Turbo C++

自学参考指南

TURBO C++

自学参考指南

张玉亭　韩兰　木林　编译

北京科海培训中心

前　言

由Borland公司推出的Turbo C++是一个完整的C++编程环境。它提供了交互式程序调试器、鼠标控制的窗口环境和高级求助系统等新颖、完美的调试环境和工具，是目前国际上最受推崇的C++编译、调试系统，在国内也越来越受到广大专家、技术人员甚至普通开发者的重视。

由于C++毕竟属于面向对象的程序设计语言，与基本的C语言相比已经有了很大的发展和不同，所以有必要向读者推出一本实用性、可读性强，同时便于自学的这方面的书。本书《Turbo C++自学指南》突出了以下几个特点：

- 边学边实践，直观易懂、印象深刻
- 关键的地方总有“自我检查题”，以便读者掌握好自己的进度
- 每章均以一组编程练习结束，读者可测试自己对书中内容的掌握程度
- 学习实用的Turbo C++编程方法的同时，帮助读者领会面向对象的程序设计的完整概念

本书对于那些已熟悉C语言，并深为这些一般的高级语言繁重、重复而单调的工作所累的程序员而言，无疑提供了一个从C编程风格过渡到C++编程风格，以至于建立起完整的面向对象的程度设计思想的很简易的参考书。书中提供的一个个短小而完整的示例，一次说明一个概念，有助于读者搞清每一个问题、每一个概念。

书中程序需要Turbo C++编译程序，编译后可运行于IBM PC、XT、AT、PS/2或兼容机。所有示例均可在文本方式下运行，因此对显示卡无特殊要求。

本书编译过程中得到了北京科海培训中心华根娣主任和夏非彼编辑的支持。清华大学软件中心沈官林同志审校。在此一并致谢。

编译者

1991、10、北京

目　录

第一章 Turbo C++介绍

Turbo C++的问世使得用C++编程更为容易。这一章主要介绍什么是Turbo C++，以及它与C语言有什么关联。此外，还粗略讲一下Turbo C++的编程特性。

Turbo C++最重要的结构——类——属本章"预"讲内容。类是面向对象编程（一种新的编程风格）的关键，应倍加注意。

这一章以一Turbo C++程序的例子做为结束，学完本章之后，你会清楚：

- C++与C语言有何关联
- 什么是类，它们与面向对象编程有何关联
- C++扩展C语言的三种基本方法
- Turbo C++程序是什么样的

1.1 什么是C++?

C++是C语言的扩充，它是由AT&T的Bjarne Stroustrup创立的。Stroustrup所添加的最为重要的扩充是类结构。它借助于60年代所研制的一种语言——Simula。如图1.1所示。早期的C++被认为是"带类的C语言"。

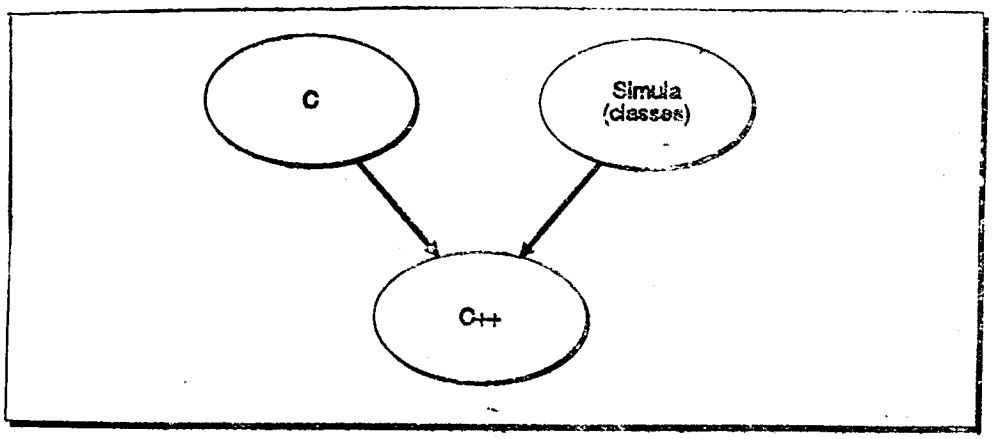

图 1.1 C++：带类的C语言

实际上，类是C语言的结构类型的扩充。有了类，就可以建立用户定义的类型，它们不仅表示数据，而且表示对该数据所要执行的操作。在类中所建立的变量称为对象，而用对象编程称为面向对象编程（OOP）。如今，面向对象的编程方法非常流行。

注释：C++在达到当前这一状况之前曾经历了数种版本。该语言的当前定义是C++2.0，而它是Turbo C++所实施的版本。

C语言千变万化，C++更是如此。C++包含了C的所有特性。然而，使用C++类可以开发出非常高级的程序。用类定义的类型可以象内部类型一样非常自然地嵌合到程序中。

C++对类型检查更为严格，特别是对函数调用。此外，C++提供了几种定义例程的方法。这些例程可以自动地被调用，以便对变量和对象初始化。这些初始化例程可减少由于使用非初始化数据而产生的许多错误。有了这些特性，就有可能用C++编写出更为安全的程序。

C++所添加的新特性仍含有C的精华，即：有效性，语法，以及对变量和函数的使用由程序员完全控制。C++试图尽可能地保持C语法。因此，如果你是一位C程序员，那么，你现在就可进一步了解在C++中更强功能的扩充。

1.2 C++对C的扩充

在C中，++运算符用于将变量值加1。C++这个名字就是由此而来的，因为C++是在C的基础上更上一层楼。所有熟悉的C结构都可用，如：赋值语句，for循环，while循环，if语句，函数，变量和结构。目前，C语言已标准化为ANSI C。C++综合了ANSI C对C语言的绝大部分修改，最为突出的是函数原型。

此外，C++还包含许多在ANSI C中没有的新特性。如图1.2所示，所提供的扩充可归为三类：

1. 使C++成为“bettet C”的杂项扩充。
2. 对数据类型系统的扩充，它使得用户能定义更多更强有力的类型。
3. 直接支持面向对象编程的扩充。

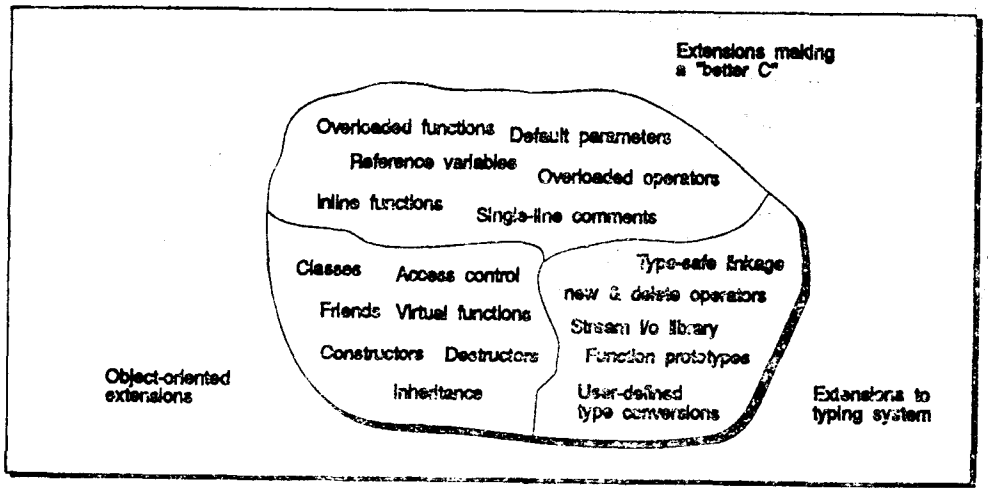

图 1.2 C++的特性

表1.1，1.2和1.3概述了每类中的扩充。请注意，表1.1和1.2中所列扩充并不是特定于面向对象风格，它们主要是支持该风格。

表 1.1 杂项扩充

扩充	目的
新式注释	使代码更具可读性。
直接插入函数	增强代码的效用。
引用类型	简化通过引用传递参数的语法。

续表

扩充	目的
函数重载	允许一组函数执行类似操作。
算符重载	允许标准C运算符与用户定义类型一起用，如：用+动算符将字符串加到一起。
缺省参数	允许指定函数参数的缺省值。

表 1.2 类型系统的扩充

扩充	目的
结构，联合和枚举是类型名	简化用户定义类型的语法。
函数原型	为函数调用提供更好的类型检查。
存储器分配	支持用户定义类型的动态分配。
运算符	便于更安全的动态存储器管理。
类型安全链接	通过在链接时执行类型检查，捕捉更多的参数传递错误。
新流库	允许在输入和输出用户定义类型中具有更大的灵活性。支持面向对象的输入/输出（I/O）。
用户定义类型转换	允许你指定待从一种类型转换到另一种类型的例程。

表 1.3 面向对象扩充

扩充	目的
类	允许函数和数据组合在一起。用于建立对象。
存取控制	允许限制对类中数据和函数的存取。
派生类	继承或复用现有类，并为了新的应用程序而扩充它们。
虚函数	函数调用的更强功能机制，其中，待调用的函数在运行时确定，而不是在编译时。
友元	允许有选择地存取类的其它受限制成员。
构造函数	用于为对象初始化建立用户定义例程。
析构函数	用于为执行对象的清理任务建立用户定义例程。

1.3 分析一个Turbo C++程序

让我们看一个Turbo C++程序。举例1.1是一个短小的C++程序，welcome.cpp。在Turbo C++中，源文件通常有一个.cpp扩展名，而不是在C中所用的.C扩展名。

[举例1.1] welcome.cpp程序

```
// welcome.cpp: Program to welcome you to Turbo C++
#inclde<iostream.h> //New stream I/O Library header
main()
```

```
{
  char name[80];                    //Variable declarations as in C
  cout<<"Enter your name: ";        //New way to output data
  cin>>name;                        //New way to input data
  cout<<"Welcome, "<<name<<", to Turbo C++\n";
}
```

该程序在结构上类似于C程序，但第一行是一种新式注释。用这一格式，每条注释以一对反斜杠字符开头，直到行尾。

该程序还说明了一种用iostream库（在第十一章讨论）进行输入/输出的新方法。iostream库分别用>>和<<运算符指示输入和输出。程序输入一个名字，然后用该名输出一条信息。

假设你已正确地安装了Turbo C++编译程序，下面是在Dos命令行上编译welcome.cpp：

```
tcc welcome.cpp
```

下面是welcome.cpp程序的运行举例：

```
Enter your name: Sam
Welcome, Sam, to Turbo C++
```

注意：你有可能在Turbo C++集成环境中编译。我们将在第二章说明如何做。

1.4 类

由于类是C++的最重要部分，所以，你也许希望看一看类到底是个什么样子。下面这段代码便是类的一个例子。

```
// Class declaration
class employee {
private:            // Start of private members
  int id;           // Employee's data
  char name[80];
  float wage;
public:             // Start of public members
  employee(int i, char *n, float w); // Constructor
  void print_payinfo(float hrs);     // Member function
};

// Class implementation

employee::employee(int i, char *n, float w)
// Implements the constructor, which is used to
// initialize an employee object
{
  id = i;  strcpy(name, n);  wage = w;
}

void employee::print_payinfo(float hrs)
// Member function to print out pay information
// for an employee. Note that printf() is still
// available in C++
```

```
{
  printf("Employee #%d: %s\n", id, name);
  printf("Hours worked:  %6.2f\n", hrs);
  printf("Amount paid: $%7.2f\n\n", hrs * wage);
}
```

这里不想详细地讲解类的定义，所以，在代码行中有不理解的地方请不要担心。在第六～九章中，你会有充分的机会了解类的详细情况。

类分为两部分：（1）类说明；（2）类执行。类说明类似于结构说明；它列出属于类的成员，类可有数据成员（事例变量）和函数（成员函数）。employee的事例变量是id，name和wage。成员函数是employee( )和print-payinfo( )。

成员函数的类说明中只是给出前向说明。这些说明称为函数原型。如果你对较新的C编译程序有所了解的话，那么，你以前已见过函数原型。函数体通常在类说明后定义，而成员函数定义集称为类执行。

成员函数employee( )是一种特殊函数，称为构造函数。每当建立employee对象时便调用该函数。下面便是建立employee对象并调用其构造函数的一个例子：

```
employee john (1, "John Jackson", 15.00);
```

该语句说明john为一个employee对象，并对john初始化：id为1，name为“John Jackson”，而wage为每小时15美元。给定的参数被传递给进行初始化的构造函数。

一旦建立了一个对象，就可以使用它，其方式与使用结构变量的方式一样。例如，你可以用点（.）存取对象的成员。不同之处在于，我们不仅可以存取事例变量，而且可以存取成员函数。因此，可调用属于某对象的函数。例如，调用对象john的函数print-payinfo( )：

```
john.print_payinfo (40.0);
```

该函数调用中，先给出工作时间数，然后打印出工资信息。这样，我们得到如下输出：

```
Employee #1: John Jackson
Hours worked: 40.00
Amount paid: $600.00
```

通过对象john调用print_payinfo( )函数，让john来决定要打印什么内容。这就是说，我们使程序面向对象。

在employee类说明中，你会注意到两个新的关键字private（私用的）和public（公用的）。这两个关键字允许你控制对类成员的存取，而且它们是用类编程的基础部分。基本说来，只有公用成员可在类外存取，私用成员内部于类，它们仅能由类的其它成员存取。

在employee类中，所有的事例变量都是私用的。这就是说，我们不能修改john的计时工资，因为wage是私用的。因此，如下语句是非法的：

```
john.wage = 100.0; //Can't do since wage is private
```

然而，我们可以存取employee( )和print_payinfo( )这两个函数，因为它们是公用的。

类的两个最主要特性是：它们可以拥有作为成员的数据和函数，你可控制对成员的存

取。这两个特性构成了封装。封装是面向对象编程中一个重要的概念。在第六章，你将会更多地了解封装。

1.5 一个Turbo C++程序实例的完整分析

我们把employee类做为例子，并根据它建立一个完整的程序。举例1.2为程序employ.ccp。这是一个短小的雇员工资程序。它定义employee类，建立某些employee对象，然后打印出有关雇员的工资情况。

使用如下命令行编译该程序：

```
tcc employ.cpp
```

程序的输出结果是：

```
Employee #1: John Jackson
Hours worked: 40.00
Amount paid: $600.00

Employee #2: Karen Butler
Hours worked: 52.00
Amount paid: $1144.00
```

[举例1.2] employ.cpp程序

```
// employ.cpp: A simple payroll program
#include <stdio.h>
#include <string.h>
// Class declaration

class employee {
private:            // Start of private members
  int id;           // Employee's data
  char name[80];
  float wage;
public:             // Start of public members
  employee(int i, char *n, float w); // Constructor
  void print_payinfo(float hrs);     // Member function
};

// Class implementation

employee::employee(int i, char *n, float w)
// Implements the constructor, which is used to
// initialize an employee object
{
   id = i;  strcpy(name, n);  wage = w;
}

void employee::print_payinfo(float hrs)
// Member function to print pay information
// for an employee
{
  printf("Employee #%d: %s\n", id, name);
  printf("Hours worked:  %6.2f\n", hrs);
  printf("Amount paid: $%7.2f\n\n", hrs * wage);
}
```

```
main()
{
  // Create some employee objects
  employee john(1, "John Jackson", 15.00);
  employee karen(2, "Karen Butler", 22.00);
  // Print out payroll information
  john.print_payinfo(40.0);  // John worked 40 hours
  karen.print_payinfo(52.0); // Karen workd 52 hours
  return 0;
}
```

1.6 小结

这一章介绍了C++较之C语言所增加的特性。所添加的某些特性对C语言会有所改进，如：类型检查和变量自动初始化。然而，所添加的多数特性是为了一个目的：支持面向对象编程。

面向对象编程的关键是类。类用于建立用户定义类型。这些类型可含有函数和与之相关的数据，而在这样的类型中所建立的变量称为对象。

1.7 练习

1) C++是一种支持面向对象编程的语言。请列出一些这样的语言。
2) 说出类的两个最主要特性。它们表示什么概念？

答案：

1) IBM PC上现有的其它一些OOP语言是：Smalltalk, Actor, Objective-C, Quick Pascal Turbo Pascal 5.5。
2) 类可拥有做为成员的数据和函数，你可通过public（公用的）和private（私用的）关键字控制对类成员的存取。这两个特性表示封装。