

高等学校计算机基础课程系列教材

艾德才 主编
杨宏宇 编著

龚涛 李俊生

C++

程序设计简明教程



中国水利水电出版社
www.waterpub.com.cn

高等学校计算机基础课程系列教材

C++程序设计简明教程

艾德才 主编

龚 涛 李俊生 杨宏宇 编著

中国水利水电出版社

内 容 摘 要

本书的突出特点是：内容简洁，重点突出，实用性强。本书把 C++ 的精华全部概括其中，书中所有程序都是作者教学经验的结晶，都是在计算机上调试过的。其中不乏指导学生学习参加 C++ 程序设计比赛的优秀作品。

全书共分八章，分别介绍了 C++ 的基本概念；数据类型、语句；函数和程序的基本构成；类的定义、派生与继承；函数和操作符的重载；数据流与文件等内容。在每一章的后面都附有适量的习题，读者可通过习题巩固已学的知识。上机练习是每章学习过程中必需的工作，通过上机练习把理论知识与实际应用结合起来，加深对 C++ 程序设计技术的理解并掌握程序设计的技巧。

本书可作为高等院校本专科各专业学习程序设计课程的教学用书，以及自学 C++ 程序设计的参考书。

图书在版编目 (CIP) 数据

C++ 程序设计简明教程 / 艾德才主编，龚涛，李俊生，杨宏宇编著. —北京：中国水利水电出版社，2000.12

高等学校计算机基础课程系列教材

ISBN 7-5084-0517-X

I. C… II. ①艾… ②龚… ③李… ④杨… 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 (2000) 第 73328 号

书 名	C++ 程序设计简明教程
作 者	艾德才 主编 龚涛 李俊生 杨宏宇 编著
出版、发行	中国水利水电出版社 (北京市三里河路 6 号 100044) 网址: www.waterpub.com.cn E-mail: sale@waterpub.com.cn 电话: (010) 63202266 (总机)、68331835 (发行部)
经 售	全国各地新华书店
排 版	北京万水电子信息有限公司
印 刷	北京市天竺颖华印刷厂
规 格	787×1092 毫米 16 开本 16.25 印张 352 千字
版 次	2001 年 1 月第一版 2001 年 1 月北京第一次印刷
印 数	0001—6000 册
定 价	22.00 元

凡购买我社图书，如有缺页、倒页、脱页的，本社发行部负责调换
版权所有·侵权必究

前 言

面向对象的程序设计技术，在近年来有了突飞猛进的发展，亦成为最热门、最实用的软件开发手段。与传统的面向过程的程序设计技术完全不同，面向对象的程序设计技术，把现实世界的问题抽象为“类”，而我们要解决的问题是对类所生成的对象的一系列操作。数据抽象性、信息封装性、继承性以及动态联编是面向对象程序设计的四个显著特点。数据抽象在对象的定义和对象的实现之间形成了明显的分界，某类事务共同拥有的属性抽象为“类”，而具体对象只是类的一个实例；继承性使得程序的代码可以重用，而以虚函数形式支持的“多态性”，使得程序更加灵活。

C++程序设计语言是一门高效、实用的程序设计语言。C++语言保持了与 C 语言的兼容，一方面是为了广大的熟悉 C 语言的程序设计人员，他们只需学习 C++语言新增加的功能部分；另一方面也是为了保护广大用户在 C 语言上所做的大量的人力物力上的投资。作为一种程序设计语言，C++有它自己的抽象特征和结构特征。

在面向对象的程序设计语言中，C++是最常用的。C++是一种新型的程序设计语言，虽然在结构和规则上是 C 语言的增强与扩充，但由于 C++引入了面向对象的程序设计思想，已不再是 C 语言的高版本，而成为一种新的面向对象的程序设计语言。C++允许程序设计人员把传统的结构化的程序设计技术与面向对象的程序设计技术结合起来，以这种方式，程序设计人员取两种程序设计手段之所长，从而可开发大型软件系统。程序设计人员利用 C++面向对象的程序设计技术，派生的类（derived classes）可被定义，这些类从一个或多个基类（base classes）继承所有特性，并可进行重新定义和增加新特性。应用结构化程序设计技术的函数定义了每种方法，以确保执行这些方法的一系列步骤是正确的。一旦实际应用中的对象的类被定义或确定，用结构化程序设计技术编写的应用程序，再一次保证程序运行期间的正确性。本书在编写过程中，采用了大量的程序实例，由浅入深，系统地介绍了 C++的特点和用 C++进行面向对象程序设计的技术手段，帮助读者真正掌握 C++语言和面向对象的程序设计方法。全书共分八章，分别介绍了 C++的基本概念、数据类型、语句、函数和程序的基本构成；类的定义、派生与继承；函数和操作符的重载；数据流与文件等内容。在每一章的后面都附有适量的习题，读者可通过习题巩固已学的知识。上机练习是每章学习过程中必需的工作，通过上机练习把理论知识与实际应用结合起来，加深对 C++程序设计技术的理解并掌握程序设计的技巧。

本书是作者多年教学经验和实际程序设计经验相结合的结果，选材从实际出发，深入浅出，语言通俗易懂，并兼顾了深度和广度。本书中所使用的实例均在计算机上调试通过，希望读者进一步上机去体会 C++的程序设计技巧和面向对象程序设计的精髓。本书第一章、第

二章由龚涛同志执笔，第四章、第五章、第六章由李俊生同志执笔，第三章、第七章、第八章由杨宏宇同志执笔。艾德才教授审校了全部书稿。

编写本教材，是在 C++程序设计教学上从繁到简的一次尝试，虽力图做好，但由于作者水平有限，不足之处在所难免，殷切希望广大同仁和读者不吝赐教，以便使本教材质量进一步得到提高。

编者

2000 年 10 月于天津

目 录

前言

第一章 C++概述	1
1.1 C++的发展与特点	1
1.1.1 程序设计方法简述	1
1.1.2 C++的发展	1
1.1.3 C++面向对象程序设计的特点	2
1.2 C++程序的构成	2
1.2.1 一个简单程序的构成	2
1.2.2 编译	3
1.2.3 注释	3
1.2.4 函数	3
1.2.5 C++程序的结构	4
1.3 变量与常量	5
1.3.1 变量	5
1.3.2 变量的定义	7
1.3.3 typedef 语句	7
1.3.4 常量	8
1.3.5 数值常量	8
1.3.6 符号常量	8
1.3.7 枚举常量	9
1.4 表达式与语句	9
1.4.1 语句	9
1.4.2 表达式	9
1.4.3 运算符	10
1.4.4 类型转换	13
1.5 输入/输出	14
1.5.1 输出	14
1.5.2 特殊输出字符	16
1.5.3 八进制和十六进制数的输出	17
1.5.4 控制输出宽度	17

1.5.5 输入	18
1.6 数组	19
1.6.1 一维数组的定义及引用	19
1.6.2 二维数组的定义及引用	20
1.6.3 数组的初始化	20
习题	22
第二章 C++流程控制	23
2.1 if 语句	23
2.2 switch 语句	25
2.3 转向语句	26
2.3.1 break 语句	26
2.3.2 continue 语句	27
2.3.3 goto 语句	27
2.4 for 循环	28
2.5 while 循环	30
2.6 do while 循环	32
2.7 程序举例	35
习题	38
第三章 函数	39
3.1 函数的定义与说明	39
3.1.1 函数定义	39
3.1.2 函数原型	40
3.2 函数调用	40
3.2.1 函数调用的概念	40
3.2.2 函数调用的形式	42
3.2.3 函数调用过程	43
3.3 C++函数中的 return 语句	45
3.3.1 从函数返回	45
3.3.2 返回值	48
3.3.3 void 型函数	48
3.4 函数的参数	49
3.4.1 函数的形式参数和函数的实参数	49
3.4.2 参数传递	50

3.4.5 指针、数组和函数的关系	58
3.4.6 函数 main () 的参数	59
3.5 局部变量和全局变量	61
3.5.1 局部变量	61
3.5.2 全局变量	62
3.6 嵌入式函数	63
3.7 递归函数	64
3.8 函数指针	65
3.9 库函数	68
习题	70
软件设计	70
第四章 C++类	71
4.1 面向对象程序设计术语	71
4.1.1 对象	71
4.1.2 类、子类、继承性和重置	72
4.1.3 对象的封装性	73
4.1.4 多态性	73
4.2 C++类的结构	73
4.2.1 类的结构	73
4.2.2 对象的实现	75
4.3 类的成员	76
4.3.1 类的数据成员	77
4.3.2 构造函数与析构函数	77
4.3.3 成员函数	79
4.3.4 在线函数 (inline)	81
4.3.5 成员函数名的重载	83
4.3.6 进一步讨论构造函数	85
4.4 类的嵌套	90
4.5 const 型成员函数	93
4.6 静态数据成员与成员函数	95
4.6.1 静态数据成员	95
4.6.2 静态成员函数	98
4.6.3 举例	98
4.7 C++程序	100
习题	101

上机练习	104
第五章 C++的复杂对象.....	105
5.1 数组	105
5.1.1 数组的定义与使用	105
5.1.2 数组的初始化	106
5.1.3 类类型数组	106
5.1.4 多维数组	109
5.2 指针	110
5.2.1 指针、地址、变量和运算符	110
5.2.2 类对象指针	112
5.2.3 堆栈和自由存储区	118
5.2.4 this 指针和 const 指针	121
5.3 引用	125
5.3.1 引用及其运算符	125
5.3.2 引用传递与值传递	127
5.4 字符串的处理	130
5.4.1 字符串	130
5.4.2 字符串库函数	132
5.4.3 字符串类	133
5.5 结构与联合	135
5.5.1 结构类型	135
5.5.2 联合	146
习题	149
上机练习	152
第六章 C++运算符的重载.....	153
6.1 二元算术运算符的重载	154
6.1.1 重载为类的成员函数	154
6.1.2 重载为类的友元函数	156
6.1.3 程序举例	157
6.2 一元自动加和自动减操作符的重载	159
6.3 关系运算符的重载	161
6.4 赋值操作符的重载	163
6.4.1 操作符“+=”和“-=”的重载	163
6.4.2 操作符“=”的重载	165
6.5 下标操作符和函数调用操作符的重载	167

6.5.1	下标操作符的重载	167
6.5.2	函数调用操作符的重载	168
6.6	操作符 new 和 delete 的重载	170
6.7	类型转换	173
6.7.1	一般意义上的类型转换	173
6.7.2	通过构造函数进行类型转换	174
6.7.3	通过类成员函数进行类型转换	176
6.8	逗号运算符的重载	177
习题		179
上机练习		180
第七章	继承和模板	181
7.1	继承	181
7.2	访问控制关键字	184
7.2.1	公有继承	184
7.2.2	私有继承	187
7.2.3	保护成员	188
7.3	继承与继承链	192
7.3.1	多继承	192
7.3.2	继承链	196
7.4	友元类和友元函数	198
7.4.1	友元类	198
7.4.2	友元函数	200
7.5	模板	203
7.5.1	函数模板	203
7.5.2	类模板	210
习题		213
软件设计		214
第八章	C++流与文件	215
8.1	C++的流	215
8.1.1	预定义流	215
8.1.2	流的优点和缺点	217
8.1.3	iostream 库	219
8.2	格式化 I/O	220

8.3 用户定义类型的输入/输出	227
8.3.1 重载操作符<<（创建用户插入符函数）	227
8.3.2 重载操作符>>（创建用户提取符函数）	230
8.3.3 创建用户操纵符函数	232
8.4 文件 I/O	234
8.4.1 流式文件 I/O 函数	234
8.4.2 流类文件 I/O	239
8.5 流状态与操作	246
习题	247
软件设计	248
参考文献	249

第一章 C++概述

1.1 C++的发展与特点

1.1.1 程序设计方法简述

多年以来，由于存储器价格昂贵以及处理器处理功能有限，计算机程序员的主要目标是力求编写短小代码的程序并使其运行速度更快。随着计算机成本的普遍降低，程序员工作时间的成本远远超过了用于工作的计算机的成本。因此，编写良好风格、易于维护的程序受到普遍欢迎。

程序设计方法也经历了过程化程序设计（procedural programming）、结构化程序设计（structured programming）、面向对象的程序设计（object-oriented programming）几个阶段。过程化的程序设计，其程序被认为是在一个数据集合上进行的一系列操作。而结构化程序设计提供了系统地组织这些过程和能够处理大量数据的有效手段。采用过程化程序设计的程序员发现每一种相对于老问题的新方法都要带来额外的开销，而采用结构化程序设计，当数据量增大时，数据与处理这些数据的方法之间的分离使程序变得难以理解。面向对象的程序设计的实质是把数据和处理这些数据的过程合并为一个单独的“对象”，一个具有确定特性的自完备的实体。面向对象程序设计作为 90 年代软件设计领域的主导技术，已受到越来越多的软件人员的重视。

1.1.2 C++的发展

各种程序设计语言对于程序设计方法的支持程度是不同的，C++完全支持面向对象的程序设计。C++的出现主要归因于 C 语言，C 语言在 C++中作为子集保留下来。C 语言是由 Dennis Ritchie 设计，并在 DEC PDP-11 上的 Unix 操作系统环境下实现的，而 C 语言是在 1970 年由 Ken Thompson 设计的 B 语言发展来的。1980 年贝尔实验室的 Bjarne Stroustrup 为了用于仿真课题，编写了称为“带类的 C”语言版本，1983 年 7 月将 C++名字定下来，并向研究小组之外发表。C++的名字强调了从 C 语言的演化特性，“++”是 C 的增量运算符，之所以没有叫 D 语言，是因为它是 C 语言的扩充，增加了面向对象的程序设计特征。因而 C++是面向对象程序设计的主流语言。C 语言本身也在发展，部分地受到 C++发展的影响。目前，常用的 C++编译程序有 Borland C++，MC C++，Visual C++，GNU C++等。随着 Internet 的高速发展，网络应用软件开发需求越来越多，Internet 服务器大都使用 Unix 操作系统，而 Unix 操作系统

与 C++有着天然的联系。Linux 操作系统的普及，使得在 Unix 操作系统开发软件不再是大型机专业人员特有的工具。因此，对于软件设计人员来说，使用 C++就意味着成功和希望。

1.1.3 C++面向对象程序设计的特点

C++完全支持面向对象的程序设计，它包含了面向对象的开发所具有四个重要特征：封装、数据隐蔽、继承和多态。

1. 封装和数据隐蔽

C++通过生成被称为类的用户定义的类型来支持封装和数据隐蔽。一个严格定义的类一旦生成后，就成为一个封装的实体，并可作为一个整体被使用。而类内部的行为被隐藏起来，用户只要知道如何使用这种严格定义的类，而不必知道它是如何工作的。

2. 继承与重用

C++通过继承来支持重用。可以通过说明一个扩展已有类型的新类型，这个新的子类是从已有类型派生的，也被称为派生类型。

3. 多态

多态性是指对象改变其形式的功能。一个多态性对象可以有多种形式。C++通过被称为函数多态与类多态的特征来支持这种“不同类型有各自的响应”的思想。

1.2 C++程序的构成

1.2.1 一个简单程序的构成

首先试编写一个完整的程序 Hello.cpp，使其显示一行文字输出：

程序 1.1

```
#include <stream.h>

main ()
{
    cout << "Hello,world.\n";
}
```

#include <stream.h>一行指示编译程序去包含标准的流（stream）输入输出设施的声明，这些设施可在 stream.h 文件中找到。没有这些声明，表达式 cout << “Hello,world.\n”就没有什么意义。运算符<<表明把它的第二个参数“Hello,world.\n 送到第一个参数”标准输出流 cout 中去。字符串是用双引号括起来的字符序列。\\n 表示换行字符。main () { ...}定义了名为 main 的函数，每一个 C++程序都必须有一个名为 main 的函数。

1.2.2 编译

输出流 `cout` 送到哪里？C++程序必须经过编译并产生可执行代码。一些专门的程序（C++编译程序）可以将 C++源程序转换为可执行代码。常见的 C++编译程序有 Turbo C++，Borland C++，MS C++，GNU C++编译程序。

如在 DOS 操作系统下使用 Turbo C++编译程序，编译 `Hello.cpp` 程序的命令如下：

`tcc hello.cpp` 后产生 `hello.exe` 可执行文件。

如在 DOS 操作系统下使用 Borland C++编译程序，编译 `Hello.cpp` 程序的命令如下：

`bcc hello.cpp <entre>` 或 `bc hello.cpp` 后产生 `Hello.exe` 可执行文件。

如在 Unix（如 Linux）操作系统下，编译 `Hello.cpp` 程序的命令如下：

`g++ hello.cpp` 或 `cc hello.cpp` 后产生 `a.out` 可执行文件。

如在 Windows 操作系统下，可使用 Visual C++编译程序。

1.2.3 注释

在编写一个程序时，为便于日后维护及修改，增加程序的可读性，需要使用注释。在 C++中有两种类型的注释（comment），一是双斜杠“//”类型，它是一种 C++风格的注释，它提示编译程序可以忽略双斜杠以后直至本行末尾的任何字符。二是“/*”和“*/”，它是一种 C 风格的注释，它告诉编译程序：可以忽略“/*”之后的任何字符直到找到“*/”标志。每一个“/*”必须有一个“*/”与之相匹配。优秀的注释告诉读者一段代码的目的，而不良的注释仅仅是重复每行代码的功能。代码本身就足以说明自身的功能。几乎没有程序员掌握适当的注释技巧，如何进行适当的得体的注释？这里有一条有益的规则：假设读你的程序的人可以读懂 C++，但不能理解你的设计思想，让源代码本身来说明程序正在做什么，而用注释来解释为什么要这样做。

程序 `Hello.cpp`，注释示例：

程序 1.2

```
#include <iostream.h>
main ()
{
    /*This is a comment and it extends until
       the closing star-slash comment mark .*/
    cout << "Hello,world!\n";
    //This comment ends at the line.
}
```

1.2.4 函数

`main()` 函数是一个特殊的函数，操作系统通过调用它来开始程序。其他函数是在程序

的运行过程中，由 `main()` 函数或另外一些函数来调用的。程序是按照源代码中的次序逐行执行，直到一个函数被调用，此时程序将转向去执行这个函数。当函数调用结束后，它把控制权交给调用函数的语句的下一行。函数由域名和函数体构成。域名的构成依次为：返回值类型、函数名称、函数形式参数。通过形式参数可以将数据传递给函数。下面的程序 1.3 是一个简单函数的示例：

程序 1.3

```
#include <iostream.h>
int add(int x,int y)
{ cout << "In add(),received "<< x <<"and " <<y <<"\n";
  return (x+y);
}
main()
{cout << "I'm in main()!\n";
  int a,b,c;
  cout << "Enter two number: ";
  cin >>a;
  cin >>b;
  cout << "\nCalling add() \n";
  c=add(a,b);
  cout << "\nBack in main() . \n";
  cout << "c was set to " << c;
}
```

输出：

```
I'm in main() !
Enter two numbers: 2 4
Calling add()
In add() ,received 2 and 4
Back in main()
c was set to 6
```

在第 2 行中定义了函数 `add()`，它接受两个整形式参数数并返回一个整形式参数数。从 `c=add(a,b)` 行，程序转向执行 `add()` 函数，在 `return (x+y)`，返回结果值，结束函数调用。`Cin` 的功能是将两个数值读入变量 `a` 和 `b` 中，而 `cout` 被用来向屏幕输出数据。

后面的几章中将会深入讨论函数的用途。

1.2.5 C++程序的结构

C++程序由头文件、主函数、函数组成。头文件由 `#include` 语句开始，这些语句通知编

译程序，程序中应包括相应头文件的内容。对程序而言，这些头文件的内容与程序体的内容起着相同的作用。头文件包含了编译程序为各个不同的操作运算提供的定义、C++ I/O 操作的定义和操作系统服务等。一个 C++ 程序总是从 `main()` 函数开始执行的，每一个 C++ 程序必须有一个且只允许有一个 `main()` 函数，它是运行程序的起始点。随着程序量的增加，可以把程序划分成几个小的、更易管理的部分，这就是函数。简单地说，函数实际上是一系列语句的集合，由它们共同完成一个专门的任务。程序中的各个函数必须有不同的函数名，每个程序至少包括一个函数。函数由函数域名和函数体组成，函数域名说明了返回值类型、函数名称、函数的形式参数组成。函数体由花括号 `{}` 括起的一系列语句组成。

C++ 程序完整的结构如下所示：

```
#include <...>
#include <...>
main ()
{
...
}
func1 (n1,n2,...)
{
...
}
fun2 (m1,m2,...)
{
...
}
...
```

1.3 变量与常量

1.3.1 变量

程序需要有一种用于存储所用数据的手段，而变量和常量为处理数据提供了一种使用内存中某个指定内存空间的存储手段。以程序员的观点，一个变量就是计算机内存中的一个存储单元，允许把一个值存入其中，且需要时能够访问它。变量只不过是某个内存地址的名字，它可以存放某个指定的值。

在 C++ 中定义一个变量时，不仅要告诉编译程序变量的名称，还需要说明它的类型，以便告诉编译程序为变量在内存中分配多大的空间。每一个存储单元是一个字节，C++ 中有许多不同的数据类型，它们的大小都是字节的整数倍。

定义了一种类型的变量，就说明了变量的三个性质：变量的大小（所占内存空间的大小）、变量的取值范围和所允许参加的运算（或可进行的操作）。因为不同类型的变量所占内存空

间的大小是不同的，其取值范围和允许参加的运算也是不同的。每种变量的类型占据的内存空间随机器类型的不同而有所区别。

下面所列程序 1.4 可用来确定计算机中不同类型变量的大小。

程序 1.4

```
#include <iostream.h>
int main ()
{
    cout << "The size of an int is: \t\t" <<sizeof (int) << "bytes.\n";
    cout << "The size of a short int is: \t" <<sizeof (short) << "bytes.\n";
    cout << "The size of a long int is: \t" <<sizeof (long) << "bytes.\n";
    cout << "The size of a char is: \t\t" <<sizeof (char) << "bytes.\n";
    cout << "The size of a flort is: \t\t" <<sizeof (float) << "bytes.\n";
    cout << "The size of a double is: \t" <<sizeof (double) << "bytes.\n";
}
```

输出:

```
The size of an int is:      2bytes
The size of a short int is: 2bytes
The size of a long int is:  4bytes
The size of a char is:     1bytes
The size of a float is:    4bytes
The size of a double is:   8bytes
```

该程序中，sizeof（）是由 C++编译程序提供的，说明了参数所占内存的大小。“\t”是水平制表符，表示输出到下一个特定位置，相当于五个空格。表 1.1 中列出了 C++程序所使用的变量的主要类型，以及变量类型的名称、所占内存大小和取值范围。由此可以看出，在定义变量类型时，应注意可能的取值范围，以防溢出。

表 1.1 变量的类型

类 型	大 小	取 值 范 围
无符号短整型 (unsigned short)	2 字节	0~65,535
短整型 (short 或 int)	2 字节	-32,768~32,767
无符号长整型 (unsigned long)	4 字节	0~4,294,967,295
长整型 (long)	4 字节	-2,147,483,648~2,147,483,647
字符型 (char)	1 字节	-128~127
浮点型 (float)	4 字节	3.4e-38~3.4e38
双精度型 (double)	8 字节	1.7e-308~1.7e308