附範例磁片

北京希望电脑公司C++技术丛书

Borland C++/Turbo C++

编程实例剖析

謝明興　編著

科学出版社

北京希望电脑公司 C++技术丛书

Borland C++／Turbo C++编程实例剖析

谢明与 著

文 都 鞠玉兰 改编

李晓花 葛 明 审校

科学出版社

1993

(京) 新登字 092 号

内容提要

随着 Borland C++和 Turbo C++编程语言的流行，世面上已有不少介绍这两种开发环境和编程语言的书籍，但提供大量编程实例的书籍却很少．本书详细描述了 Borland C++和 Turbo C++语言文本，同时给出了近 300 个短小、精巧的程序实例，并对它们进行了说明．本书中的程序涉及各种 OOP 技巧，而且经过上机调试，可供读者借鉴或直接用于自己的程序中．

本书中的所有程序均可在 Turbo C++和 Borland C++环境下运行．

本书适合于大中专院校师生和计算机程序员使用．

需要本书的读者可与北京希望电脑公司资料部联系．邮政编码：100080．电话：2562329．通信地址：北京市 8721 信箱．

版 权 声 明

Borland C++／Turbo C++编程实例剖析

谢明与 著

文 都 鞠玉兰 改编

李晓花 葛 明 审

责任编辑 张建荣

科学出版社出版

北京东黄城根北街 16 号

邮政编码：100717

双青印刷厂印刷

新华书店北京发行所发行 各地新华书店经售

*

1993 年 10 月第 一 版 开本：787×1092 1／16

1993 年 10 月第一次印刷 印张：23 3／4

印数：1-3000 字数：550 000

ISBN 7-03-004059-7／TP・347

定价：49.00 元(含盘)

目　录

第一章　简介

1.1　C与C++

1972年，美国AT&T贝尔实验室的Dennis Ritchie和Ken Thompson在共同设计UNIX操作系统时开发出C语言。

由于C语言程序容易维护，功能极强，所以已成为计算机界开发系统软件或数据库软件的重要语言。但当系统较庞大，程序代码大于数千行时，变量、函数的维护就成了一个巨大的负担，而具备代码可重用性、可扩充性及对数据或模块能进行封装保护的"面向对象程序设计(Object Oriented Programming，OOP)"的概念将受到广泛注目。

1980年，贝尔实验室的Bjarne Stroustrup开发出C++语言，其中包含大部分C语言及无数C语言程序库，并具备面向对象程序设计的方法。

OOP提供新的程序开发概念，包括数据隐藏、封装、操作符和函数的重载等。1988年，美国Borland公司集中美国各地编译程序专家的智慧，设计出能与ANSI C及ANSI C++ 2.0兼容的Turbo C++程序设计语言。

1990年，美国Microsoft软件公司推出第三代窗口操作系统Window 3.0，该系统提供了用户和计算机间良好的用户界面。Borland公司于1991年开发出与Turbo C++类似的编译环境Borland C++ 3.0，它除了具备Turbo C++的功能外，还为专业软件开发人员提供增强的软件开发工具，它是有助于开发Window操作系统的程序环境，这些开发工具包括汇编器(assembler)、增强的调试器(debugger)及其它实用程序。

1.2　Turbo C++与Borland C++

Turbo C++和Borland C++包含Turbo C 2.0的所有功能，用Turbo C 2.0开发的程序能在Turbo C++中编译和运行。Turbo C++和Borland C++另行扩充增强的部分包括:

1. 强化的语法

   包括函数引用调用(call by reference)、引用参数(reference argument)、函数声明时缺省参数(default argument)、简单易用的I/O语句、增强的内存分配命令NEW等。

2. 面向对象的程序设计(OOP)技术

   包括类定义(class)、友元函数(friend function)、构造函数(constructor)、析构函数(destructor)、虚拟函数(virtual function)、虚拟类(virtual class)、继承(inheritance)、操作符重载(operator overloading)、函数重载(function overloading)、抽象数据类型(abstract data type)、面向对象的文件和I/O流(I/O stream)，以及各种类库(class library)。

Borland C++还包含:

1. 调试器(debugger)：允许程序设计者跟踪程序设计的过程和各类对象(class object)的内容等.
2. 剖析器(profiler)：剖析程序设计的效率.
3. Windows 3.0 Toolkit(工具箱)，充分支持用户界面的强化.

Turbo C++和 Borland C++可以在 PC／XT 、PC／AT 、PS／2 系列、386、486 及其它兼容机上运行. Turbo C++至少必须在 MS-DOS 2.0 以上，Borland C++则至少必须在 MS-DOS 3.0 以上的环境中运行.

Turbo C++和 Borland C++至少必须在内存为 640KB 的环境中运行（硬盘约 6MB 字节）. 它们使用 Small 内存模式，在 A 磁盘上存放 TC.EXE，在 B 磁盘上存放头文件(head file)及程序库(library). 另外，在至少 1MB 的 RAM 中开设一虚盘，以存放程序和可执行文件. Turbo C++亦可在无硬盘，但带两个 1.2MB 软驱及 1MB RAM 的环境中运行.

1.3 面向对象的方法

用面向对象(Object Oriented)的方法解决问题的基本概念是将数据以及处理这些数据的函数结合在一个单元中，这样的单元称为对象(Object). 规范化的对象中包含变量和函数的那些数据类型称为类(class)，因此对象可以看成类的成员(member of class)，也称为类变量.

对象中的每个函数都称为成员函数(member functions)，它们仅用来存取对象中的成员变量(member variables). 程序设计者不能直接存取对象中的数据，而必须通过成员函数来存取，以避免无意中改变变量的值. 这种保护对象数据的方法称为数据隐藏(Data Hiding). 数据和函数结合在一个单元(对象)中称为封装(encapsulated).

Turbo C++和 Borland C++中的所谓成员函数在其它面向对象语言(如 Smalltalk)中称为方法（methods），对象中的成员变量或数据项在 Smalltalk 中称为实例变量（instance variable），调用对象的成员函数称为发送消息给对象(sending message).

以面向对象的方法来解决问题，不再将问题划分为解决它的函数，而是将问题细分为对象. 在真实世界中，这些对象可能是用户定义的数值数据类型(如复数)、数据库(如人事文件)、程序结构(如二元树)，以及如计算机游戏中的角色等.

第二章　C++中的基本I/O语句

2.1　基本程序结构

在讨论C++程序结构之前，先看看以下程序，它能用来说明简单、具体的C++程序结构。

【范例】intro.cpp

换行后，分别打印出两行信息。

```
// intro.cpp: A First Glance

#include <iostream.h>      // for cout

main()
{
    cout << endl;
    cout << "I like Torbo C++/Borland C++...\n";
    cout << "C++ is more than a superset of c.\n";
}
```

【输出】

```
I like Turbo C++/Borland C++...
C++ is more than a superset of C.
```

【说明】

1. 函数是C++最基本的组成部分。C/C++程序在运行时均把一个名称为main()的特殊函数作为程序的第一条语句开始运行。程序是由程序语句、函数或对象组成的，它可以完成特定的工作。在链接(linking)阶段，当链接程序找不到main()函数时会产生错误信息。

2. 函数名(如main)后面必须跟着小括号，避免程序在编译阶段把该函数名当成变量或其它程序语句。小括号内可以包含一个参数序列，可以接受传给函数的值(value)或地址等。我们将在有关函数的章节中进一步讨论这些问题。

3. 函数中由一对大括号"{}"括住的部分称为函数体(function body)。函数体包含程序语句、函数和其它控制结构。范例intro.cpp中只有三条简单的语句，每条语句需用分号";"结束。

4. #include <iostream.h>

预处理指令(preprocessor directive)#include告诉编译器在原始程序代码中插入(insert)其它程序，编译时实际上是以头文件(head file，也称包含文件)的内容取代#include预处理指令。

5. <iostream.h>

C++为使程序本身更简洁，移植性更强，只定义语言本身的基本结构，I/O及与系

统有关的部分则交给类库(class library)处理．类库包含抽象的数据类型(类，如代表输出串流的类 ostream)以及处理各种类对象的函数．

有关类、对象以及类库请参看各有关章节．

若没有文件 iostream.h，则编译器不能识别标识符 cout 及插入操作符(insertion operator 或 put to operator)．

6. cout 与＜＜操作符

标识符 cout(发音为 C out)实际上是一个属于 ostream 的对象，它是预先定义在 iostream.h 中的标准输出串流．利用操作符＜＜把数据输出到标准输出流上的格式是：

```
cout << output_object
```

其中 output 可以是字符串形式的数据或表达式以及其它控制字符，例如：

```
cout << "Hello, C++ New World !";
cout << 3+7;
cout << "\n";
```

7. 字符串常数

C++并未提供实际的字符串数据类型，但可以把由双引号(")括住的字符作为字符串常量，例如上例中的"Hello"．

8. cout 使用的下列控制字符及操作符均用来表示在输出流中换行：

```
cout << "\n";
cout << endl;
```

其它控制字符及操作符将在以后详述．

9. 注释

注释是程序的重要部分之一，它用来帮助用户了解、阅读及编写程序．C++使用以下两种注释方式：

(1) /＊ ＊/　　适用于单行或多行注释说明

(2) //　　适用于单行注释说明

以下几种注释均是合法的：

```
/* Old-style comment */
// fit in C++
/* This is a
   potentially
   multi line comment */
```

"/＊ ＊/"这种注释法在 C++中使用得较少．习惯上对于多行注释说明可采用这种形式的注释，而对于单行注释可采用"//"．

编译器遇到注释时会略过它，因此不会增加程序代码，程序在运行时也不会额外花费时间．

10. 空白符

空白符包括空格、换行符、回车符和制表符．编译程序会忽略这些空白符，因此编写程序时可以在同一行上安排多条程序语句，或者一条语句占多行．以下程序与运行 intro.cpp 时的输出完全一样．

```
// whitespa.cpp
#include <iostream.h>      // for cout

main ( ) {
      cout << endl; cout <<
      "I like Turbo C++ / Borland C++...\n";
      cout << C++ is a more than a superset of C.\n";
}
```

在编译、链接之后，在集成环境中按 Alt-F5 即可观察运行结果.

2.2 基本变量类型与变量的声明

变量是一个符号名称，在任何程序语言中均扮演重要的角色．变量可以被赋予各种不同的值，变量以及变量值均占用内存空间.

有关在 MS-DOS 中 C++的基本变量类型，可参看下表.

C++中的基本变量类型

变量类型	关键字	数值范围	小数位数	占用的字节数
字符	char	−128 到 127		1
整数	int	−32768 到 32767		2
长整数	long	−2147483648 到 2147483647		4
浮点数	float	$3.4E10^{-38}$ 到 $3.4E10^{38}$	7	4
双精度浮点数	double	$1.7E10^{-308}$ 到 $1.7E10^{308}$	15	8
长双精度浮点数	long double	$3.4E10^{-4932}$ 到 $3.4E10^{4932}$	19	10

【说明】

1. C++含有三种基本整数类型 char、int 和 long，三种浮点数类型 float、double 和 long double．另外，在 MS-DOS 下还另有一个较少使用的 short 整型类型，它占用的内存和表示的数值范围均与 int 类型相同，但在某些计算机系统中 short 比 int 占用的存储空间要小.

2. 长整型及双精度浮点型变量都比整型变量及浮点数能容纳的数值范围要大，但在进行数学运算时速度较慢，占用内存空间较大.

3. 长精度浮点数(long double)在占用 10 个字节(80 位)，与协处理器(8087、80287、80387)处理的数值数据兼容，因而可作为与协处理器有关的变量声明.

有关整型变量的声明，请参看以下程序范例:

【范例】intvar.cpp

```
// intvar.cpp
#include <iostream.h>      // for cout

main()
{
   int v1 = 198;   // define and Initialize
   int v2;
```

```
    v2 = v1 * 20;
    cout << endl;
    cout << v1 << " * 20 = " << v2 << endl;
}
```

【输出】

```
198 * 20 = 3960
```

【说明】

1. 声明整型变量也就是为那些被声明的变量分配内存空间．声明 v1 为整型变量的方法为：

```
int v1;
```

2. 变量声明可以放在同一行中，例如：

```
int v1; int v2;
int v1, v2;
```

3. 对变量名的规定如下：

(1) 变量名可以使用大小写字母、数字或下划线(_)，但变量的第一个字符必须是字母或下划线．

(2) 变量名长度不受限制，但编译程序仅能识别最前面的32个字符．

(3) 编译程序能分辨大小写字母，因此以下变量名代表不同的变量：

```
int Var1, var1, VAR1;
```

(4) 变量名不可使用保留字，如 class、int、while 等．

(5) 习惯上变量名使用小写字母．

4. 变量可以在声明时就赋初值，赋值操作符为"="，表示将"="右边的值或变量赋给"="左边的变量，例如：

```
int v1=120;
int j1=2; j4=8, j6=20;
```

5. 对基本算术操作符的说明如下：

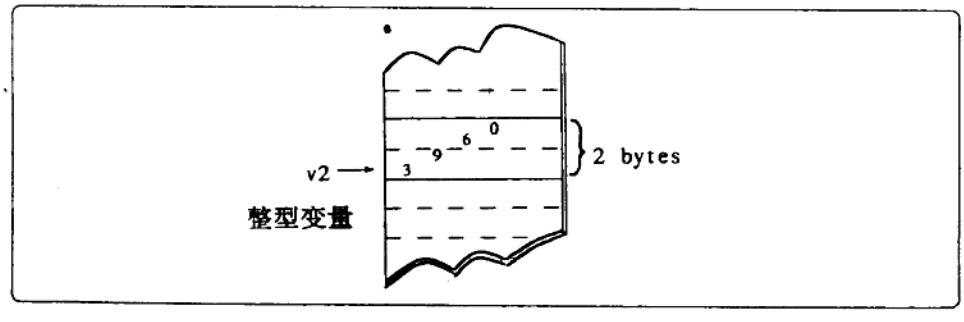
+、－、*、／为 C++中的基本四则运算符号，分别代表加、减、乘、除．

下述语句表示将变量 v1 的值乘以 20 后赋给 v2：

```
v2 = v1 * 20;
```

其中，分号表示语句结束．"v1 * 20"先被求值，再将求值结果赋给 v2．

6. 整型变量在内存中占用两个字节，请参看下图．



2.3 字符变量与转义控制字符

char 型字符变量存储的整数值范围为−128 到 127，每个 char 类型的变量值在内存中占 1 个字节．char 型变量经常用来存储 ASCII 码字符或−128 到 127 间的整数值．

字符常量是用单引号括住的那部分字符，如'A'、'8'、'$'．C++编译器遇到字符常量时会将这些 ASCII 码转换为相对的整数值存储在内存中．

另外，字符常量中紧跟在反斜线"\"后的字符称为转义控制字符，用来控制显示或打印的方式．常用的转义控制字符可参考下表．

转义字符	功 能
\a	发声
\b	退格
\f	换页
\n	换行
\t	移到下一个制表位置
\\	显示"\"
\"	显示双引号
\'	显示单引号
\xdd	十六进制表示

【范例】charvar.cpp

发出"嘟"声后显示 W，隔一个字节(缺省为 8 位)后显示 c．

```
// charvar.cpp
#include <iostream.h>     // for cout
main()
{
    char c1 = 'W'  // define and Initialize
    char t1 = '\a';
    char t2 = '\t';
    char c2 = 'c'
    cout << c1 << t1;
    cout << t2 << c2 << endl;
}
```

【输出】

```
W   'c'
```

【说明】

1. 字符变量不可使用双引号．下面的字符变量声明有错误:

```
char c1 = "W";      <- 错误声明
```

2. 变量声明以及赋值可写在不同的行中，例如:

```
char c1 = 'W'
```

与下列语句的效果相同:

```
char c1;
c1 = 'W';
```

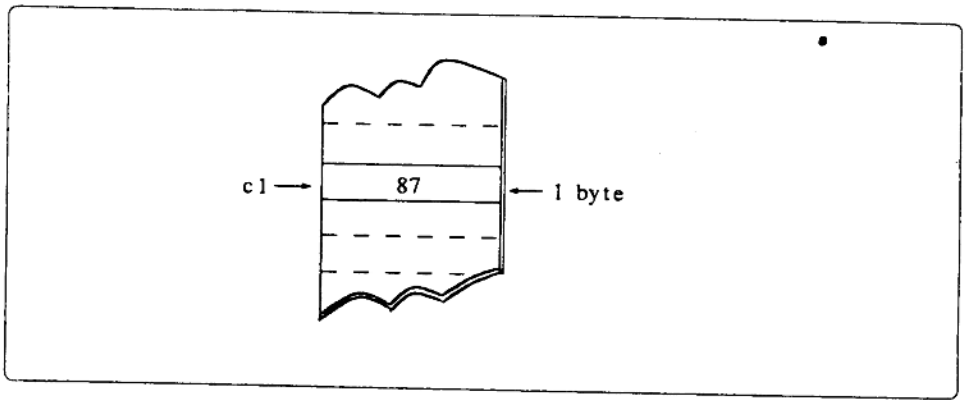
3. 当多次给变量赋值时，该变量的值将是最后赋予的那个值，如：

```
char c1 = 'W';
c1 = 'M';
```

则 c1 变量的值为'M'。

4. '\t'为水平制表符，缺省值为 8 位。要修改它，可使用集成式环境中的"Options – Environment– Editor 选项。

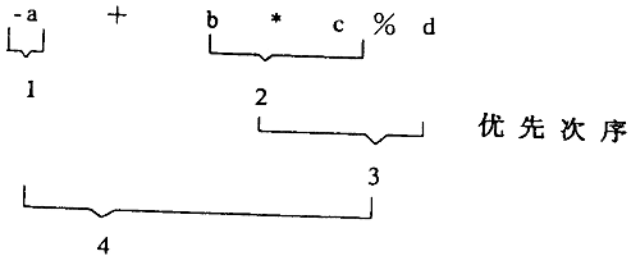
5. 字符'W'在内存中占 1 个字节，请参考下图。



2.4 基本算术运算

C++基本的操作符包括加(+)、减(–)、乘(*)、除(/)及除数(%)，这些操作符都需要两个表达式作为操作数，此外还有一个仅需要一个操作数的符号，这些符号有不同的优先级，请参考下表。

操作符	优先级
() 括号	1(高)
– 负号	2
*，/，%	3
+，–	4(低)



不同的操作符有不同的执行优先级，运算时先处理优先级高的符号，优先级相同时由左向右运算，请参考上述示意图。

【范例】mathproi.cpp

处理不同优先级的操作符。

```
// mathproi.cpp
#include <iostream.h>     // for cout

main()
{
    // Define and Initialize
    int a = 8,  b = 12, c = 2, d = 7;
    cout << endl;
    cout << "-a + b * c % d = " << -a+b*c%d << endl;
}
```

【输出】

```
-a + b * c % d = -5
```

【说明】

在"b * c % d"中，先计算 b * c 的值，然后再除以 d。

【范例】ballvol.cpp

使用浮点数计算球的体积($4\pi r^3 / 3$)。

```
// ballvol.cpp
#include <iostream.h>      // for cout

main()
{
    // Define and Initialize
    float rad = 12; float PI = 3.1416;
    cout << endl;
    cout << "ball volume :" << 4 / 3 * PI * rad * rad * rad;
    cout << endl;
}
```

【输出】

```
ball volume: 5428.6857
```

【说明】

两条以上的语句可以写在同一行，例如：

```
float rad = 12; float PI = 3.1416;
```

2.5 输入数据流 cin 与输出数据流 cout

C++ I/O 流类库的头文件 iostream.h 设置了 C++ I/O 的相关环境，包括 ostream 类及 istream 类。属于 ostream 类的对象（如 cout）配合插入操作符 << 可将数据输出到标准输出流，如屏幕或打印机上。属于 istream 类的对象（如 cin）配合提取操作符(ex-