全美经典
学习指导系列

C++ 编程习题与解答

PROGRAMMING WITH C++

最佳的复习资料，实用的辅助教材

与国外高校计算机水平保持同步

为考研和出国深造奠定坚实基础

John R. Hubbard 著
徐漫江 王栋 何路 等译

全球销售超过3000万册！

机械工业出版社
China Machine Press

中信出版社
CITIC PUBLISHING HOUSE

全美经典
学习指导系列

C++ 编程习题与解答

PROGRAMMING WITH C++

John R. Hubbard 著
徐漫江 王栋 何路 等译

机械工业出版社
China Machine Press
中信出版社
CITIC PUBLISHING HOUSE

本书由浅入深地介绍了C++语言的各个方面，并在所涉及的各个知识点给出了详细的例子，使读者能够更容易了解C++的内容。无论读者是从未接触过C++语言的新手，还是对C++语言有一定经验的开发人员，本书都能使你对这门编程语言有全面而系统的了解。

本书版权登记号：图字：01-2001-5204

图书在版编目（CIP）数据

C++编程习题与解答/（美）哈伯德（Hubbard, J. R.）著；徐漫江等译. - 北京：机械工业出版社，2002.8

（全美经典学习指导系列）

书名原文：Prorgamming with C++, Second Edition

ISBN 7-111-10821-3

Ⅰ. C… Ⅱ. ① 哈… ② J… ③ 徐… Ⅲ. C语言 - 程序设计 - 解题 Ⅳ. TP312-44

中国版本图书馆CIP数据核字（2002）第062694号

机械工业出版社（北京市西城区百万庄大街22号 邮政编码100037）

责任编辑：华章

北京忠信诚印刷厂印刷·新华书店北京发行所发行

2002年8月第1版第1次印刷

787mm × 1092mm 1/16 · 28印张

印数：0 001-5 000册

定价：39.00元

作 者 序

像所有的 Schaum 系列丛书中的书籍一样，本书主要是为方便读者自学而编写的，适合与一门正规的C++编程语言或计算机科学的课程一同使用。同时也很适于独立学习或作为一本参考书使用。

本书包括了 200 多个精彩例题和精心组织的有详细解答的习题。通过学习书中精心编写并做了详尽解释的例题，读者可以熟练掌握数据结构的基本原则，这也正是本书的编写意图。

C++是早在 20 世纪 80 年代由 Bjarne Srtoustrup 开发的，它基于 C 和 Simula，是当今最流行的面向对象的编程语言之一。在 1998 年，美国国家标准化协会（ANSI）和国际标准化组织（ISO）为该语言制定了标准，这个新的 ANSI/ISO 标准包含了功能强大的标准模板库（STL），本书是完全遵照该标准而编写的。

虽然很多要学习 C++的人已经有了一些编程经验，作者还是假定本书的读者没有任何编程经验，而将 C++看做是读者所学习的第一门编程语言。这样，那些具有编程经验的读者可以跳过前面的几章。

C++是一门很难学的语言，原因至少有两个：第一，它继承了 C 语言的表达体制，很多含义很模糊；第二，作为一门面向对象的编程语言，普遍使用了类和模板的概念，这对以前没有这方面思想的读者而言是个极大的挑战。本书的编写意图是为那些刚刚起步的程序员们克服这些障碍而提供所需的帮助。

本书中的所有例题和问题，包括附加问题的源代码，都可以从以下网址下载：http://projectEuclid.net/schaums，http://www.richmond.edu/~hubbard/schaums，http://hubbards.org/schaums，http://jhubbard.net/schaums。另外，本书的修正和附录也可以在这些站点找到。

我想感谢所有的朋友、同事、学生和为本书提供了许多批评和建议的 McGraw-Hill 的人员，包括 John Aliano、Arthur Biderman、Francis Minhthang Bui、Al Dawson、Peter Dailey、Mohammed El-Beltagy、Gary Galvez、Libbie Geiger、Sergei Gorlatch、Chris Hames、John Troncale、Maureen Walker、Stefan Wentzig 和 Nat withers，在此对他们的建议和所做的调试工作表示衷心的感谢。

特别感谢我的妻子和工作伙伴 Anita H. Hubbard，感谢她为本书所提供的建议、鼓励和创造性意见，书中的许多源程序都是她编写的。

目　录

第1章　C++程序设计基础

程序就是能够由计算机执行的一组指令序列，所有的程序都由某种程序设计语言写成。C++是功能最强大的程序设计语言之一，程序设计人员使用它能够写出高效、结构化及面向对象的程序。

1.1　入门

为了编写及运行C++程序，在计算机中必须已经安装有文本编辑器及C++编译器。文本编辑器是一种可以在计算机上创建及编辑文本文件的软件，程序设计人员使用它编写某种程序设计语言的程序。编译器是一种将程序转化成操作系统能够执行的机器语言（称为二进制编码）的一种软件。这种转化过程称做“编译”。一个C++编译器可以将C++程序编译成机器语言。

如果计算机正运行着微软公司Windows操作系统的一个版本（如Windows 98或Windows 2000），则其中已经带有两个文本编辑器：写字板和记事本。可以通过“开始”按钮来启动。在Windows 98系统中，它们在“附件”里。

Windows操作系统没有内建的C++编译器，所以除非已经有人在机器上安装过C++编译器，否则必须自己安装。如果使用的是由别人管理的运行Windows系统的计算机（比如公司或学校的信息服务部门），则C++编译器可能已经安装好了。使用“开始”按钮，在“程序”目录下查找Borland C++ Builder，Metrowerks CodeWarrior，Microsoft Visual C++或任何名字中包含C++的程序即可。如果不得不自己购买C++编译器，则可到网上查找所有上面提到的编译器中不太贵的版本。这些通常是指IDE（集成开发环境），因为它们包含特定的文本编辑工具及调试工具。

如果使用的计算机是运行着个人版本UNIX操作系统的工作站（如运行Sun Solaris的SPARC工作站），它可能已经安装有C++编译器。一个简便的寻找方法就是建立如例1.1所示的程序，将它命名为hello.c，并用以下命令尝试编译它：

```
CC  hello
```

自由软件基金有一套UNIX软件，称为“GNU”软件，它们可以从以下网址免费下载：

```
http://www.gnu.org/software/software.html
```

可以使用包含C++编译器的GCC包和Emacs编辑器。在DOS系统下，可以使用包含C++编译器的DJGPP。

1.2 程序实例

假设已有一个文本编辑器来编写 C++程序，也有一个 C++编译器来编译它们。若在个人计算机上使用集成开发环境如 Borland C++ Builder，则可以通过按一个合适的按钮来编译与运行程序。其他的系统可能会要求在命令行中运行程序。如果是这样的话，则可像键入命令一样键入文件名。例如，若源代码在名为 hello.cpp 的文件中，则键入：

```
hello
```

就可以在编译完成之后运行这个程序。

在编写 C++程序时，必须注意 C++是大小写敏感的，也就是说main ()不同于Main ()。最安全的方法是除非必须大写，否则全部采用小写。

例 1.1 “Hello, World”程序

本例简单地打印出“Hello, World!”：

```
#include <iostream>
int main ()
{ std:: cout << "Hello, World! \n";
}
```

代码的第 1 行是一条预处理伪指令，它告诉 C++编译器在什么地方寻找第 3 行中使用的 std:: cout 对象的定义。标识符 iostream 是标准 C++库中一个文件的名字。所有用到标准输入输出的 C++程序都必须包含这个预处理伪指令。注意所需的符号：必须用井号 # 指明 include 是一个预处理伪指令；用符号 < > 指明 iostream（它代表输入输出流）是一个标准 C++库文件的名称。表达式 <iostream> 称做标准头。

第 2 行也是所有 C++程序中必须有的。它声明了程序在何处开始。标识符 main 是一个函数的名称，称为程序的主函数。所有的 C++程序都必须有且仅有一个 main () 函数。main 后面所跟的圆括号表明这是一个函数。关键词 int 是 C++中的一种数据类型，它代表整数。在这里使用它表明 main () 函数返回值的类型。当程序结束运行时，它可以向操作系统返回一个表示某种结果状态的整数类型值。

最后两行组成了程序的体，程序体是包含在花括号“{}”中的一系列程序语句。在本例中只有一条语句：

```
std:: cout << "Hello, World! \n";
```

它说明将字符串“Hello, World! \n”发送给标准输出流 std:: cout 对象。符号“<<”代表 C++的输出运算符。当这条语句被执行时，包在引号“”“”中的字符被送至标准输出设备，一般是计算机屏幕。最后两个字符 \n 代表换行符。当输出设备遇到这个字符时，它向前移动至屏幕上文件下一行的开始处。最后，注意所有的程序语句都必须以分号“;”结尾。

注意，例 1.1 的程序是如何在四行中进行安排的，这种格式有助于阅读。C ++ 编译器忽略这些格式，它所读到的内容就如程序全部写在一行中，如下所示：

```
# include < iostream >
int main ( ) { std:: cout < < "Hello, World!  \ n";}
```

编译器会忽略空格符，除非它们被用于分开标识符如：

```
int main
```

注意预编译命令必须在程序之前的单独一行中。

例 1.2　另一个“Hello, World”程序

本例输出同例 1.1：

```
# include < iostream >
using namespace std;
int main ( )
{ // 打印 "Hello, World!";
  cout < < "Hello, World!  \ n";
  return 0;
}
```

第 2 行：

```
using namespace std;
```

告诉 C ++ 编译器在需要使用前缀的地方使用 std::。它表明可以用 cout 代替 std:: cout。这使大型程序比较容易阅读。

第 4 行：

```
{ // 打印 "Hello, World!";
```

包含注释“打印"Hello, World!";”。程序中的注释是在程序编译之前被预处理去掉的一串字符，它们被用于向阅读者提供解释。在 C ++ 中，所有从双斜杠“//”至行尾之间的文字被视为注释。也可以像下面一样使用 C 语言风格的注释：

```
{ /* 打印 "Hello, World!"; */
```

C 语言风格的注释（由程序设计语言“C”引入）是指在符号“/ *”与“ * /”间的任何字符串，这些字符串可以跨过多行。

第 6 行：

```
return 0;
```

在标准 C ++ 的 main ( ) 中是可选的。在这里包含这行只是因为有一些编译器希望main ( ) 函数在最后一行包含它。

名字空间是一个被命名的定义集合。当一个名字空间中定义的对象被用于此名字空间以

外时，或者他们以所属名字空间为前缀，或者必须在采用“using namespace”语句预先定义的块中。名字空间允许程序对不同的对象使用同一个名字，就像不同的人可以有相同的名字一样。cout 对象是在 < iostream > 头文件中 std（代表标准）名字空间中定义的。

在本书的其余部分，所有的程序都假定以下面两行开始：

```
# include < iostream >
using namespace std;
```

在例子中这必要的两行将被省略，并且也将在 main（）函数中省略：

```
return 0;
```

如果使用要求这行语句的编译器（如 Microsoft Visual C ++），则应将它包含进程序。

1.3 输出运算符

符号“< <”在 C ++ 中被称为输出运算符（也被称为 put 运算符或流插入运算符）。它将值插入到左边的输出流中。经常会使用 cout 输出流，它通常指向计算机屏幕。所以语句：

```
cout < < 66;
```

将在屏幕上显示数字 66。

运算符将动作作用于一个或多个对象。输出运算符 < < 完成将其右边表达式的值送到左边流的任务。由于这个动作的方向看起来是从右向左的，因此选用 < < 来表示。它用指向左边的箭头进行提醒。

对象 cout 被称为一个“流”，因为传送给它的输出数据象水流一样流动。如果有多个数据被插入到 cout 流中，它们就像落入水中一样一个接一个排成一条线。这就像树叶从树上落入真正的水流中。插入 cout 流中的数据按照这样的顺序被显示在屏幕上。

例 1.3 又一个“Hello, World”程序

本例输出同例 1.1：

```
int main ()
{ // 打印 "Hello, World!":
  cout < < "Hel" < < "lo, Wo" < < "rld!" < < endl;
}
```

这里输出运算符被使用了四次，按顺序将四个对象"Hel","lo, Wo","rld!"及"endl"放入输出流。前三个对象是字符串，它们被连接（头尾相连排成一列）在一起组成一个单个字符串"Hello, World"。第四个对象是流操作对象 endl（意思是“行尾”），它的作用相当于在字符串后添加行尾符'\n'：将打印光标送到下一行的开始处，并且清空缓冲区。

1.4　字符与文字

例 1.3 中的"Hel"、"lo，Wo"及"rld!"三个对象被称为“字符串文字”。每个文字由一组被引号界定的字符序列组成。

字符是记录有意义书写的基本符号。英语作家使用拉丁字母表中的 26 个小写字母和 26 个大写字母，同时还有 10 个阿拉伯数字及其他一些符号。字符在计算机中是以整数形式存储的。字符集码就是列出字符集中每个字符的整数表示值的表。在 2000 年前使用最广泛的字符集是 ASCII 码，见附录 A 中。它是美国信息交换标准码的首字母缩写（读作“as-key”）。

换行符"\n"是不可打印字符之一。它是由反斜杠"\"与字母 n 组成的单个字符。其他还有一些字符也采用相同的方法组成，包括制表符"\t"及报警符"\a"。反斜杠还用于指示两个无法在文字中使用的可打印字符，引号"\""及反斜杠自身"\\"。

字符在程序语句中可以作为字符串文字的一部分，也可以作为单独的对象。当单独使用时，必须作为字符常量。字符常量是由单引号括起来的单个字符。作为单独的对象，字符常量可以像字符串文字一样被输出。

例 1.4　“Hello, World”程序的第四个版本

本例输出同例 1.1：

```
int main ()
{ // 打印 "Hello, World!":
  cout << "Hello, W" << 'o' << "rld" << '!' << '\n';
}
```

这个程序表明输出运算符可以像处理字符串文字一样很好的处理字符。三个单独的字符'o','!'及'\n'以与两个字符串文字"Hello, W"及"rld"相同的方式被连接到输出上。

例 1.5　在标准输出流中插入数字文字

```
int main ()
{ // 打印 "The Millennium ends Dec 31 2000.":
  cout << "The Millennium ends Dec " << 3 << 1 << ' ' << 2000 << endl;
}
```

当 3 和 2000 这样的数字文字被送至输出流时，它们被自动转换成字符串文字，并以与字符相同的方式被连接至输出流。注意，为了不使数字靠在一起，必须明确地传送空格字符' '。

1.5　变量及其他声明

变量是代表计算机内存中一个存储位置的符号。存储在此位置的信息被称做这个变量的值。变量得到值的通用方法是赋值，语法形式是：

变量 = 表达式;

表达式先被求值并将结果赋值给变量。等号“=”在 C++ 中是赋值运算符。

例 1.6 使用整型变量

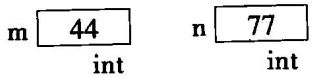
本例中，整数 44 被赋值给变量 m，表达式 m + 33 被赋值给变量 n:

```
int main ()
{ // 打印 "m = 44 and n = 77":
  int m, n;
  m = 44;    // 将值 44 赋给变量 m
  cout < < "m = " < < m;
  n = m + 33;    // 将值 77 赋给变量 n
  cout < < " and n = " < < n < < endl;
}
```

这个程序的输出在以下的阴影框中:

```
m = 44 and n = 77
```

可以这样看变量 m 和 n:



变量 m 就像一个邮箱，它的名称 m 像邮箱上的地址；它的值 44 像邮箱内的物品；它的类型 int 像合法性检查，以确保只有哪些东西可以放入，类型 int 表明这个变量只拥有整数值。

注意在这个例子中，变量 m 和 n 是在一行中声明的。具有相同类型的任意多个变量都可以像这样声明在一行之中。

C++ 中所有变量在使用前都必须声明，语法是:

```
specifier type name initializer;
```

specifier 是一个可选的关键字，如 constant；type 是 C++ 的数据类型之一，如 int；name 是变量的名称；initializer 是可选的初始化子句，如“=44”(见 1.7 节)。

声明变量的作用是在程序中引入一个名称，也就是告诉编译器一个名称的意思。类型 type 告诉编译器这个变量值的范围以及可以对它进行哪些操作。

变量在程序中声明的位置决定了这个变量的作用域：即变量可以在程序中的哪部分使用。通常，变量的作用域是从它被声明处到包含它的声明及操作的直接块的结束。

1.6 程序标记

计算机程序是由一系列称为“标记”的单元组成。这些标记包括关键字如“int”；标识符如“main”；符号如“{”；运算符如“< <”。当编译程序时，编译器将检查源代码，将它分析成标记。如果它发现意外的标记或未发现期望的标记，它将中断编译并给出错误信息。比如，如果忘记在一行的结尾加上分号，则错误信息会指出缺少分号。一些语法错误，如缺