

■ 高职高专计算机系列规划教材

C++ 教程

郑阿奇 丁有和 主编



高职高专计算机规划教材

C++教程

郑阿奇 丁有和 主编

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

内 容 简 介

本书以最新 C++ 标准 (ISO/IEC 14882:2003) 为依据, 把 C++ 程序设计作为一个完整的体系, 特别适合直接作为 C++ 课程教材。同时, 内容讲解的顺序上分为两个部分, 第一部分: 第 1 章~第 9 章, C++ 语言结构化程序设计; 第二部分: 第 10 章~第 14 章, 以类为核心的面向对象程序设计。全书主要包含教程、上机实验指导和综合应用实习三部分内容。“实用教程”部分一般在讲解内容后紧跟实例, 各章配套的“习题”突出对 C++ 基础内容的训练; “上机实验指导”以 Visual C++ 6.0 (SP6 中文版) 为开发工具, 实验与教程同步配套, 通过实例先引导操作和编程, 然后提出问题思考或在原有基础上自己进行操作和编程练习。“综合应用实习”则是用 C++ 类进行综合应用训练, 能培养和提高掌握 C++ 编程思想和解决实际问题的应用能力。

本书可作为高职高专有关课程教材, 也可供广大学习 C++ 语言的人员参考使用。本套教程可免费下载教学课件、教程和上机实验指导中的源程序。

未经许可, 不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有, 侵权必究。

图书在版编目 (CIP) 数据

C++ 教程 / 郑阿奇, 丁有和主编. —北京: 电子工业出版社, 2009.8
高职高专计算机规划教材
ISBN 978-7-121-09318-0

I. C… II. ①郑…②丁… III. C 语言—程序设计—高等学校: 技术学校—教材 IV. TP312

中国版本图书馆 CIP 数据核字 (2009) 第 127997 号

策划编辑: 赵云峰

责任编辑: 张云怡

印 刷: 北京东光印刷厂

装 订: 三河市万和装订厂

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编: 100036

开 本: 787×1092 1/16 印张: 23 字数: 587 千字

印 次: 2009 年 8 月第 1 次印刷

印 数: 4 000 册 定价: 33.00 元

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 zltz@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线: (010) 88258888。

前言

C++语言是高职高专许多专业必须开设的课程，适合高职高专教学，有特色的 C++语言教材是老师和学生寻找的。为此，我们首先对目前市场上高职高专的 C++教材进行了深入分析，找出主要特点、存在问题，对如何让学生理解并且在此基础上解决小的应用问题，如何方便教师教学进行了专门研究，在继承实用教程系列的成功经验的基础上，针对高职高专的教学情况，编写了此书。

本书第一部分：第 1 章～第 9 章，C++语言结构化程序设计。该部分侧重于结构化程序的设计方法，通过文字、图形和实例对知识进行消化理解。第二部分：第 10 章～第 14 章，以类为核心的面向对象程序设计，通过综合应用实例，对知识进一步进行消化和理解。

本书有如下特点：

(1) C++的各个部分内容都有相应的侧重点，必须要掌握的难点在于点破其本质，就是说清、说透、说到位。让没有任何编程经验的读者也能理解 C++的内容。

(2) 教程部分一般在讲解内容后紧跟示例，凡标有【例 Ex_Xxx】均是一个完整的程序，且都通过上机调试。

(3) 解释尽可能图形化。例如下面两个示例图：

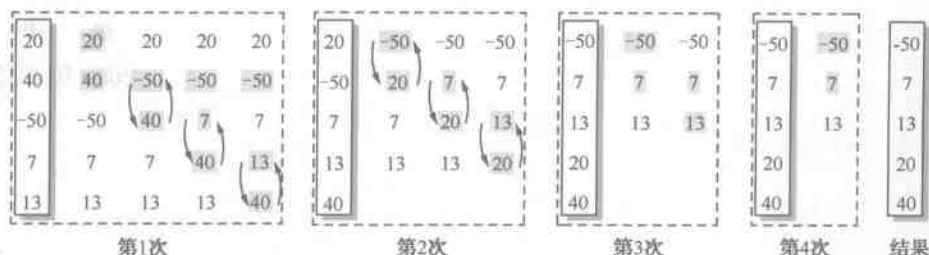


图 6.6 冒泡排序过程

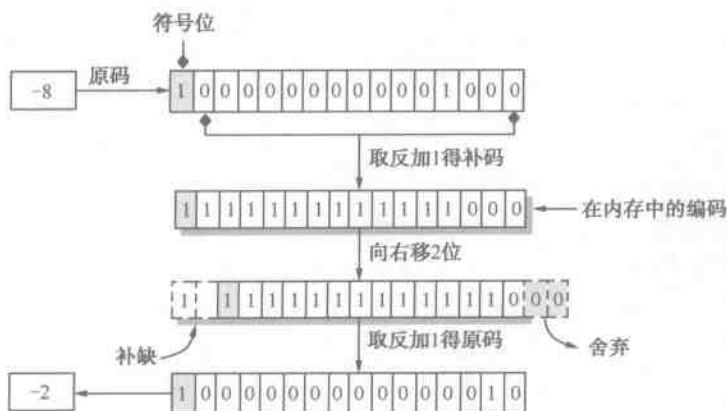


图 3.4 $(-8) \gg 2$ 操作及换算图例

(4) 本书以 Visual C++ 6.0 (SP6 中文版) 为 C++程序的开发工具，书中实例程序运行结果屏幕化，不易出错，更便于理解。例如：

12	76	4	1
-19	28	55	-6
2	10	13	-2
3	-9	110	22
最大值 = 110 位于矩阵的第 4 行，第 3 列。			

(5) 书中的源代码用底纹显示，对于 C++ 语言的语法内容、运行结果、提示和讨论内容、图例等均采用具有立体效果的方框来呈现，对于需要强调的文字内容则使用**黑体**来区分。另外，书中还用一些图标进行修饰。如“”表示说明、提示和本书约定的内容；“”表示讨论的内容；“”表示该程序是通过 Visual C++ 编译通过的。

(6) 实验 1 熟悉 Visual C++ 开发环境，并能掌握修正代码语法错误的方法。实验 8 在结构化程序设计学习之后掌握调试功能，让读者学用结合，克服学习了 C++ 却不会在开发环境中解决 C++ 问题的不足。

本书把 C++ 程序设计作为一个完整的体系，特别适合直接作为 C++ 课程教材。

本教程由电子工业出版社的教学服务平台 (<http://www.hxedu.com.cn>) 为读者提供服务，可免费下载教学课件、实例源文件等资料。

本书由南京师范大学郑阿奇、丁有和主编。其他很多同志也对本书的编写提供了许多帮助，在此一并表示感谢！

由于作者水平有限，不当之处在所难免，恳请读者批评指正。

编 者
2009 年 6 月

目 录

第 1 章 C++概述	1
1.1 C++程序创建	1
1.2 C++程序结构	2
1.2.1 main 函数	2
1.2.2 头文件包含	3
1.2.3 新头文件格式和名称空间	3
1.2.4 注释	3
1.2.5 C++程序组成	4
习题 1	5
第 2 章 数据类型和基本输入/输出	6
2.1 计算机内的数据表示	6
2.1.1 数制	6
2.1.2 机内的数值和字符表示	6
2.2 基本数据类型	7
2.2.1 整型	7
2.2.2 实型	8
2.2.3 字符型	8
2.2.4 布尔型	9
2.3 字面常量	9
2.3.1 整数常量	10
2.3.2 实数常量	10
2.3.3 字符常量	10
2.3.4 字符串常量	11
2.3.5 布尔常量	13
2.4 变量	13
2.4.1 变量名命名	13
2.4.2 变量定义	14
2.4.3 变量赋值和初始化	15
2.5 标识符常量	16
2.5.1 const 只读变量	16
2.5.2 #define 标识符常量	17
2.5.3 枚举常量	18
2.6 基本输入/输出	19
2.6.1 输入流 (cin)	19

2.6.2 输出流 (cout)	20
习题 2	21
第 3 章 运算符和表达式	22
3.1 算术运算	22
3.1.1 算术运算符	22
3.1.2 优先级和结合性	23
3.1.3 数据类型转换	23
3.1.4 代数式和表达式	24
3.2 赋值运算	25
3.2.1 左值和右值	25
3.2.2 数值截取和数值溢出	26
3.2.3 复合赋值	27
3.2.4 多重赋值	28
3.3 sizeof 运算符	29
3.4 逗号运算符	29
3.5 自增和自减	29
3.5.1 一般使用	29
3.5.2 前缀和后缀的区别	30
3.5.3 几点注意	31
3.6 位运算	32
3.6.1 位逻辑运算	32
3.6.2 移位运算	33
习题 3	35
第 4 章 基本语句和基本程序结构	37
4.1 C++ 语句概述	37
4.1.1 说明语句	37
4.1.2 表达式语句	37
4.1.3 块语句	38
4.2 分支语句	39
4.2.1 关系和逻辑表达式	39
4.2.2 if 语句	41
4.2.3 ?: 运算符及其表达式	46
4.2.4 switch 语句	48
4.3 循环语句	50
4.3.1 while 语句	50
4.3.2 do...while 语句	52
4.3.3 for 语句	53
4.3.4 循环语句的嵌套	54

4.4	转向语句.....	56
4.4.1	break 语句.....	56
4.4.2	continue 语句.....	57
4.4.3	goto 语句.....	58
4.5	结构化程序设计应用.....	59
4.5.1	算法和流程图.....	59
4.5.2	求最大公约数和最小公倍数.....	60
4.5.3	自动出题器.....	62
4.5.4	打印图案.....	62
习题 4		64
第 5 章	函 数.....	67
5.1	函数概述.....	67
5.2	函数的定义和声明.....	67
5.2.1	函数的定义.....	68
5.2.2	函数的调用和声明.....	70
5.3	函数的参数特性.....	72
5.3.1	全局变量和局部变量.....	72
5.3.2	参数传递方式.....	73
5.3.3	函数的默认形参值.....	74
5.4	函数的调用特性.....	75
5.4.1	函数重载.....	76
5.4.2	内联函数.....	76
5.4.3	函数嵌套调用.....	77
5.4.4	递归函数.....	79
习题 5		81
第 6 章	数 组.....	82
6.1	一维数组.....	82
6.1.1	一维数组的定义和引用.....	82
6.1.2	一维数组的初始化和赋值.....	84
6.2	二维数组.....	85
6.2.1	二维数组的定义和引用.....	86
6.2.2	二维数组的初始化和赋值.....	87
6.3	字符数组和字符串.....	89
6.3.1	一维字符数组.....	89
6.3.2	二维字符数组.....	90
6.4	数组与函数.....	91
6.4.1	地址传递和值传递.....	91
6.4.2	传递数组.....	92

6.5	排序	93
	习题 6	95
第 7 章	指针和引用	97
7.1	指针的定义和操作	97
7.1.1	地址和指针	97
7.1.2	指针的定义和引用	98
7.1.3	const 指针	101
7.2	指针和数组	102
7.2.1	指针和一维数组	102
7.2.2	指针和二维数组	105
7.2.3	字符指针和字符串	107
7.3	指针和函数	109
7.3.1	指针作为函数的参数	109
7.3.2	返回指针的函数	112
7.4	动态内存——使用 new 和 delete	113
7.5	引用	115
7.5.1	引用的声明和操作	115
7.5.2	引用传递	118
7.5.3	返回引用	119
7.6	josephus 问题	120
	习题 7	122
第 8 章	结 构	124
8.1	结构类型	124
8.1.1	结构类型声明	124
8.1.2	结构类型变量的定义	125
8.1.3	结构类型变量的引用	127
8.1.4	重名问题	128
8.2	结构数组与指针	128
8.2.1	结构数组	128
8.2.2	结构指针	131
8.3	联合	133
8.3.1	联合的声明	133
8.3.2	联合变量的定义和使用	134
8.4	使用 typedef	135
8.5	简单链表	138
8.5.1	链表概述	138
8.5.2	链表的创建和遍历	139
8.5.3	链表的删除	141

8.5.4	链表结点的插入和添加.....	142
8.5.5	用链表求解 josephus 问题.....	144
习题 8		146
第 9 章	程序组织和编译预处理.....	149
9.1	作用域和可见性.....	149
9.1.1	函数原型作用域.....	149
9.1.2	函数作用域.....	149
9.1.3	块作用域.....	150
9.1.4	文件作用域.....	151
9.1.5	域运算符.....	151
9.2	内存区和存储类型.....	152
9.2.1	内存区.....	152
9.2.2	自动类型和寄存器类型.....	152
9.2.3	静态类型.....	153
9.2.4	外部类型.....	155
9.3	编译预处理.....	157
9.3.1	宏定义.....	157
9.3.2	文件包含命令.....	158
9.3.3	文件重复包含处理.....	159
9.4	名称空间.....	160
习题 9		163
第 10 章	类和对象.....	165
10.1	类和对象概述.....	165
10.1.1	从结构到类.....	165
10.1.2	类的声明.....	166
10.1.3	对象的定义和成员的访问.....	168
10.1.4	类作用域和成员访问权限.....	170
10.2	构造函数和析构函数.....	173
10.2.1	构造函数.....	173
10.2.2	析构函数.....	177
10.3	对象的使用.....	178
10.3.1	对象赋值和拷贝.....	178
10.3.2	浅拷贝和深拷贝.....	179
10.3.3	对象成员的初始化.....	182
10.3.4	const 对象.....	184
10.3.5	对象的生存期.....	185
10.4	综合应用实例.....	185
习题 10		190

第 11 章 数据共享和成员特性.....	193
11.1 静态成员.....	193
11.1.1 静态成员概述.....	193
11.1.2 静态数据成员.....	193
11.1.3 静态成员函数.....	195
11.2 友元.....	197
11.2.1 友元概述.....	198
11.2.2 友元函数.....	198
11.2.3 友元类.....	201
11.3 const 成员.....	203
11.4 综合应用实例.....	205
习题 11.....	209
第 12 章 继承和派生.....	211
12.1 继承和派生概述.....	211
12.1.1 继承的概念.....	211
12.1.2 继承的特性.....	212
12.1.3 派生类的定义.....	213
12.2 继承方式.....	213
12.2.1 公有继承.....	214
12.2.2 私有继承.....	216
12.2.3 保护继承.....	219
12.2.4 不同继承方式的比较.....	219
12.3 派生类的构造和析构.....	221
12.3.1 构造和析构次序.....	221
12.3.2 派生类数据成员初始化.....	223
12.3.3 基类成员的访问.....	226
12.4 二义性和虚基类.....	226
12.4.1 二义性概述.....	226
12.4.2 二义性解决方法.....	230
12.4.3 虚基类和虚继承.....	230
12.5 综合应用实例.....	232
12.5.1 类间关系.....	232
12.5.2 设计实例.....	234
习题 12.....	237
第 13 章 多 态.....	239
13.1 多态和虚函数.....	239
13.1.1 多态概述.....	239
13.1.2 虚函数定义.....	241

13.1.3	虚析构函数.....	242
13.1.4	纯虚函数和抽象类.....	244
13.2	运算符重载.....	246
13.2.1	运算符重载函数.....	246
13.2.2	运算符重载限制.....	247
13.3	典型运算符重载.....	248
13.3.1	赋值运算符的重载.....	248
13.3.2	自增自减运算符的重载.....	249
13.3.3	下标运算符重载.....	251
13.4	综合应用实例.....	253
习题 13		256
第 14 章	输入/输出流.....	258
14.1	概述.....	258
14.1.1	流和流类.....	258
14.1.2	标准流对象.....	259
14.1.3	提取和插入运算符重载.....	259
14.2	格式控制和错误处理.....	260
14.2.1	设置输出宽度和填充字符.....	261
14.2.2	控制实数显示.....	261
14.2.3	左右对齐输出.....	262
14.2.4	流的错误处理.....	263
14.3	使用输入/输出成员函数.....	264
14.3.1	输入操作的成员函数.....	264
14.3.2	输出操作的成员函数.....	266
14.4	文件流.....	267
14.4.1	文件和文件流概述.....	267
14.4.2	文件流的使用方法.....	267
14.4.3	顺序文件操作.....	270
14.4.4	随机文件操作.....	271
14.5	综合应用实例.....	275
习题 14		279
实验 1	认识 VISUAL C++ 6.0 中文版开发环境.....	280
实验 2	数据类型、运算符和表达式.....	290
实验 3	分支语句.....	294
实验 4	循环语句.....	297
实验 5	函数.....	300
实验 6	数组.....	304
实验 7	指针和引用.....	308

实验 8 结构、程序组织和编译预处理	312
实验 9 类和对象	320
实验 10 继承和派生	325
实验 11 多态	331
实验 12 输入/输出流	336
综合应用实习	342
附 录	350
附录 A 常用 C++ 库函数及类库	350
附录 B ASCII 码表	351
附录 C 格式算子	352
附录 D 格式控制成员函数	353
附录 E 运算符优先级和结合性	353
附录 F 从 C 到 C++	354

第 1 章

C++概述



C++是在 20 世纪 80 年代初期由贝尔实验室设计的一种在 C 语言的基础上增加了对面向对象程序设计支持的高级语言，它是目前应用最为广泛的编程语言。

1.1 C++程序创建

使用 C++高级语言编写的程序称为**源程序**。由于计算机能识别和执行的是由 0 和 1 组成的二进制指令，称为**机器代码**，因而 C++源程序是不能被计算机直接执行的，必须转换成机器代码才能被计算机执行。这个转换过程就是编译器对源代码进行**编译**和**连接**的过程，如图 1.1 所示。

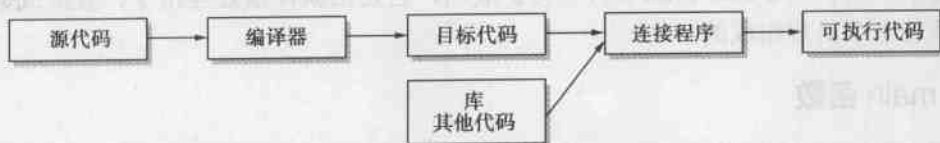


图 1.1 C++程序创建过程

例如：编写计算圆面积的程序，圆的半径从键盘输入。

(1) 用 Windows XP 记事本输入下列内容，文件名指定为“Ex_Simple.cpp”。

注意：扩展名“.cpp”不能省略，cpp 是 C Plus Plus 的缩写，即“C++”的意思。将文件定位到“D:\C++程序\第 1 章”文件夹中。

如果进入开发环境（例如：Visual C++6.0），则可以在开发环境下直接输入即可。

```
/* 第一个简单的 C++程序 */
#include <iostream.h>
int main()
{
    double r, area;           // 定义变量
    cout<<"输入圆的半径:";   // 显示提示信息
    cin>>r;                   // 从键盘上输入变量 r 的值
    area = 3.14159 * r * r;    // 计算面积
    cout<<"圆的面积为:"<<area<<"\n"; // 输出面积
    return 0;                 // 指定返回值
}
```

(2) 编译连接。如果是 Visual C++6.0 开发环境，对 Ex_Simple 进行编译、连接，同时在输出窗口中显示编译的有关信息，当出现：

Ex_Simple.exe - 0 error(s), 0 warning(s)

表示 Ex_Simple.exe（可执行文件）已经正确无误地生成了。如果程序有错误，则显示出错信息，用户根据出错信息查找错误，更正错误后重新编译，直到正确为止。

(3) 运行。在 Visual C++开发环境运行，会在 DOS 窗口执行 Ex_Simple.exe 文件，如下所示：



此时等待用户输入一个数字。当输入 10 并按 Enter 键后，控制台窗口显示为：



其中，“Press any key to continue”是 Visual C++ 自动加上去的，表示 Ex_Simple 运行后，按一个任意键将返回到 Visual C++ 开发环境，这就是 C++ 程序的创建、编连和运行过程。

1.2 C++ 程序结构

一个程序是由若干个程序源文件组成的。为了与其他语言相区别，每一个 C++ 程序源文件通常是以“.cpp”（C Plus Plus, C++）为扩展名，它是由编译预处理指令、数据或数据结构定义以及若干个函数组成的。

1.2.1 main 函数

如图 1.2 所示的 Ex_Simple.cpp 程序代码。代码中，main 表示**主函数**，由于每一个程序执行时都必须从 main 开始，而不管该函数在整个程序中的具体位置，因此每一个 C++ 程序或由多个源文件组成的 C++ 项目都必须包含一个且只有一个 main 函数。

```

1// [Ex_Simple] 一个简单的C++程序
2/* 第一个简单的C++程序 */
3#include <iostream.h>
4int main()
5{
6    double r, area;           // 定义变量
7    cout<<"输入圆的半径: ";  // 显示提示信息
8    cin>>r;                   // 从键盘上输入变量r的值
9    area = 3.14159 * r * r;    // 计算面积
10   cout<<"圆的面积为: "<<area<<"\n"; // 输出面积
11   return 0;                 // 指定返回值
12}

```

图 1.2 Ex_Simple.cpp 的程序代码

在 main 函数代码中，“int main()”称为 main 函数的**函数头**，函数头下面是用一对花括号（“{”和“}”）括起来的部分，称为 main 函数的**函数体**，函数体中包括若干条语句（按书写次序依次顺序执行），每一条语句都由分号“;”结束。由于 main 函数名的前面有一个 int，它表示 main 函数的类型是整型，需在函数体中使用关键字 return，用来将其后面的值作为函数的返回值。由于 return 语句运行后，函数体 return 后面的语句不再被执行，因此除非想要函数提前结束，否则 return 语句应写在函数体的最后。

main 函数体的第 1 条（行号为 6）语句是用来定义两个双精度实型（double）变量 r 和 area 的；第 2 条（行号为 7）语句是一条输出语句，它将双引号中的内容（即字符串）输出到屏幕上，cout 表示标准输出流对象（屏幕），“<<”是插入符，它将后面的内容插入到 cout 中，即输出到屏幕上；第 3 条（行号为 8）语句是一条输入语句，cin 表示标准输入流对象（键盘），“>>”是提取符，用来将用户键入的内容保存到后面的变量 r 中；最后一条（行号为 10）语句是采用多个“<<”将字符串和变量 area 的内容输出到屏幕中，后面的“\n”是换行符，即在内容输出后回车换行。

1.2.2 头文件包含

行号为 3 的代码是 C++ 文件包含 `#include` 的编译指令，称为**预处理指令**。`#include` 后面的 `iostream.h` 是 C++ 编译器自带的文件，称为 **C++ 库文件**，它定义了标准输入/输出流的相关数据及其操作，由于程序用到了输入/输出流对象 `cin` 和 `cout`，因而需要用 `#include` 将其合并到程序中，又由于它们总是被放置在源程序文件的起始处，所以这些文件被称为**头文件** (Header File)。C++ 编译器自带了许多这样的头文件，每个头文件都支持一组特定的“工具”，用于实现基本输入/输出、数值计算、字符串处理等方面的操作。

在 C++ 中，头文件包含两种格式。一是将文件名用尖括号“<>”括起来，用来包含那些由编译系统提供的存放在指定子文件夹中的头文件，这称为**标准方式**。二是将文件名用双引号括起来的方式，称为**用户方式**。这种方式下，系统先在用户当前工作文件夹中查找要包含的文件，若找不到再按标准方式查找（即再按尖括号的方式查找）。所以，一般来说，用尖括号的方式来包含编译器自带的头文件，以节省查找时间；而用双引号来包含用户自己编写的头文件。

1.2.3 新头文件格式和名称空间

由于 `iostream.h` 是 C++ 的头文件，因此这些文件是以“.h”为扩展名，以便与其他文件类型相区别，但这是 C 语言的头文件格式。尽管 ANSI/ISO C++ 仍支持这种头文件格式，但已不建议再采用，即包含头文件中不应再有“.h”这个扩展名，而应使用 C++ 的 `iostream`。例如：

```
#include <iostream>
```

但为了使 `iostream` 中的定义对程序有效，还需使用下面名称空间编译指令来指定：

```
using namespace std;
```

// 注意不要漏掉后面的分号

`using` 是一个在代码编译之前处理的指令。`namespace` 称为**名称空间**，它是 ANSI/ISO C++ 一个新的特性，用于解决在程序中同名标识存在的潜在危机。由于 `iostream` 是 ANSI/ISO C++ 标准组件库，它所定义的类、函数和变量均放入名称空间 `std` 中，因此需要在程序文件的开始位置处指定“`using namespace std;`”，以便能被后面的程序所用。

事实上，`cin` 和 `cout` 就是 `std` 中已定义的流对象，若不使用“`using namespace std;`”，则还可采用下列两种方式来指定。

第 1 种方式是在使用前用下列代码来指定：

```
using std::cout;
```

// 指定以后的程序中可以使用 `cout` 对象

```
using std::cin;
```

// 指定以后的程序中可以使用 `cin` 对象

第 2 种方式是在调用时指定它所属的名称空间，如下述格式：

```
std::cout<<"输入圆的半径: ";
```

// :: 是域作用运算符，表示 `cout` 是 `std` 域中的对象

```
std::cin>>r;
```

显然，用下列两句代码来替代 C 语言风格的头文件包含 `#include <iostream.h>` 是一种最为简洁的做法，这也是本书所采用的方法。

```
#include <iostream>
```

```
using namespace std;
```

1.2.4 注释

程序 `Ex_Simple` 中的“`/*...*/`”之间的内容或“`//`”开始一直到行尾的内容是用来注释的。在 C++ 中，“`/*...*/`”是用来实现多行的注释，它是将由“`/*`”开头到“`*/`”结尾之间所

有内容均视为注释，称为**块注释**。块注释（“/*...*/”）的注解方式可以出现在程序中的任何位置，包括在语句或表达式之间。而“//”只能实现单行的注释，它是将“//”开始一直到行尾的内容作为注释，称为**行注释**。

注释的目的只是为了提高程序的可读性，对编译和运行并不起作用。正是因为这一点，所注释的内容既可以用中文来表示，也可以用英文来说明，只要便于理解就行。

一般来说，注释应在编程的过程中同时进行，不要指望程序编制完成后再补写注释。那样只会多花好几倍的时间，更为严重的是，时间长了以后甚至会读不懂自己编写的程序。

通常，必要的注释内容应包含：

(1) 在源文件头部进行必要的源程序的**总体注释**：版权说明、版本号、生成日期、作者、内容、功能、与其他文件的关系、修改日志等，头文件的注释中还应包含函数功能简要说明。

(2) 在函数的头部进行必要的**函数注释**：函数的目的/功能、输入参数、输出参数、返回值、调用关系（函数、表）等。

(3) **其他的少量注释**：如全局变量的功能、取值范围等。千万不要陈述那些一目了然的内容，否则会使注释的效果适得其反。

1.2.5 C++程序组成

下面再看两个 C++ 程序。

 **【例 Ex_Simple1】** 在屏幕上输出一个由星号形成的三角形

```
// 输出星号的三角形阵列
#include <iostream>
using namespace std;
void DoDraw(int num);           // 声明一个全局函数
int main()
{
    int num = 5;                // 定义并初始化变量
    DoDraw(num);                // 函数的调用
    return 0;                   // 指定返回值
}
void DoDraw(int num)            // 函数的定义
{
    for (int i = 0; i < num; i++) // 循环语句
    {
        for (int j = 0; j <= i; j++) cout << '*';
        cout << '\n';
    }
}
```

本程序包括两个函数：主函数 main 和被调用的函数 DoDraw。DoDraw 函数是在屏幕上输出星号的三角形阵列，这个阵列的行数以及每行星号的个数由 num 决定。程序运行的结果如下：

```
*
**
***
****
*****
```

在以后的 C++ 程序运行结果中，本书不再完整显示其控制台窗口，也不再显示“Press any key to continue”，仅显示控制台窗口中运行结果部分，并加以单线阴影边框，本书做此约定。