C++ 程序设计教程

李代平　王琼英　罗寿文　编著

地震出版社

2002

图书在版编目（CIP）数据

C++程序设计教程 / 李代平等编著. — 北京：地震出版社，2002.1
ISBN 7-5028-1988-6

I. C… II. 李… III. C 语言—程序设计—教材
IV. TP312

中国版本图书馆 CIP 数据核字（2001）第 081584 号

内 容 简 介

本书全面系统地讲述了 C++的主要概念、语法、数据类型与表达式、控制结构、函数、数组、指针、结构体与共用体、类和对象、继承和派生类、多态与类属机制、C++I/O 流库、异常处理以及编程技巧等内容。例题丰富，各章配有练习题和实验指导。内容安排上循序渐进，深入浅出，力求通俗易懂，突出重点，侧重应用。

本书不仅可作为大专院校理科学生 C++程序设计的教材，也可作为 C++语言培训教材和工程技术人员的参考书。

C++程序设计教程
李代平　王琼英　罗寿文　编著
责任编辑：李小明
责任校对：王花芝

出版发行：地 震 出 版 社
北京民族学院南路 9 号　　邮编：100081
发行部：68423031　68467993　　传真：68423031
门市部：68467991　　传真：68467972
总编室：68462709　68423029　　传真：68467972
E-mail：seis@ht.rol.cn.net
经销：全国各地新华书店
印刷：广州家联印刷有限公司

版（印）次：2002 年 1 月第一版　2002 年 1 月第一次印刷
开本：787 × 1092　1/16
字数：553 千字
印张：24
印数：0001~3000
书号：ISBN 7-5028-1988-6 / TP · 58　（2539）
定价：35.00 元

前　言

C++是一门含 C 语言子集的高效程序设计语言，它既可以进行过程化程序设计，也可以用于面向对象的程序设计。C++语言具有较强的功能，也十分灵活，有些概念又比较抽象，因此一般认为比较难学。特别是不熟练的程序员，常常在发生错误后不知其所以然。现在 C++语言的应用已经很广泛，许多人都希望有一本既好学又实用的 C++语言入门教材。为此，我们在长期讲授和应用 C++语言的基础上编写了这本教材。

本书是作者集多年对本科学生的 C 语言和 C++语言教学经验，针对理科学生在学习 C++语言时经常遇到的概念问题进行详细讲解。为了读者更好地掌握 C++语言，作者根据自己长期在科研项目中开发应用 C++的体会，精心编写。本书在编写上由浅入深，容易理解；语言简洁，概念明确；内容全面，实例丰富；通俗易懂，便于自学。读者通过对本书的学习，能够正确地理解 C++语言的基本概念，基本掌握 C++面向对象的程序设计方法，能够进行基本的程序设计。

在教学实践中，作者深感学习 C++语言程序设计不仅是学习一门语言，更重要的是学习从问题到程序设计的整个过程的处理方法。因此，把如何帮助读者建立从问题到程序的整个思维过程和处理方法，使读者感到学习 C++语言并不是一件非常难的事作为本书的宗旨。基于此，本书根据读者的认知规律，在编排上采用难点分散的策略，使读者能循序渐进地逐步深入，在学习每一章内容时不会感到困难。

本书内容包括：

第 1 章绪论，主要介绍面向对象的主要概念，C++的词法与规则，C++程序的结构，程序设计的概念，算法的知识，C++程序的实现，Visual C++6.0 基本用法等内容。

第 2 章数据类型与表达式，主要介绍数据类型，常量，变量，存储类型说明，运算符与表达式，流控制，数据的输入输出等内容。

第 3 章控制结构，主要介绍 C++语句，分支结构，循环程序设计等内容。

第 4 章函数，主要介绍 C++语言程序组件，函数定义与声明，函数调用等内容。

第 5 章数组，主要介绍一维数组，二维数组与多维数组，字符数组和字符串等内容。

第 6 章指针，主要介绍指针的概念，指针运算，指针作函数参数，指针与数组，指针与函数，指向指针的指针等内容。

第 7 章结构体与共用体，主要介绍结构体类型变量的定义方法、结构体变量的初始化，结构体数组，结构体类型数据的指针，位段、共用体、枚举数据类型等内容。

第 8 章类和对象，主要介绍类，对象，对象的初始化，成员函数的重载，友元函数，指向类成员的指针等内容。

第 9 章继承和派生类，主要介绍基类和派生类，单继承，多继承，虚基类和应用实例等内容。

第 10 章多态，主要介绍运算符重载，函数重载，虚函数，抽象类等内容。

第 11 章类属机制，主要介绍函数模板，类模板等内容。

第 12 章 C++的 I/O 流库，主要介绍 I/O 标准流类，键盘输入，屏幕输出，插入符和提取符的重载，磁盘文件的输入和输出，字符串流等内容。

第 13 章异常处理，主要介绍异常的概念，基本原理，异常处理机制，异常处理的方法和多

第 1 章 绪论

计算机程序设计语言经过了从低级语言到高级语言的发展过程。在程序设计中，所使用的高级语言有许多种，如：BASIC、COBOL、FORTRAN、ALGOL、PASCAL、C 等。

C 语言是在 B 语言的基础上发展起来的。

1960 年出现了一种面向问题的高级语言 ALGOL 60，但它离硬件比较远，不宜用来编写系统软件。

1963 年英国剑桥大学推出了 CPL（Combined Programming Language）语言，后来经简化为 BCPL 语言。

1970 年美国贝尔（Bell）实验室的肯•苏姆普森（K.Thompson）以 BCPL 语言为基础，设计了一种类似于 BCPL 的语言，取其第一字母 B，称为 B 语言。并用 B 语言写了第一个 UNIX 操作系统，并在小型计算机 PDP-7 上得到了实现。

1972 年美国贝尔实验室的戴尼斯 M.里奇（Dennis M.Ritchie）为克服 B 语言的诸多不足，在 B 语言的基础上重新设计了一种语言，取其第二字母 C，故称为 C 语言。并在 DEC 公司的小型计算机 PDP-11 上，在 UNIX 操作系统环境下得到了实现。

1978 年布朗 W.卡尼翰（Brian W.Kernighan）和戴尼斯 M.里奇对 C 语言做了进一步充实和完善，出版了影响深远的名著《C 程序语言》（The C Programming Language）。

1980 年贝尔实验室对 C 语言进行了扩充，多次修改后起名为 C++。以后又经过不断的改进，发展成为今天的 C++。

C++是一种面向对象的程序设计语言。但是又同时含有 C 的子集。也就是说它既可以用于结构化程序设计，又可以进行面向对象程序设计。为了对面向对象程序设计有一个基本的了解，下面将介绍面向对象的基本概念、C++的规则、结构、C 与 C++的关系以及上机实验环境与步骤等知识。

1.1 面向对象的主要概念

面向对象是 C++中的主要概念，在学习 C++之前首先要了解这些概念。实际上面向对象强调在软件开发过程中面向客观世界（问题域）中的事物，采用人类在认识世界的过程中普遍运用的思维方法，所以在介绍它的基本概念时力求与客观世界和人的自然思维方式联系起来。

1.1.1 对象

在程序设计中，经常要用到对象这个概念，从一般意义上讲，对象是现实世界中一个实际存在的事物，它可以是有形的（比如一辆汽车），也可以是无形的（比如一项计划）。对象是构成世界的一个独立单位，它具有自己的静态特征和动态特征。静态特征即可以用某种数据来描述的特征，动态特征即对象所表现的行为或对象所具有的功能。

现实世界中的任何事物都可以称作对象。不过，人们在开发一个系统时，通常只是在

一定的范围（问题域）内考虑和认识与系统目标有关的事物，并用系统中的对象来抽象地表示它们。所以面向对象的方法在提到“对象”这个术语时，既可能泛指现实世界中的某些事物，也可能专指它们在系统中的抽象表示，即系统中的对象。这里主要对后一种情况讨论对象的概念，其定义是：

对象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。一个对象由一组属性和对这组属性进行操作的一组服务构成。

属性和服务，是构成对象的两个主要因素，其定义是：

属性是用来描述对象静态特征的一个数据项。

服务是用来描述对象动态特征（行为）的一个操作序列。

另外需要说明以下两点：第一点是对象只描述客观事物本质的、与系统目标有关的特征，而不考虑那些非本质的、与系统目标无关的特征。这就是说，对象是对事物的抽象描述。第二点是对象是属性和服务的结合体，二者是不可分的；而且对象的属性值只能由这个对象的服务来读取和修改，这就是后文将要讲述的封装概念。

1.1.2 类

实际上它是把众多的事物归纳、划分成一些类是人类在认识客观世界时经常采用的思维方法。分类所依据的原则是抽象，即：忽略事物的非本质特征，只注意那些与当前目标有关的本质特征，从而找出事物的共性；把具有共同性质的事物划分为一类，得出一个抽象的概念。例如：马、树木、石头等等都是一些抽象概念，它们是一些都具有共同特征的事物的集合，都可以被称作类。类的概念使我们能对属于该类的全部个体事物进行统一的描述。例如，“树具有树根、树干、树枝和树叶，它能进行光合作用”，这种描述适合所有的树，从而不必对每棵具体的树都进行一次这样的描述。

类的定义是：类是具有相同属性和服务的一组对象的集合，它为属于该类的全部对象提供了统一的抽象描述，其内部包括属性和服务两个主要部分。

在面向对象的编程语言中，类是一个独立的程序单位，它应该有一个类名并包括属性说明和服务说明两个主要部分。类的作用是定义对象。比如，程序中给出一个类的说明，然后以静态声明或动态创建等方式定义它的对象实例。

类与对象的关系如同一个模具与用这个模具铸造出来的铸件之间的关系。类给出了属于该类的全部对象的抽象定义，而对象则是符合这种定义的一个实体。所以，一个对象又称作类的一个实例（instance）。所谓“实体”、“实例”意味着什么呢？最现实的一件事是：在程序中，每个对象需要有自己的存储空间，以保存它们自己的属性值。同类对象具有相同的属性与服务，是指它们的定义形式相同，而不是说每个对象的属性值都相同。

读者可以对照非面向对象语言中的类型（type）与变量（variable）之间的关系来理解类和对象的关系。二者十分相似，在C++中，用类定义对象，用类型定义对象的成员变量。

1.1.3 封装

封装是面向对象方法的一个重要原则。它有两个涵义：第一个涵义是，把对象的全部属性和全部服务结合在一起，形成一个不可分割的独立单位（即对象）。第二个涵义也称作

“信息隐蔽”，即尽可能隐蔽对象的内部细节，对外形成一个边界（或者说形成一道屏障），只保留有限的对外接口使之与外部发生联系。这主要是指对象的外部不能直接地存取对象的属性，只能通过几个允许外部使用的服务与对象发生联系。

封装的信息隐蔽作用反映了事物的相对独立性。当站在对象以外的角度观察一个对象时，只需要注意它对外呈现什么行为（做什么），而不必关心它的内部细节（怎么做）。规定了它的职责之后，就不应该随意从外部插手去改动它的内部信息或干预它的工作。封装的原则在软件上的反映是：要求对象以外的部分不能随意存取对象的内部数据（属性），从而有效地避免了外部错误对它的“交叉感染”，使软件错误能够局部化，因而大大减少了查错和排错的难度。

1.1.4 继承

继承的定义是：特殊类的对象拥有其一般类的全部属性与服务，称作特殊类对一般类的继承。

继承意味着“自动地拥有”，或“隐含地复制”。就是说，特殊类中不必重新定义已在它的一般类中定义过的属性和服务，而它却自动地、隐含地拥有其一般类的所有属性与服务。面向对象方法的这种特性称作对象的继承性。

继承具有重要的实际意义，它简化了人们对事物的认识和描述。比如认识了轮船的特征之后，在考虑客轮时只要知道客轮也是一种轮船这个事实，那就认为它理所当然的具有轮船的全部一般特征，只需要把精力用于发现和描述客轮独有的那些特征。在软件开发过程中，在定义特殊时，不需把它的一般类已经定义过的属性和服务重复地书写一遍，只需要声明它是某个类的特殊类，并定义它自己的特殊属性和服务。无疑这将明显地减轻开发工作的强度。

继承对于软件复用是很有益的。在开发一个系统时，使特殊类继承一般类，这本身就是软件复用，然而其复用意义不仅如此。如果把用面向对象方法开发的类作为可复用构件提交到构件库，那么在开发新系统时不仅可以直接地复用这个类，还可以把它作为一般类，通过继承而实现复用，从而大大扩展了复用范围。

1.1.5 多态性

对象的多态性是指在一般类中定义的属性或服务被特殊类继承之后，可以具有不同的数据类型或表现出不同的行为。这使得同一个属性或服务名在一般类及其各个特殊类中具有不同的语义。

例如，在一般类“几何图形”中定义了一个服务“绘图”，但并不确定执行时到底画了一个什么图形。特殊类“椭圆”和“多边形”都继承了几何图形类的绘图服务，但其功能却不同：一个是画出一个椭圆，一个是画出一个多边形。进而，在多边形类更下层的一般类“矩形”中绘图服务又可以采用一个比画一般的多边形更高效的算法来画一个矩形。这样，当系统的其余部分请求画出任何一种几何图形时，消息中给出的服务名同样都是“绘图”（因而消息的书写方式可以统一），而椭圆、多边形、矩形等类的对象接收到这个消息时却各自执行不同的绘图算法。

1.1.6 抽象

从许多事物中舍弃个别的、非本质性的特征，抽取共同的、本质性的特征，就叫作抽象（abstraction）。抽象是形成概念的必要手段。

抽象原则具有两方面的意义：第一，尽管问题域中的事物是很复杂的，但是分析员并不需要了解和描述它们的一切，只需要分析研究其中与系统目标有关的事物及其本质性特征。对于那些与系统目标无关的特征和许多具体的细节，即使有所了解，也应该舍弃。第二，通过舍弃个体事物在节细的差异，抽取其共同特征而得到 -批事物的抽象概念。抽象是面向对象方法中使用最为广泛的原则。在软件开发领域中，早在面向对象方法出现之前就已经开始运用抽象的原则，主要是过程抽象和数据抽象。

1. 过程抽象

过程抽象是指任何一个完成确定功能的操作序列，其使用者都可把它看作一个单一的实体，尽管实际上它可能是由一系列更低级的操作完成的。

运用过程抽象，软件开发者可以把一个的功能分解为一些子功能；如果子功能仍然比较复杂则可以进一步分解。这使得开发者可以在不同的抽象层次上考虑问题，在较高的层次上思考时可以不关心较低层次的实现细节，即：只注意一个过程完成什么功能，而不注意它是怎样完成的。过程抽象不是面向对象分析的主要抽象形式，因为面向对象方法不允许超出对象的界限在全系统的范围内进行功能的描述。但是过程抽象对于在对象范围内组织对象的服务是有用的。

2. 数据抽象

数据抽象是根据施加于数据之上的操作来定义数据类型，并限定数据的值只能由这些操作来修改和观察。数据抽象是面向对象分析的核心原则。它强调把数据（属性）和操作（服务）结合为一个不可分的系统单位（即对象），对象的外部只需要知道它做什么，而不必知道它如何做。

1.1.7 C 与 C++的关系

C 语言是 C++的一个子集，C++包含了 C 的全部。

1. C++保持与 C 的兼容

兼容性的表现是许多 C 语言代码不需修改就可以为 C++所用。以前用 C 语言编写的许多库函数和应用软件都可以用于 C++。如果已经掌握了 C 语言的程序员，在学习 C++时只要再了解并掌握 C++的新特性就可以了。

由于这种兼容性，很快就会想到，C++不是一个纯正的面向对象的程序设计语言。因为 C 语言是面向过程的语言。也由于其兼容性，所以 C++也要支持面向过程的程序设计。由此可以看出 C++把两种不同的程序设计风格溶于一体。这就使初学者感到困惑。要提醒读者特别注意的是，不能用面向过程的思维方法去学习面向对象的技术。然而，在面向对象的程序设计中，对于一个专用函数内部的实现时，要用到面向过程的方法。

2. C++对 C 作了很多改进

C++保持了 C 的简洁与接近汇编的特点，同时又有重要的改进。

1）增加了一些运算符，如::，new，delete 等。

2）改进了类型系统，增加了安全性。C++规定数据类型转换采用强制转换。规定函数的说明必须采用原型。对于缺省函数作了限制。增加了编译系统检查类型的能力。

3）引进了引用概念，将引用作为函数参数为程序设计带来很大方便。

4）允许函数重载；允许设置缺省参数，这些措施提高了程序设计的灵活性。引入内联函数的概念，提高了程序设计的有效性。

5）变量更加灵活。在 C 语言中仅允许变量存在于函数体内或分段程序内。而且对变量必须先说明后使用，不能交叉使用，C++打破了这一规定，可根据要求对变量进行说明。

3. C++与 C 的差别

C++与 C 语言的差别在于 C 是面向过程的，C++是面向对象的。因此，C++是在对 C 语言改造的基础上又添加了面向对象的内容。

1.2 C++的词法与规则

1.2.1 C++的字符集

C++中有以下含字符。

1）数字：0, 1, 2, 3, 4, 5, 6, 7, 8, 9。

2）小写字母：a, b, …, y, z。

3）大写字母：A, B, …, Y, Z。

4）运算符：+, -, *, /, %, <, <=, =, >=, >, !=, ==, <<, >>, &, |, & &, ||, ∧, ~, ( ), [ ], { }, ->, . , ! , ? , ? :, , ,;, ", #。

5）特殊字符：（连字符或下划线）。

6）不可印出字符：空白格（包括空格、换行和制表符）。

1.2.2 词与词法规则

由上述字符构成了 C++的基本语法单位——单词（词汇）：标识符、关键字、运算符、分隔符、字符串、常量和注释。

下面逐一介绍一下。

1. 标识符

标识符是对实体定义的一种定义符，由字母或下划线（或连字符）开头、后面跟字母或数字或下划线（或空串）组成的字符序列，一般有效长度是 8 个字符（而 ANSI C 标准规定 31 个字符），用来标识用户定义的常量名、变量名、函数名、文件名、数组名和数据类型名和程序等。

2. 关键字

关键字是具有特定含义，作为专用定义符的单词，不允许另作它用，下面列举一些常用的关键字。

auto break case char class const continue default

do	ddefault	delete	double	else	enum	explicit	extern
float	for	friend	goto	if	inline	int	long
mutable	new	operator	private	protected	public	register	return
short	signed	sizeof	static	static_cast	struct	switch	this
typedef	union	unsigned	virtual	void	while		

3. 运算符和分隔符

有些特殊符号用作运算符。这里主要作为特殊单词加以说明，某些符号有几种含义，比如“%”，在作运算符时表示取余数（模），如9%5结果是4；在打印输出时，又表示某种格式如：printf（“%d”，a），意味着把变量a以十制数（d）形式印出。又如“,”既可作为运算符，也可作为一般的标点符号。

有时两个字符作为单词看待，如++和- -分别用于增量和减量运算；+=、-=、*=、/= 作为特殊运算符处理。

广义上说，C++语言的运算符也都是分隔符，分隔运算分量；但狭义上说，C++ 语言的分隔符主要是：空格、制表和换行符。

4. 字符串

字符串是由双引号括起来的字符。一个本来有意义的串，当被双引号括起来时，其原先的意义就失去了，而仅仅是一个字符串。

5. 常量

C++语言中常量包括实型常量（浮点常量）和整型常量（十进制常量、八进制常量、十六进制常量）、浮点常量、字符常量和字符串常量。

6. 注释

以“//”开头的内容是注释，用以帮助阅读、理解及维护程序。在编译时，注释部分被忽略，不产生目标代码。如下程序：

【例 1.1】在屏幕上打印 hello。

```
#include <iostream.h>
void   main ( )
{
        cout<<"hello";
      // printf"hello"
}
```

1.3 C++程序的结构

1.3.1 构成

C++语言程序由一个或多个函数构成，其中必须有一个主函数 main（）。可执行的程序总是从 main（）函数开始执行的。函数是完成某个算法的一个程序段，是C++语言的基本构成单位。

1. 函数

一个C语言是由若干个函数构成的。函数分为库函数（标准函数）和自定义函数。库函数一般是系统提供的。一个完整的的C++语言程序只有一个主函数。

2. 预处理命令

预处理命令以位于行首的符号“#”开始，C++提供的预处理有宏定义命令、文件包含命令和条件编译命令三种。

• 宏定义命令

格式一：

#　define　标识符（宏名）　字符串

如#define PI 3.14159 即指定用标识符“PI”来代替“3.14159”这个字符串常量，在编译预处理时把所有出现 PI 的地方都用 3.14159 来代替。该法使用户能以一个简单的名字代替一个长的字符串。

带参数的宏定义命令：

格式二：

#　define　宏名（参数表）　字符串

字符串包含在括弧中所指定的参数，例如：

```
#  define  S（a,b）  a*b
…
area=S（3,2;）
```

上例定义矩形面积 S，a 和 b 是边长，把 3、2 分别代替宏定义中的形式参数 a、b，即用 3*2 代替 S（3,2）。

因此赋值语句展开为：

```
area=3*2;
```

• 文件包含命令

指一个程序文件将另一个指定的文件的内容包含进来，即将另外的文件包含到本程序文件之中。一般称被包含的文件为嵌入文件（标题文件或头文件），即“.h”文件。

格式一：

```
#  include  "filename"
```

其中，filename 是一个确定的文件名，该语句向预处理程序表明，首先在源程序文件所属的当前文件目录中寻找指定的包含文件，如果找不到，则再在系统指定的标准目录中检索有关文件目录。

格式二：

```
#  include  <filename>
```

该语句提示预处理系统要它不检索源文件所在的文件目录，而直接按系统规定的标准目录中检索指定的文件。

文件包含命令很有用，可减少程序设计人员的重复劳动。

• 条件编译命令

根据给定的条件决定编译的范围，即可对程序源代码的各部分有选择地进行编译。一般形式：

```
#  if  条件
      源程序部分 1
```

```
#  else
        源程序部分 2
#  endif
```

如果条件成立，则保留源程序部分 1 参加编译，源程序部分 2 从源文件中删除，而不能参加编译；否则，保留源程序部分 2 而删除源程序部分 1。

3. 程序语句

一条完整的语句必须以分号“;”结束。程序语句有如下几类：

• 说明语句

用来说明变量的类型和初值。如下面语句是把变量说明为浮点数。

```
float   a,b,c;
```

又如下面语句是把变量 sum 说明为整型变量，并赋初值为零。

```
int   sum=0;
```

• 表达式语句

由一个表达式构成一个语句，用以描述算术运算、逻辑运算或产生某种特定动作，在任何表达式最后加一个分号就构成了一个语句。如下例由赋值表达式加“;”就构成一个赋值表达式语句。

• 程序控制语句

用来描述语句的执行条件与执行顺序的语句，C++语言的控制语句有 9 种：

```
if   （ ）   ~   else        条件语句
for  （ ）   ~               循环语句
while   （ ）   ~            循环语句
do   ~   while   （ ）       循环语句
continue                     结束本次循环语句
break                        中止循环式 switch 语句
switch                       多分支选择语句
goto                         转移语句
return                       从函数返回语句
```

以上语句中的括号（ ）表示其中是条件，~表示内嵌的语句。如：

```
if   （x > y）  z=x; else z=y;
```

【例 1.2】比较两个数是否相等。

```
main（ ）
{int a,b;
cout<<"a=";
cin>>a;
cout<<"b=";
cin>>b;
if（a==b）
cout"a==b"<<endi;
else
cout<<"a!= b"<<endi;
}
```

【例 1.3】打印九九表。

```
#include <iostream.h>
void   main（ ）
```

```
{int i,j;
for ( i=1;i<10;i++ )
cout<<i;
{for ( j=1;j<i;j++ )
cout <<i*j;
}
```

- 复合语句

复合语句是一种十分重要的语句，由大括号｛和｝把一些说明和语句组合在一起，使它们在语法上等价于一个简单语句；可由若干简单语句或复合语句组成。

如：

```
if  ( x>y )
      {  z=x;
         s=z*z;
        }
 else
      {  z=y;
         s=1/ ( z*z ) ;
        }
```

- 函数调用语句

函数调用语句是由一次函数调用加一个分号而构成的一个语句。

如：

```
sub(&*&  y);
```

- 空语句

空语句：

```
;
```

即只有分号“;”的语句，什么也不做。

1.3.2 书写格式

C++语言程序的书写格式自由度高，灵活性强，随意性大，如一行内可写一条语句，也可写几条语句；一个语句也可分写在多行内。不过应采用适当的格式书写，便于人们阅读和理解。

为了增加程序的可读性和利于理解，编写程序时按如下要点书写：

1）一般情况下每个语句占用一行。

2）不同结构层次的语句，从不同的起始位置开始，即在同一结构层次中的语句，缩进同样的字数（如每次缩进5个字符的位置）。

3）表示结构层次的大括弧，写在该结构化语句第一个字母的下方，与结构化语句对齐，并占用一行。

4）适当加些空格和空行。如在for和while后面应用一个空格与左括号隔开，数学运算符的左右各留一个空格以与表达式区别，表示参数时逗号后面留一个空格；按程序特性设置空行，表示上下是程序的两个部分。

这些注意点可以用也可以不用，仅是为了阅读理解方便而已，如例1-4这段程序。

【例1.4】打印出5与7的乘机。

```
#include <iostream.h>
void   main ( )
```