造函数。

第 10 章类的重用：介绍类的组合和继承这两种类的重用机制，以及在组合和继承的情况下对象的构造与析构过程。

第 11 章多态性：介绍多态的基本概念、虚函数和抽象类。

第 12 章流类库与输入/输出：介绍流的概念及流类库结构，以及文件输入与输出。

第 13 章运算符重载：介绍运算符重载的方法，并给出了几种典型运算符重载的示例。

第 14 章模板：介绍函数模板和类模板的基本概念、定义和应用。

第 15 章异常处理：介绍异常处理的基本思想和语法，C ++ 的标准异常类。

与本书配套的电子讲稿和例题源代码可以从出版社的网站下载。

参与本书编写工作的还有朝乐门、李超（男）、廖学良、王荣祥。

如对本书内容有任何意见或建议，欢迎与作者联系。作者 Email 地址：zhengli@ tsinghua. edu. cn。

作　者

2011 年 4 月于清华大学

目　录

第1章　程序与数据

作为全书的开始，本章首先介绍有关计算机程序设计的一些基本概念，并结合简单的C++程序简单介绍面向对象的思想和C++程序的特点。然后介绍信息在计算机中的表示和存储。

1.1　程序设计概述

1946年2月14日，世界上第一台数字电子计算机ENIAC（“电子数字积分计算机”的简称，英文全称为Electronic Numerical Integrator And Computer）在美国宾夕法尼亚大学诞生。自此以后，在短暂的60多年时间里，计算机与信息科学得到了超乎寻常的迅猛发展。

计算机系统包括两大部分，即硬件和软件。硬件就是指各种看得见摸得着的实实在在的物理电子器件，而软件则是存储在硬件系统中的计算机运行时所需要的各种程序以及与之有关的文档和数据资料。计算机的硬件要正确地协同工作，就需要靠软件中的程序来进行正确控制。可以说硬件是计算机的身体，而软件是计算机的头脑和灵魂。

所谓程序，简单地讲就是我们写给计算机的一系列操作指令以及要处理的数据。我们知道，一台裸机是做不了什么事情的。为什么呢？那是因为计算机硬件系统本身并不具有智能，计算机之所以能够协助我们进行计算、事务处理，或者为人们提供娱乐，是因为软件工程师将处理这些计算和事务的方法、步骤写成了指令序列，并且输入到计算机中。计算机识别并执行这些指令，才具有了丰富多彩的功能。

所以说，程序就是“计算任务的处理对象和处理规则的描述”。

那么我们如何表述一系列的操作指令，也就是如何向计算机准确描述我们希望的操作呢？这就需要借助于计算机语言。

多数情况下，我们是使用高级语言来编写程序，因为高级语言的抽象程度更高，更接近人类的思维习惯，对人来说更好用。但是计算机能够识别的却只有机器语言。这中间就需要由翻译程序来进行翻译。

翻译程序有三种不同类型：汇编程序、编译程序和解释程序。

1）汇编程序：其任务是把用汇编语言写成的源程序，翻译成机器语言形式的目标程序。所以用汇编语言编写的源程序先要经过汇编程序的加工，变为等价的目标代码。

2）编译程序：若源程序是用高级程序设计语言所写，经翻译程序加工生成目标程序，那么该翻译程序就称为“编译程序”。所以高级语言编写的源程序要上机执行，通常首先要经编译程序加工成为机器语言表示的目标程序。若目标程序是用汇编语言表示，则还要经过一次汇编程序的加工。

3）解释程序：这也是一种翻译程序，同样是将高级语言源程序翻译成机器指令。它与编译程序不同点就在于它是边翻译边执行的，即输入一句、翻译一句、执行一句，直至将整个源程序翻译并执行完毕。解释程序不产生整个的目标程序，对源程序中要重复执行的语句

(例如循环体中的语句)需要重复地解释执行,因此较之编译方式要多花费执行时间,效率较低。

C++语言是高级语言,将C++程序翻译为机器语言的是编译程序。

1.1.1 计算机程序设计语言

语言是人类思维与表达的工具,人类的自然语言经过几千年文明的发展,具有了丰富的语法和词汇,可以方便地表达人的思想和情感。但是目前的计算机没有能力理解如此复杂的语言。虽然我们通俗地将计算机称为“电脑”,但是它只是机器,完全不具备“脑”的功能。计算机能够理解的只有二进制代码,称为“机器语言”。

实际上,早期的程序员工作十分艰辛,需要用二进制的机器语言来编写程序。那时,机器语言与人类自然语言间的鸿沟阻碍了人与机器的自如交流,用机器语言难以表述、模拟现实社会中的复杂问题以及解决问题的方法。即使稍后出现了汇编语言,那也只是机器语言的一种助记符,仍然属于低级语言,语法十分简单,词汇也很少,基本上只用来写科学计算程序。

1956年IBM公司的J. Backus等人研制出了第一个高级语言——Fortran语言。高级语言采用了英语中的词汇作为关键字,其语句也与英语有些相近。相对于低级语言来说,高级语言与人类自然语言之间的鸿沟小了一些。高级语言能够描述数据、运算、控制和传输,能够实现相对复杂的算法,但是对于大型、复杂的程序,比如目前普遍使用的图形界面程序,其开发的难度仍然非常大,甚至可以说不能胜任了。

由于这时的程序设计的思想都是面向任务解决过程的,因而这种程序设计思想也称为面向过程的编程思想。基于面向过程的编程思想而设计的高级程序设计语言就是面向过程的语言。

然而,到了20世纪80年代末,传统的面向过程的编程思想的缺点渐渐地变得突出起来。因为随着时代的发展和计算机及其程序设计语言功能的强大,人们越来越需要计算机处理更大量的数据。随着数据量的增大,数据和操作的严格分离使得程序可理解性越来越差,这种分离的思想渐渐地成为了程序开发能力和效率的瓶颈。

始于20世纪60年代的SIMULA 67、形成于20世纪70年代的Smalltalk是面向对象语言的开始。面向对象的语言,可以支持以人类习惯的面向对象的观点描述问题,适合大型复杂程序的开发。目前应用最为普遍的面向对象语言是C++和Java,而C++不仅支持面向对象的程序设计,也支持面向过程的程序设计、泛型程序设计(Generic Programming)、生成式程序设计(Generative Programming)。

那么,面向对象的本质是什么呢?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。对同类型对象抽象出其共性,形成类。类中的大多数数据,只能用本类的方法进行处理。类通过一个简单的外部接口与外界发生关系,对象与对象之间通过消息进行通信。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。通过继承与多态性,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。

一般来讲,对象可以看做是对客观世界事物的抽象,被抽象的事物可以是有形的物理实体,也可以是无形的计划、规则等。在面向对象的编程思想中,对象是程序反映客观世界的基本单元。客观世界的事物都拥有其静态属性(如组成部件等)和动态属性(如运作方式等),

与之相对应的，对象也包含属性和行为两个部分。其中，属性是用于描述客观事物静态属性的一组数据项，而行为则是用于描述客观事物动态属性的一组对数据项进行操作的操作序列。

在面向对象的语言中，与对象相关的一个重要概念是类，类可以看做是对具有相同属性的对象的抽象。类是对所有具有同样属性和操作的对象的抽象，而每个对象是它所属的类的具体实例。面向对象的三个重要性质——封装性、可复用性与多态性。

虽然面向对象的编程思想是程序设计思想的巨大飞跃，但我们仍应该认识到这样两点。第一，面向对象的思想是对面向过程的思想的改进而非完全否定。因为，在每一个类的设计内部，特别是操作序列（表现为函数、过程、方法等）的设计时，仍然会用到顺序、分支、循环的结构，仍然会用到面向过程的思想。因此，面向过程的思想仍然是程序设计的基础，在学习面向对象思想的同时，仍然要注意面向过程思想的学习；第二，程序设计的思想仍在发展中，比如最近提出的面向服务的编程思想，以及有些学者提出的下一代编程语言的思想等。因此，我们在学好面向对象编程思想的同时，还应该紧跟时代的步伐，随时准备好接受新的编程思想和新的编程语言。

1.1.2　C++语言

C++语言是由 C 语言发展变化而来的。C 语言是由贝尔实验室的 Dennis Ritchie 在 B 语言的基础上开发而来的，1972 年在一台 DEC PDP－11 计算机上实现了最初的 C 语言。

C 语言设计严谨，而且与硬件实现无关，这在设计理念上使得用 C 语言编写的程序可以比较容易地移植到不同的平台下使用。因此，在它在诞生之后，就很快在各种平台的计算机上推广发展，从而出现了许多类似但不兼容的 C 语言版本，这种情况反而给 C 语言程序的跨平台移植带来了极大不便。因此，为了明确定义一种标准的与机器无关的 C 语言，1989 年美国国家标准协会制定了 C 语言的标准（ANSI C），目前流行的 C 语言版本基本上都是以它为基础的。

C 语言有着诸多独特的优点：

1）它的语言简洁、方便、灵活。

2）它提供了丰富的运算符和数据结构。

3）C 语言同时具备高级语言和汇编语言的优点，它能够直接访问内存地址和进行位操作，这一特性使它在操作系统的开发领域中担任了重要角色，著名的 UNIX、Linux、Solaris 等操作系统都是由 C 或 C++开发完成的。

4）同时，C 语言程序又比汇编语言程序更具有良好的可读性和可移植性。

5）具备结构化控制语句，生成的目标代码质量高，程序运行效率高。

尽管如此，C 语言仍然只是一个面向过程的语言，随着时代的发展，思想的变革，人们有了将 C 语言改造成面向对象语言的需求。1980 年，AT&T 贝尔实验室的 Bjarne Stroustrup 博士开始对 C 语言进行改进和扩充，创建了一个新的语言，最初这个语言被称为“带类的 C”，1983 年正式取名为 C++。C++的两个“+”号各有含义，第一个“+”号表示C++语言首先是一个更好的 C 语言，它根除了 C 语言中存在的一些问题；第二个“+”号表示 C++在 C 语言的基础加入了对面向对象程序设计的支持。

1989 年，C++语言的标准化工作正式开始，到 1994 年制定了 ANSI C++标准草案，之后又几经改进和完善，在 1998 年 11 月被国际标准化组织（ISO）批准为国际标准，就成为

了现在的C++标准。尽管如此，目前C++语言仍然在不断发展和改进中。

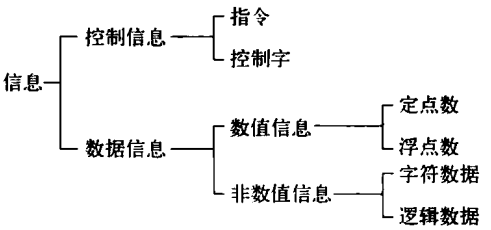
由于C++语言是由C语言发展而来的，因此，C++语言完全兼容C语言。这意味着，许多C语言程序不加修改就可以被C++语言使用；但另一方面，正是因为完全兼容C语言，使得C++语言不是一个完全纯正的面向对象的语言，它同时支持面向过程的程序设计和面向对象的程序设计，而也正像它的名字一样，它的面向对象的程序设计部分某种意义上像是原有的面向过程的程序设计部分的一个功能无比强大的附加体。

虽然C++具有这样的两重性，但从理念上，C++与C已经是完全不同的语言了。而且我们要更好地使用C++，就必须更注重学习其面向对象的思想，这样才更便于我们更高效地开发出功能更强大的程序来。

同时，也正是因为C++具有这样的两重性，使得有C语言编程基础的读者可以跳过本书与面向过程程序设计有关的章节，只学习C++语言的新加功能，从而大大节省学习时间；而对于没有编程基础的读者来说，则可以直接学习C++的面向对象的思想与功能，而把面向过程的部分作为程序编写的基本技能来学习，从而避开编程思想转换的困难。

1.2 数据的存储与表示

计算机加工的对象是数据信息，而指挥计算机操作的是控制信息，因此计算机内部的信息可以分成二大类：



本节主要介绍数据信息，有关控制信息的细节请参考有关硬件书籍。

1.2.1 数据的存储

在介绍C++语言里面数据的定义和使用之前，我们还是应该来介绍一点更为基本的东西，就是数据在计算机里面是如何存储和表示的。这一部分内容比较简单，但是比较多，因此篇幅很长，如果之前已经有这方面基础的读者可以略过本小节不看，没有基础的读者还是建议耐下心来慢慢学习一下，毕竟本部分是C++语言里数据定义与使用的基础。

1. 信息的存储单位

在计算机内部，各种信息都是以二进制编码形式存储，因此这里有必要介绍一下信息存储的单位。信息的单位通常采用“位”、“字节”、“字”。

1）位（bit）：度量数据的最小单位，表示一位二进制信息。

2）字节（Byte）：一个字节由八位二进制数字组成（1 Byte = 8 bit）。字节是信息存储中最常用的基本单位。

计算机的存储器（包括内存与外存）通常也是以多少字节来表示它的容量。常用的单位有：

1 KB = 1024 Byte

1 MB = 1024 KB

1 GB = 1024 MB

3）字（Word）：字是位的组合，并作为一个独立的信息单位处理。字又称为计算机字，它的含义取决于机器的类型、字长以及使用者的要求。常用的固定字长有 8 位、16 位、32 位等。

信息单位用来描述机器内部数据格式，即数据（包括指令）在机器内的排列形式，如单字节数据，可变长数据（以字节为单位组成几种不同长度的数据格式）等。

在讨论信息单位时，还有一个与机器硬件指标有关的单位，这就是机器字长。机器字长一般是指参加运算的寄存器所含有的二进制数的位数，它代表了机器的精度。如 32 位，64 位等。

2. 二进制数的编码表示

一个数在机内的表达形式称为“机器数”，而它代表的数值称为此机器数的“真值”。

前面已经提到，数值信息在计算机内是采用二进制编码表示。数有正、负之分，在计算机中如何表示符号呢？一般情况下，用“0”表示正号，“1”表示负号，符号位放在数的最高位。

例如，8 位二进制数 A = （+1011011）和 B = （-1011011）在机器中可以表示为：

A:	0	1	0	1	1	0	1	1
B:	1	1	0	1	1	0	1	1

其中最左边一位代表符号位，连同数字本身一起作为一个数。

数值信息在计算机内采用符号数字化处理后，计算机便可以识别和表示数符。为了改进符号数的运算方法和简化运算器的硬件结构，人们研究了符号数的多种二进制编码方法，其实质是对负数表示的不同编码，详见表 1-1。

表 1-1　计算机二进制编码方式

	说　明	举　例
原码	将符号位数字化为 0 或 1，数的绝对值与符号一起编码，即所谓“符号——绝对值表示”的编码，称为原码	X = +0101011 [X]原 = 00101011 X = -0101011 [X]原 = 10101011 X = 0.1011 [X]原 = 0.1011 X = -0.1011 [X]原 = 1.1011
反码	反码很少使用，但作为一种编码方式和求补码的中间码。正数的反码与原码表示相同。负数的反码与原码有如下关系： 负数反码的符号位与原码相同（仍用 1 表示），其余各位取反（0 变 1，1 变 0）	X = +1100110 $[X]_{原}$ = 01100110 $[X]_{反}$ = 01100110 X = -1100110 $[X]_{原}$ = 11100110 $[X]_{反}$ = 10011001 X = 0.1011 $[X]_{原}$ = 0.1011 $[X]_{反}$ = 0.1011 X = -0.1011 $[X]_{原}$ = 1.1011 $[X]_{反}$ = 1.0100
补码	对于一个负数，其补码由该数反码的最末位加 1 求得。 对于正数来说，其原码、反码、补码形式相同	求 X = -1010101 的补码， $[X]_{原}$ = 11010101 $[X]_{反}$ = 10101010 $[X]_{补}$ = 10101011 求 X = -0.1011 的补码 $[X]_{原}$ = 1.1011 $[X]_{反}$ = 1.0100 $[X]_{补}$ = 1.0101

3. 定点数和浮点数

数值数据既有正、负之分，又有整数和小数之分，本节要介绍小数点如何处理。在计算机中通常都采用浮点方式表示小数，下面我们就来介绍数的浮点表示法。

一个数 N 用浮点形式表示（即科学表示法），可以写成：

$$N = M \times R^E$$

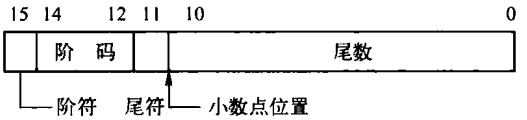
其中 R 表示基数，一旦机器定义好了基数值，就不能再改变了。因此基数在数据中不出现，是隐含的。在人工计算中，一般采用十进制，10 就是基数。在计算机中一般用二进制，因此以 2 为基数。

E 表示 R 的幂，称为数 N 的阶码。阶码确定了数 N 的小数点的位置，其位数反映了该浮点数所表示的数的范围。

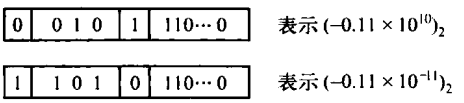
M 表示数 N 的全部有效数字，称为数 N 的尾数。其位数反映了数据的精度。

阶码和尾数都是带符号的数，可以采用不同的码制表示法，例如尾数常用原码或补码表示，阶码多用补码表示。

浮点数的具体格式随不同机器而有所区别。假设有一台 16 位机，其二进制浮点数组成为阶码 4 位，尾数 12 位，则浮点数格式如下：



下面是一个实际的例子，其中阶码用补码表示，尾数用原码表示：



4. 数值的范围

机器中数的表示范围与数据位数及表示方法有关。一个 M 位整数（包括一位符号位），如果采用原码或反码表示法，能表示的最大数为 $2^{m-1}-1$，最小数为 $-(2^{m-1}-1)$。若用补码表示，能表示的最大数值为 $2^{m-1}-1$，最小数为 -2^{m-1}。

这里要说明一点，由于补码中的“0”表示唯一的，故 $[X]_{补} = 100\cdots0$，对应的真值 $X = -2^{m-1}$，从而使补码的表示范围与原码有一点差异（注意：补码 100…0 的形式是一个特殊的情况，权为 2^{m-1} 位的 1 既代表符号，又表示数值）。对补码的表示范围，本书不做证明，读者如果感兴趣，可以自行验证一下。

例如，设 M = 8，则原码表示范围为 -127 ~ +127，反码的表示范围也是 -127 ~ +127，补码的表示范围是 -128 ~ +127。

一个 n 位定点小数，小数点左边一位表示数的符号，采用原码或反码表示时，表数范围为 $-(1-2^{-n})$ ~ $(1-2^{-n})$。采用补码表示时，表数范围为 -1 ~ $(1-2^{-n})$。

至于浮点数的表示范围，则由阶码位数和尾数位数决定。

若阶码用 r 位整数（补码）表示，尾数用 n 位定点小数（原码）表示，则浮点数范围如下。

$$-(1-2^{-n}) \times 2^{2^{r-1}-1} - 1 \sim +(1-2^{-n}) \times 2^{(2^{r-1}-1)}$$

为了扩大数的表示范围，应该增加阶码的位数，每加一位，数的表示范围就扩大一倍。