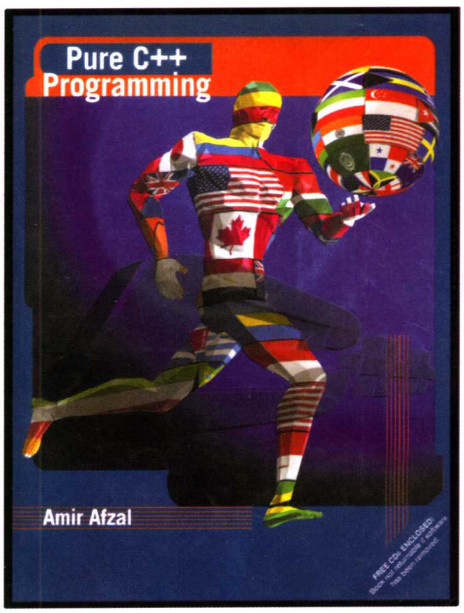
·国外经典教材系列·

Pure C++ Programming

纯粹 C++ 编程教程

北京希望电子出版社　总策划
（美）　Amir Afzal　著
吴　平　译



科学出版社
www.sciencep.com

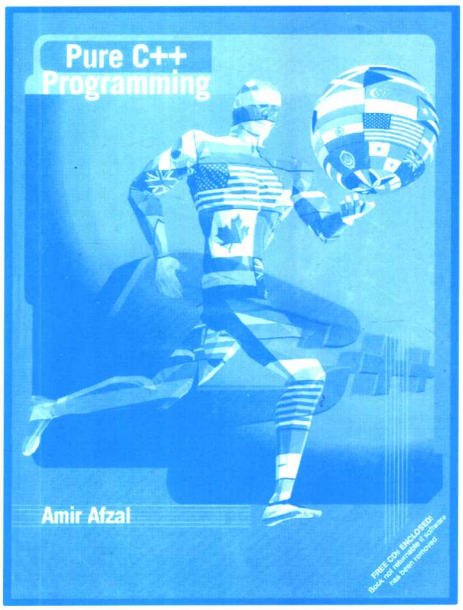
PEARSON
Prentice Hall

·国外经典教材系列·

Pure C++ Programming

纯粹 C++ 编程教程

北京希望电子出版社 总策划
(美) Amir Afzal 著
吴 平 译



科学出版社
www.sciencep.com

PEARSON
Prentice Hall

内 容 简 介

本书是专门为具有C语言编程知识的读者与学生编写的，是论述C++语言概念及语法的通用C++书，它围绕C++语言这个中心对全书进行谋篇布局，从介绍非面向对象的函数和运算符开始，逐步过渡到引出C++面向对象特性，讨论更复杂的概念。读者凭借该书打下坚实的C++知识基础之后，即可触类旁通、举一反三，在C++理解和实践方面满怀信心地更上一层楼，把C++语言知识运用到自己感兴趣的领域。本书特点：每章内容简明扼要，并配有说明、例子、表格和图形；用图标和颜色提高文本可读性；体例保持一致性；每章最后附有复习题和编程练习。本书配套光盘内容为部分程序源代码。

需要本书或技术支持的读者，请与北京中关村083信箱（邮编：100080）发行部联系，电话：010-82702660，82702658，62978181转103或238，传真：010-82702698，E-mail：tbd@bhp.com.cn。

图书在版编目（CIP）数据

纯粹C++编程教程 /（美）阿弗扎尔（Afzal，A.）著；吴平译. —北京：科学出版社，2005.5

国外经典教材

ISBN 7-03-014523-2

Ⅰ. 纯… Ⅱ. ①阿…②吴… Ⅲ. C语言—程序设计—教材

Ⅳ. TP312

中国版本图书馆CIP数据核字（2004）第109318号

责任编辑：谢建勋　/责任校对：佳　宜

责任印刷：双　青　/封面设计：王　煊

科学出版社 出版

北京东黄城根北街16号

邮政编码：100717

http://www.sciencep.com

双青印刷厂 印刷

科学出版社发行　各地新华书店经销

*

2005年5月第　一　版　开本：787×1092 1/16

2005年5月第一次印刷　印张：24 1/4

印数：1—3 000册　字数：561678

定价：38.00元（配光盘）

序

纯粹 C++编程教程

《纯粹 C++编程教程》主要针对具有 C 语言编程知识的读者与学生编写的。本书首先介绍 C++非面向对象的函数和运算符及其与 C 语言的差别，还有在 C 语言基础上的改进。然后介绍 C++面向对象特性，讨论更复杂的概念。本书提供了坚实的 C++基础，使读者可以轻松而充满信心地进一步学习更高级的参考书。

本书的编写有两个目标：

- 使具有一般 C 语言编程知识的编程人员也能读懂本书的内容。
- 能提升读者理解和实践 C++的水平，使他们能更高效、更轻松地利用参考书和编译器帮助功能。

《纯粹 C++编程教程》不是面向商业、科学、任何特定编译器或任何开发环境的，正如书名所述，是纯粹的 C++编程教程。本书强调的是 C++语言而不是程序设计技术或程序开发方法。当然，这些编程方面并非不重要，只是不属于本书要讨论的范围。这是一本通用的 C++书，它论述的是 C++概念和语法，读者可以把这些 C++语言知识运用到他们感兴趣的领域。

纯粹 C++编程教程的特点：

- 各章内容尽量保持简短。内容多的分成两章。
- 段落和小节尽量保持简短，并配有说明、例子、表格和图形。
- 用图标和颜色提高文本的可读性。
- 体例保持一致性。每一章用部分程序解释所阐述的问题，并用完整的程序进一步在程序环境中探索相关的材料。
- 每一章的程序都是完全注释的，但书中没有实际使用的编程例子。实用程序通常既大又复杂，不适合在本书中进行 C++教学。
- 程序例子尽量保持简短，作为对所介绍的课题和正文的补充。
- 每章末尾提供复习题和编程练习。编程练习中的程序可以在上课时间内完成。
- 每章末尾的复习题和编程练习是本书总体教学方法的重要部分。编程练习中引入了另一种复杂性，是对正文中编程例子的补充。
- 正文中的有些程序本可以用更好的设计与算法写得更有效，但我们故意让编程例子保持简洁。本书中各章的例子尽量少用书中没有介绍的或出现在后面章节中的项目，便于读者阅读程序清单和了解关键字与函数的来源。
- 为了保证程序平台无关性，我们不用终端属性函数之类的编译器特定函数。
- 第 15 章是个完整的菜单驱动程序，采用了编写 C++程序的许多标准方法，可以演示生成大型多文件程序的过程。针对不同的学生，这个驱动程序可以作为课堂上的长期项目，逐步开发，也可以作为改进某个项目的起点。

资　源

若想得到更多的资源、纠正和更新信息，请访问作者 Web 站点：

http://www.afzalbooks.com

为了保证文本的正确性，我们已尽了最大的力量，但错误仍在所难免。欢迎读者批评指正，包括指出例子程序中的错误、光盘中的源代码文件、插图和输入错误。希望今后再版或更新光盘时能纠正这些错误。有关意见和建议请发送到下列网址：

afzalbooks@ afzalbooks.com

致　谢

在本书成书过程中得到了学术界和业界众多同事的帮忙和支持，在此谨致衷心的感谢。以下所列，挂一漏万。

- 感谢我在 C/C++与 Unix 课堂上的学生所提供的反馈和测试例子程序。他们是促成写本书的主要原因。
- 感谢摩托罗拉公司的同事们在时间安排上支持我的编写工作。
- 感谢 Unix 管理一书的合作者 Tom Swanson 提供的建议、意见和技术编辑工作。
- 感谢 Terry Ralph，他作为我的技术编辑，编写并测试了许多程序以及帮我准备教师手册。
- 感谢 Prentice Hall 公司人员，特别是 Charles Stewart 长期的支持与鼓励。他的合作对创作本书和其他“纯粹”系列教程是至关重要的。
- 感谢 David Afzal 维护 http://www.afzalbooks.com 网站。

还要感谢下列审校人员的建议、改正意见和睿智的批评：Eltayeb S.Abuelyaman（亚里桑那州立大学），Iem Heng（DeVry 理工学院），Leei Mao（Greenville 技术学院），Kris Petersen（新墨西哥州立大学），Randal Reid（Chattahoochee 技术学院）和 Jerome Zornesky（技术职业学院）。他们的辛勤劳动为本书增色不少，使之成为一部名副其实的优秀教科书。

Amir Afzal

译 者 序

本书翻译过程中得到了刘文红、周阳生、邹能东、彭振庆、黄志坚、李耀平、郭王旋等同志的大力帮助，刘文琼、黄素平、李富招等同志完成了本书的录入工作，刘云昌、刘昌和帮助进行了译稿与打印稿的校对，在此深表感谢。

吴　平

目 录

第1章 引 言

主要内容

- C++编程语言
- 第一个C++程序
- C++程序部件
- 风格问题
- C++保留字
- C++非面向对象特性
- 全局范围解析运算符

1.1 C++编程语言

程序一旦超过两万行，就会变得很复杂，很难进行总体把握。为了管理大程序和帮助编程人员理解越来越复杂的程序，C++语言应运而生。

C++是80年代由AT&T贝尔实验室的Stroustrup开发的，作为C语言的扩展与改进。第一版称为“带类的C语言”，然后不断改进，几年后成为C++语言。

C语言是C++的基础，C++包含了整个C语言。C++不是生成全新的编程语言，而是改进已经成功的编程语言。C++能维护C语言原有的特征，如效率与灵活性。和C语言中的情况一样，编程人员可以控制语言，而不是被语言主导。

C++支持面向对象编程。尽管C++最初是为了帮助管理超大型程序，但它适用于所有类型的编程环境。C++可以用来开发编辑器、数据库、通信程序和许多高性能系统。

1.2 第一个C++程序

程序清单1.1是第一个C++程序的源代码，它是个交互式程序，可显示5个笑脸符号。这个程序很简单，但展示了C++语言的几个特性，是编程例子的良好开端。

要显示这个程序的输出，就要编译和运行程序。附录B介绍了程序开发的概念与机制：生成源代码、编译、链接和运行程序。

C++编译器可以编译用C或C++编写的程序。大多数编译器认识扩展名为.cpp的源代码文件，知道要进行C++编译。扩展名.c表示C语言编译。注意检查编译器文档，并不是每个编译器都需要或认识扩展名.cpp。

程序清单1.1 源代码

```
//
//      文件名：Prog0101.cpp
//      描述：
//      是个交互式程序，提示用户输入Yes或No，在用户输入Yes时显示5个笑脸符。
// // // // // // // // // // // // // // // // // // // // // // // // //
```

```
// 1- 预处理器指令
#include  <iostream.h>              // 包括 I/O 头文件

// 2- 声明函数
void displayFaces(void);           // 函数原型

int  main()
{
// 3- 声明变量
  char  answer;                    //声明变量

// 4- 用输出运算符显示提示消息
  cout  << "Do you want to see smiley faces?"  << endl;
  cout  << "[Y]es or [N]o: ";

// 5- 用输入运算符读取键盘
  cin  >> answer;

// 6- 检查用户输入并采取相应的操作
  if (answer == 'Y' || answer == 'y')          // 检查 yes 答案
    displayFaces();                            // 函数调用
  else if (answer == 'N' || answer =='n')      //检查 no 答案
    cout  << "OK! No smiley faces."  << endl;  // 显示 OK!
  else                                         // 不良输入
    cout  << "Wrong input!"  << endl;          // 显示错误消息

// 7- 结束程序消息
  cout  << "End of my act! :-)"  << endl;
return 0;
}                                              // 结束程序

// // // // // // // // // // // // // // // // // // // // // // // // // // //
//  displayFaces
//  这个函数显示 5 个笑脸符
// // // // // // // // // // // // // // // // // // // // // // // // // // //
void                                           //返回类型
displayFaces(void)                             // 函数头
{
  for (int index = 0; index < 5; ++ index)     // 循环 5 次
    cout  << ":-) ";                           // 显示一个笑脸符
```

```
  cout  << endl;                                    // 新行
return;
}
```

程序输出

```
Do you want to see smiley faces?
[Y]es or [N]o:Y
:-) :-) :-) :-) :-)
End of my act! :-)
```

1.3 C++程序部件

下面解释第一个C++程序。解释不求详尽，只是简要地介绍C语言与C++的一些差别。不要为面向对象术语发愁，后面各章会介绍所有这些术语。别着急，好戏在后面。

1.3.1 注释行

注释行是程序内部文档，可以提高可读性。行中任何位置的//号（两个斜杠）表示注释行开头，此后输入的任何内容均被编译器忽略。下列行是第一个C++程序中的注释行：

```
// // // // // // // // // // // // // // // // // // // // // // // // // //
// 1- 预处理器指令
```

我们仍然可以用/*与*/构成的C语言符号表示注释行开始和结束。典型C++程序可能混用这两种注释行。C语言注释通常用于多行注释，而C++式的双斜杠（//）则常用于单行注释和半行注释。

说明

- 两个斜杠（//）之间没有空格。
- C++注释不像C语言式注释一样需要终止符。

1.3.2 包括库文件

下列代码行在程序中包括指定的库文件：

```
// 1- 预处理器指令
#include  <iostream.h>          //包括 I/O 头文件
```

iostream.h是头文件名，包含输入/输出函数原型和必要的定义，其作用与C语言程序中的stdio.h相同，可用来访问预定义的C++输入/输出函数。

说明

- 要使用printf()与scanf()之类C语言输入/输出函数，就要在程序中包括stdio.h头文件。

1.3.3 输入/输出：C++样式

下面两条语句显示提示消息：

```
// 4- 用输出运算符显示提示消息<<
  cout  << "Do you want to see smiley faces?"  << endl;
  cout  << "[Y]es or [N]o: ";
```

cout 加上输出运算符，<<可以将消息输出到显示器中。cout 对象表示输出设备，输出运算符<<对 cout 对象进行操作。得到的结果是在屏幕上显示<<运算符右边的消息。注意，cout 是足够聪明的，知道引号中的消息是字符串，不必像 C 语言中用 printf()显示字符串那样需要指定。

我们还用输出操纵符 endl 产生回车符和换行符序列，将光标放到下一行，相当于 C 语言中的换行符（\n;）。

下面要读取用户输入。

```
// 5- 用输入运算符读取键盘 >>
  cin  >> answer;                                    //读取输入字符
```

我们用 cin 和输入运算符>>从键盘读取用户输入。cin 对象表示输入设备，对 cin 对象进行操作。得到的结果是把用户输入存放在输入运算符>>右边的变量中。注意，cin 是足够聪明的，知道 answer 是个 char 类型变量，不必像 C 语言中用 scanf()进行输入操作时一样需要指定变量类型。

图 1.1 描述了 cin 与 cout 对象使用>>与<<运算符时的输入/输出操作。

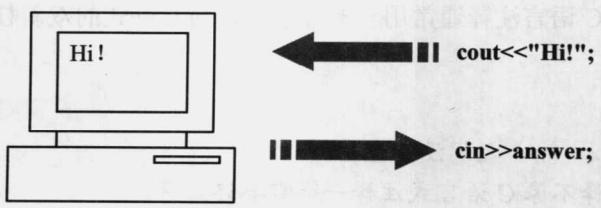


图 1.1 使用 cin 与 cout 的输入/输出操作

然后，程序用 if-else 结构检查输入数据。如果 if 条件为真，即用户输入 Y 或 y 表示 yes，则调用 displayFaces()函数。

```
// 6- 检查用户输入并采取相应操作
  if (answer == 'Y' || answer == 'y')                // 检查 yes 答案
    displayFaces();                                  // 函数调用
```

这个函数显示程序输出所示的 5 个笑脸符号。否则，如果用户输入 N 或 n，表示 no，则执行 else if 内容，即 cout 语句：

```
else if (answer == 'N' || answer =='n')              //检查 no 答案
  cout  << "Ok! No smiley faces."  << endl;          // 显示 Ok!
```

并显示下列消息：

```
Ok! No smiley faces.
```

如果答案既不是 yes 也不是 no，则进入 else 段。

```
else                                                    // 不良输入
  cout  << "Wrong input!"  << endl;                      // 显示错误消息
```

cout 语句在屏幕上显示下列错误消息：

```
Wrong input!
```

无论如何，总是会执行程序中最后一行并显示程序结束消息：

```
End of my act! : -)
```

1.3.4　用户定义函数 dsiplayFaces()

dsiplayFaces()函数的原型语句如下：

```
// 2- 声明函数
void displayFaces(void);                                // 函数原型
```

这个用户定义函数用一个 for 循环迭代 5 次，每次显示一个笑脸符号：

```
  for (int index = 0; index < 5; ++ index)              // 循环 5 次
    cout  << ":-) ";                                    // 显示一个笑脸符号
```

注意：for 循环初始化部分的 int index = 0，这是合法的。C++可以在需要时随时声明变量，只要先声明后使用就行。下列 cout 语句产生新行，将光标放在下一行就行。

```
  cout  << endl;                                        // 新行
```

说明

- 可以在同一程序中使用 C 语言和 C++提供的输入/输出函数，只要包括相应的库头文件。

1.4　风格问题

和 C 语言一样，C++也是个自由格式语言，即可以在输入程序时使用任意空格符。空格可以放在一行的项目之间或跨行的项目之间，增加程序的可读性。例如，可以声明相同类型的变量表如下：

```
int
x,y,z;                  //变量名
```

将变量类型 int 放在一行，将变量表放在另一行。dsiplayFaces()头行使用了这种风格：

```
// // // // // // // // // // // // // // // // // // // // // // // // //
//  displayFaces
```

```
//  这个函数显示 5 个笑脸符号
// // // // // // // // // // // // // // // // // // // // // // // //
void                                                    //返回类型
displayFaces(void)                                      // 函数头
{
  body of the function
}
```

注意：函数头行包括两行，返回类型（这里是 void）放在一行，函数名放在一行。

说明

- 编译器忽略空白符。
- 编写程序时一定要保持风格一致性。通常，在结构化编程技术中，我们要按预定义标准风格代码。这在小组工作开发环境中特别重要。
- 目前先用程序例子中提供的编程风格。

1.5 C++保留字

C++语言有 48 个保留字和许多预定义函数。这些预定义函数在几个标准库中提供，通常在编译器软件中提供，使用时在程序中包括相关头文件。34 个保留字是 C 语言和 C++共有的，另外 14 个保留字是 C++独有的，只能在使用 C++编译器时使用。

表 1.1 列出了 C++语言的 48 个保留字，为了突出 C++独有的保留字，表中用黑体字显示。

表 1.1 C/C++语言保留字

C/C++语言保留字				
asm	auto	break	Case	**catch**
char	**class**	const	Continue	default
delete	Do	double	Else	enum
extern	**friend**	float	For	goto
if	**inline**	int	Long	new
operator	**private**	**protected**	**Public**	register
return	short	signed	Sizeof	static
struct	switch	**template**	**This**	**throw**
try	typdef	union	Unsigned	**virtual**
void	volatile	while		

说明

- C++语言是区分大小写的，C/C++语言保留字应用小写字母。

1.6 C++非面向对象特性

C++是对C编程语言的改进，其中包括单纯的C语言改进和面向对象编程（OOP）特性。本章余下部分介绍C++语言中常用的非面向对象编程（non-OOP）特性，是新增的或与C语言不同的。

1.6.1 注释行：//

双斜杠（//）表示注释行，分隔单行注释。编译器忽略//号后面的一切。如：

```
//这是个C++式注释行
int x = 10;                                    //声明和初始化x
```

注释行是程序内部文档，可以提高可读性。但太多注释行与源代码交织在一起也会影响代码清晰性。

常见错误

➢ 在两个斜杠（//）之间放上空格是错误的。注释分隔符的两个字符应该连接。

1.6.2 变量声明

C语言中所有变量声明都要放在程序或代码块开头。C++中可以在任何地方进行变量声明，只要先声明后使用。这样就可以在靠近需要变量的点声明变量。例如，下列代码中在for循环初始化部分声明count，放在首次使用之前：

```
//声明和初始化count
  for (int count = 0;count<5;++ count)          //循环5次
     count<<count;                              //显示count值
```

但是，不能在while与if之类语句的条件部分声明变量。例如，下列语句中的变量声明都是非法的：

```
while (int n < 10)                             //非法使用新变量声明
```

或

```
if(int n = 10 < 10)                            //非法使用新变量声明
```

说明

➢ 这个特性要慎用，因为新定义的变量很难看到，会掩盖代码的真实意图。

1.6.3 函数原型

函数原型指声明函数和编写函数头的样式，在括号中包括数据类型，C语言中的函数原型是可选的，而C++语言的函数原型是强制的。每个函数都要有原型，就是参数表中列出数据类型和列出返回类型的函数声明。例如，下列语句在C++中是等价的：

```
int myFunction(void);                          //函数原型语句
```

与

```
int myFunction();                              //另一种形式的函数原型语句
```

都声明函数 myFunction，返回一个 int 值，不需要变元。下列函数声明：

```
float myFunction(int num,char ch);          //指定变量名
```

声明函数 myFunction，返回 float 类型值，参数表中包括一个 int 值和一个 char 值。函数声明中的变量名（这里是 num 和 ch）是可选的，但为了明确起见，通常要加上变量名。因此，上述函数声明可以写作：

```
float myFunction(int,char);                 //不指定变量名
```

说明

- 函数原型以分号结尾。
- 函数声明语句（原型）不分配任何存储空间。

C++中函数原型或定义的空参数表解释为不指定参数，即函数不取任何变元。例如，下列语句：

```
int myFunction();                           //C++形式的函数原型语句
```

表示函数 myFunction 没有参数，返回一个 int。这与 C 语言中的含义大不相同。在 ANSI C 中，函数定义的空参数表在调用 myFunction 时停止参数表的类型检查。例如，下列声明：

```
int myFunction();                           /*C 形式的函数原型语句*/
```

表示函数 myFunction 可以取 0 个或多个任何数据类型的变元。下列函数调用语句是正确的函数调用语句：

```
myFunction(10,1.5, 'A' );                   /*3 个变元的函数调用*/
myFunction("hello",10);                     /*2 个变元的函数调用*/
myFunction();                               /没有变元的函数调用*/
```

说明

- 本书假设读者具有 C 语言编程经验，熟悉函数原型。更多信息见《纯粹 C 语言编程教程》或其他 C 语言书籍。

1.6.4 数据类型转换

C++提供了新的数据类型转换方法和语法。C 语言中数据类型转换运算符的语法如下：

(*typecast*)variable

typecast 对变量值采用的新数据类型

variable 要临时变为新数据类型的变量

下面代码块显示了 C 语言中数据类型转换的例子：

```
/*C 语言中数据类型转换的例子*/
int_value = int (2.6) *int (100.50)              /*使用 int 类型*/
xptr = (struct x*) malloc(sizeof(struct x));     /*使用 structure 类型*/
```

C++中的数据类型转换语法相当于函数调用。转换运算符不带括号，变量放在括号中。