北京科海培训中心

·精华·简明·实用系列丛书
Made Simple Books

C++简明教程

（英）Conor Sexton 著

张 红 译

机械工业出版社

著作权合同登记号:图字 01-98-1774

内 容 提 要

C++简明教程对如何使用 ANSI C++语言编程进行了深入浅出的阐述。本书语言轻松,行文简练,重点俱全。目的在于能帮助读者在最短的时间内取得最大的进步。

全书包含 C++语言的所有精华。第 1 章概述 C++语言的基本内容;第 2 章讲述 C++中类的规则;第 3 章讨论类的作用,其中有重载操作符和构造函数等;第 4 章讲述 C++中的继承;第 5 章为最新引入的异常处理、名字空间等;第 6 章为 C++中的输入输出操作(包括 I/O);第 7 章为全书所有习题的答案。

读者对象:C++程序设计初学者及选修 C++程序设计课程的学生。

图书在版编目(CIP)数据

C++简明教程/(英) 塞克斯顿(Sexton,C.) 著;
张红译. —北京:机械工业出版社,1998.12
(计算机精华·简明·实用系列丛书)
书名原文:C++ Programming Made Simple
ISBN 7-111-06788-6

Ⅰ.C… Ⅱ.①塞… ②张… Ⅲ.C 语言-程序设计-教材
Ⅳ.TP312

中国版本图书馆 CIP 数据核字(98)第 35195 号

出版人:马九荣 (北京市百万庄大街 22 号 邮政编码 100037)
责任编辑:科培 责任校对:成昊
门头沟胶印厂印刷·新华书店北京发行所发行
1998 年 12 月第 1 版·1999 年 6 月第 2 次印刷
787mm×1092mm 1/16·12 印张·232 千字
5001—8000 册
定 价:18.00 元

计算机精华·简明·实用系列丛书

Made Simple Books

丛书序

在计算机新技术迅猛发展、新知识应接不暇、新软件层出不穷的今天，对学会操作电脑的人需要拓宽使用面，让电脑发挥真正的作用。对有经验的用户，跟上潮流的发展而不落伍，也需要不断地更新自己的知识。

我们热情慎重地向广大读者推荐这套布局谋篇上独具匠心、内容精辟、讲叙简明而又实用的系列丛书；这套丛书取自英国非常畅销的“Made Simple”系列中的一部分，透过此系列让人感受到原作者的写作水平，对要介绍的软件、语言和系统的深刻理解，以及作者群体学术的严谨和扎实。“Made Simple”顾名思义使问题简单化，也就是一种将厚书写薄的丛书。每一本书抓住重点，将讲述的对象介绍得简明扼要，通俗易懂。无论是介绍操作系统，还是介绍编程语言或是开发环境，都坚定不移地遵循了这一原则。

- **书不在厚而在于精**。
  此系列，每本小册子不足 200 页，每本书讲解一种软件或是一门编程语言，内容相对独立，使读者能迅速定位自己的需求。每个问题辅以三两实例、言简意赅、点到为止。最为可贵的是作者不搞“大而全”，而是直书精要之处，将基本概念、难点、常用方法及相关技巧一一展示给读者。
- **语言简朴，引导有方**。
  本套丛书是很好的教材，特别是针对初学者，尤为难得。对关键概念，作者舍得花笔墨，用通俗的语言加以阐释；枝节之处，则当删则删，当漏则漏；而且全书都是用简明的图示来表达要点，让读者学得轻松、容易树立信心。
- **实用性（这是许多书称有而最不容易达到的）**。
  此系列丛书的实用性称得上扎扎实实。全书以问题、任务为主线，辅以大量实例而构成。这些例子实用性强，并且这些例子并不单一，往往例子彼此相关，最后可能组成的是一个比较大的程序，或是一个复合技术。这与我们常见到的一些书，往往一个例子表达一个简单的功能，彼此无关，无助于读者构筑自己较复杂的应用程序。此系列对例子的解释也是用图示表达，其中读者可自行修改、替换。

此系列丛书大部分每章末均有习题，书后有习题答案，这些习题也很有特点，它不是简单地复述前面的概念，也不是前面例子的翻版，而比例子更具有创造性、思考和提高的余地和价值。这是很难得的，也是一本好教材的内涵所在。

我们在翻译此系列丛书时，尽可能地聘请有经验、高水平的译者，目的是为了保持原丛书鲜明的风格。翻译了其中 JAVA 和 UNIX 两本书的钟向群先生认为："此套丛书很像一个隽永精品集，读者极易理解，却又回味无穷，科技书籍中有此效果者廖廖"，翻译 Visual Basic 和 Visual C＋＋两书的熊桂喜教授认为："这是一套非常难得的轻松型教材，值得推荐与学习"。

在当今电脑书籍让人眼花缭乱，汗牛充栋的现状下，指导读者发现和正确选择一些好的读本是我们的义务，也是我们的责任，为读者编写诸如此类的教材是我们工作的方向。

希望此系列丛书帮助你开启电脑知识和程序设计的大门，相信读者是好书真正的评判者。欢迎来函来电联系与指正。

欢迎选购：

《C＋＋简明教程》
《VISUAL BASIC 简明教程》
《VISUAL C＋＋简明教程》
《JAVA 简明教程》
《PASCAL 简明教程》
《DELPHI 简明教程》
《UNIX 简明教程》
《Windows NT 简明教程》
《多媒体简明教程》
《硬盘管理简明教程》

科海丛书编译委员会
1998 年 10 月

前　言

《C++简明教程》一书对如何使用 ANSI C++语言编程进行了深入浅出的阐述。本书语言轻松，尽量避开 C++程序设计中比较复杂的内容，将重点置于使 C++程序员能尽快入门。

本书适合作为入门指南，其适用对象是 C++程序设计初学者以及选修程序设计课程的学生。本书专门为选修这门课的学生们设计了 20 个左右有趣而实用的练习。

全书覆盖 C++的所有精华，但没有涉及它的一些"死角"，比如变量赋值和初始化的详细过程。它是一本一般的入门参考书，但不是为 C++高级程序员或 C 编译员准备的。

本书在讲述 C++之前并没有介绍 C 语言，因为这样就会超过了本书的篇幅。如果你对 C 语言一无所知，请你就此停止，先去阅读本丛书中的《C 简明教程》这本书。

当 Butterworth-Heinemann 请我写这本书时，我感觉到最大的挑战是本书在份量上（文字和图解说明）要比我以前的 C 及 C++著作轻得多。Made Simple 的目的在于帮助读者能在最短的时间内使用最少的精力取得最大的进步。对这样一本书来说，第 1 章的内容就应该对 C++语言最基本的部分做一浏览，让读者能写出一个完整的程序。

第 2 章对编写 C++类的规则进行重点解释。第 3 章讨论类的作用，其中有重载操作符和构造函数等。第 4 章对 C++继承中最重要的内容及其特点进行了解释。第 5 章介绍了最新引入 C++的异常处理、名字空间以及运行时类型识别等。第 6 章解释了如何在 C++程序中进行包括文件 I/O 在内的输入输出操作。本书在每章的最后都有 3～4 个练习，并在第 7 章对这些问题都一一给出了答案。

我很高兴写这本书，它的读者对象是初学者而不是专家。本书语言轻松，行文简练，重点俱全。我希望这本书能对你有所帮助！

Conor Sexton

1997 年 1 月

目　录

第1章 快速学会使用C++

1.1 C++语言的背景

C++是由C语言派生出来的一种面向对象程序设计语言，可以说，C是C++的一个子集。实际上，任何一个用新的方式编写(特别是使用了新型函数头文件)的ISO C程序，只要它避开使用某些C++保留字，其本身就是一个C++程序，尽管这些程序不是面向对象的。

C++主要用于专业计算机程序设计。它除了可以实现C语言的所有功能之外，还有其他一些自己独特的新功能，它可以和C语言一样用来开发专业性的软件，比如操作系统、图形界面、通信驱动程序和数据库管理程序。

C++首先由AT&T Bell实验室的Bjarne Stroustrup博士在20世纪80年代初期提出。最初被称为带"类"的C，这表明C++语言在语法上对C的主要扩展是对类结构的实现。

当开始开发C++时，C已经使用了多年，其语法的某些弱点已经变得明显起来，而且一些新技术，尤其是面向对象的程序设计技术无法用C语言实现。为此，ANSI在1983年成立了一个委员会X3J11对C语言和运行库进行标准化。ANSI C标准(美国国家标准X3.159-1989)在1989年被ANSI采用并在1990年被ISO9899:1990 C标准取代。早期C++语言的一些创新，尤其是函数原型法，被合并到ANSI和ISO C标准中(它们在技术上是完全等同的)。

第一个C++"编译器"由AT&T发布，它实际上是一个前端翻译器，被称为cfront，它的作用是把C++代码转换成C代码。C代码然后被C编译器和本地装入程序处理成可执行代码。第一个真正的C++编译器于1988年诞生；目前比较流行的是由Borland，IBM，Microsoft和其他一些公司开发的编译器和集成开发环境(IDEs)，在PC机和工作站环境上更是如此。

C++2.0版本诞生于1989年，这是一个重大进步，其中主要包括类的多继承，3.0版本(1991年)引入了模板。目前的版本是4.0，其中包括异常处理、运行时类型识别(RTTI)和名字空间。ANSI C++标准草案是以4.0版本为基础的，最后的ANSI C++标准在1997年被批准。

C++在以下这些方面对C进行了扩展：

- 以类定义的形式实现了"对象"，类中不仅包括C结构中的数据定义，而

且还包括对数据进行操作的函数的声明和定义，这种把数据和函数封装在一个对象中的技术是C++的一个主要革新。

- 类的实例可以用构造函数和析构函数自动进行初始化和释放，这就消除了程序的初始化错误。
- C++中类的定义方式增强了数据隐藏性，在默认情况下，类中定义的数据只能被类中的成员函数引用。外部(客户)程序在使用类时不会改变类的内部实现，它们只能通过调用成员函数来访问类。
- C++允许对操作符和函数进行重载。对一个函数的多个定义可以采用相同的名字，编译器可以在函数调用时识别出合适的定义形式。像"++"和"->"这样的普通操作符也可以被重载为加法的含义。
- C++允许类类型的特征——数据和函数——被子类所继承，子类也称为派生类，子类反过来可以添加更多的数据和函数定义。这就鼓励了用共享类库的方式重用已有代码，从而节省了软件开发过程中的费用。多继承允许派生类从多个基类中继承特征。
- C++允许类定义虚函数：一个函数有多个定义，在程序运行时决定使用哪个定义。这叫做多态性，它的意思就是在运行时才从函数定义中进行选择，这称为后期绑定(late binding)或动态聚束(dynamic binding)。
- 可以定义模板类，它允许在代码不变的情况下，用不同类型的数据定义同一个类的不同实例。这更进一步提高了代码的重用率。

C++中面向对象程序设计的工具是类、继承和虚函数。这些工具使得C++语言非常适合于编写处理大量相关对象的软件。也许C++最合适的应用是用来实现图形用户界面(GUIs)，其中许多不同但相关的对象可以在屏幕上表现出来，而且可以交互。运用面向对象的方法，C++把这些对象存储在类层次结构中，利用虚函数给这些对象提供一个通用界面(比如绘图对象)，它可以使程序员不必了解如何使用对象的细节，这就使程序员开发和维护程序变得方便起来，而且使错误(bugs)进入已有程序代码的概率减小。

和C语言相比，C++有许多较小的语法改进。它提供了一种新的引用机制，从而对C语言中的指针间接引用进行了补充。C++简化了动态分配和释放内存的过程，并且实现了一种新I/O流库，它用层次分明的类对输入和输出流进行了定义。

C++是C的扩展，它通过简化设计，使得软件能够更好地反映真实世界，降低代码的长度和复杂性，从而增加代码的可靠性。

以上已对C++的内容做了足够的介绍，让我们开始写第一个程序吧！

1.2　空操作程序

最小的 C++程序和 C 语言中最小的程序一样，这就是：

```
main () {}
```

这是一个完整的 C++程序。每个 C++程序必须包括一个或多个函数。上面的代码是一个只含有 main 函数的程序。每个 C++程序必须有且只有一个 main 函数。当这个程序执行时，它不产生任何操作。

严格的 C++形式的空操作程序为：

```
#include <iostream.h>
int main() {return 0;}
```

整个上述程序被保存在名为 donowt.cpp 的文件中。

.cpp 部分是必需的，它表明这个文件包含一个 C++程序；“donowt”是由用户自己选择的名字。

iostream.h 是一个标准头文件，它包含了跟在它后面的程序在进行编译和执行时所需要的声明。iostream.h 和 C 语言中的 stdio.h 头文件等同，但不可以代替它，iostream.h 声明了 C++库函数和工具（参看第 6 章）；stdio.h 声明了标准 C 库函数，比如 printf 等。

main 前面的 int(integer)定义当程序执行时向操作系统返回一个数值（本例是 0）。函数小括号中是空的，在 C++中这种情况和 C 中的定义空变量列表是类似的，在这种情况下 main 函数不能接受任何参数。

下面是 C++空操作程序的一种比较复杂的形式，它的名字是 donowt.cpp。它使用了一个很小的 C++类来表示不产生输出：

```
Do nothing
```

// 表明它后面的文字属于注释内容——详细内容请看下节。

```
// donowt.cpp - program using a
      simple C++ class to display nothing
#include <iostream.h>
class nodisp
{
```

private 和 public 的解释请参看第 2 章。

```
private:

  public:
      void output()
          {
          return;
      }
  };
  int main()
  {
      nodisp screen;
      screen. output();
      return(0);
  }
```

这时程序声明了一个C++类nodisp,它唯一的一个成员是函数output。在main函数中,我们定义类的一个实例:

```
nodisp screen;
```

当在主程序中执行程序行screen. output();时,函数nodisp::output()(类nodisp的成员函数输出)便简单地将控制转向它后面的语句。在本例中,因为后面的语句是位于主程序末尾的return(0),donowt. cpp便停止执行,不产生任何输出。

1.3 建立和运行C++程序

C程序的文件名后缀都是.c,C++程序的文件名后缀没有这样统一的标准。在UNIX操作系统下,C++源码文件名可以用.c或者.C结尾。对许多个人计算机系统和工作站来说,其后缀是.cpp。一些基于个人计算机的C++编译器要求.cxx后缀。本书只使用.cpp后缀。

程序donowt. cpp必须首先被编译程序和链接程序转换为可执行程序码。如果在个人计算机上有Borland Turbo C++编译器和链接程序,应该用下面的命令行来编译donowt. ccp:

```
tcc donowt. cpp
```

这样就可以产生一个称为 donowt.exe 的输出文件，它可以在命令行下运行。类似的 Borland(非 Turbo)C++系统的命令行为：

```
bcc donowt.cpp
```

对于一些 Microsoft C++编译器来说，你可以使用“c-ell”命令：

```
cl donowt.cpp
```

如果你在使用 UNIX 操作系统，则可以用下面的命令行编译和加载(“链接”是在 UNIX 上的称谓)程序(注意 C 是大写字母)：

```
CC donowt.C
```

执行程序被放在名为 a.out 的文件中(对汇编器输出而言)。

虽然一些简单程序可以在命令行下建立(编译和链接)，目前更多的是使用由 Microsoft，Borland，IBM 或其他公司提供的 IDE(集成开发环境)。它所具有的菜单驱动界面对管理较大的程序来说更有利。但本书的目的并不是告诉你如何来使用软件供应商提供的这些 IDE 产品。

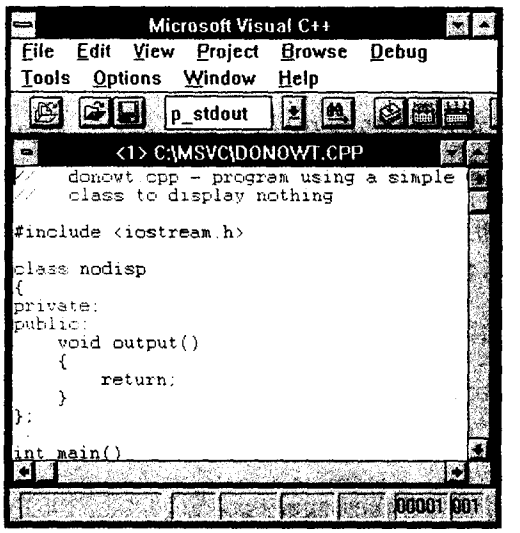
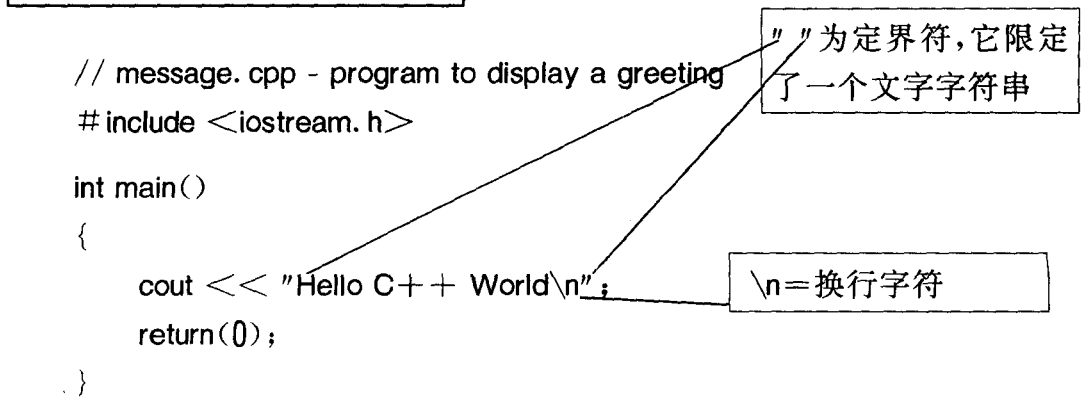


图 1-1　在 Microsoft Windows 3.11 环境下运行的 Visual C++顶层窗口

现在你所掌握的知识已能建立简单的 C++程序了，下面我们继续来写一些有一定实际功能的程序。下面是一个名为 message.cpp 的程序：

Message




```
// message.cpp - program to display a greeting
#include <iostream.h>

int main()
{
    cout << "Hello C++ World\n";
    return(0);
}
```

C++程序中采用了许多C程序中没有被正式采用的双斜杠注释符:在一行中双斜杠//后面的所有字符都被忽略。C程序中的/*……*/在C++中被保留下来,但对较短的注释来讲,经常使用//。

stdio.h被头文件iostream.h所代替。iostream.h中包括了将由预处理器嵌入源程序文件中的类和函数的声明,而这些声明对C++流I/O库的使用是必要的。

其中的一个工具便是cout,它代表标准输出流对象。假如终端屏幕是标准输出设备,操作符<<右面的字符被送到cout,然后cout将这些字符显示在用户终端上。<<操作符实际上是逐位左移操作符,它被C++系统重载,意思是"向流中插入"。C++的流I/O系统将在第6章中讨论。

你可以在自己的计算机上输入并建立这个程序。作为练习,自己建立一个程序message.c,使它能显示两行:

```
Ask not what your country can do for you
Ask rather what you can do for your country
```

下面是一个基于类的message.cpp版本,它和上面显示的较简单的message.cpp可以产生相同的结果:

Message 2

```
// message.cpp - uses a simple C++ class to display a greeting
#include <iostream.h>

class message
```

```
{
private:
public:

    void greeting()
    {
        cout << "Hello C++ World\n";
    }
};
int main()
{
    message user;
    user.greeting();
    return(0);
}
```

<< 被 C++ 系统重载，意思是“向流中插入”

message 后面花括号内的所有一切都是类 message 的成员。message 的所有成员都被声明为 public(公有成员)；它们一般都是可以访问的。message 只有一个公有成员，这就是函数 greeting，它没有返回类型和变量列表。

在主程序 main 中，我们定义了类 message 的一个实例 user。通过调用函数 greeting：

```
user.greeting();
```

“Hello C++ world”便可以被显示出来。

在本例中使用类未免有些大材小用，但从中你能够理解类结构的简单特性。

注释：

关于类的进一步讲述见第 2 章。

1.4　C++对 C 的扩展

本节将总结从 ISO C 到 C++中最重要的语法变化，而不讨论 C++所增加的面向对象的程序设计工具和 I/O 流，模板以及其他库等。

1.4.1 语句

在C++中,声明语句不一定在函数开始部分,而是可以跟在其他语句的后面:

```
#include <iostream.h>

int main()
{
  for (int x = 5; x < 10; x++)
      ;
  cout << "First declaration: " << x << "\n";
  int y=6;
  cout << "Both declarations:" << x << " " << y << "\n";
  return(0);
}
```

可以把变量声明和初始化当成for循环中控制语句的一部分。

1.4.2 类和结构标记

类、结构和枚举类型标记本身就是一个类型。在C语言中,可以这样声明一个结构:

```
struct list
{
  int x;
  double y;
};
```

并可以用下面的方法来声明结构的一个实例inst:

```
struct list inst;
```

在C++中,可以使用同样的方法进行声明,也可以采用简写方式:

```
list inst;
```

1.4.3 关键字

C++引入了许多新关键字,它们包括:

class delete friend inline new operator