HOPE

计算机知识普及系列丛书

石 清 小 立 编写

Turbo C 培训教程

·该书是学习编写Turbo C程序的最好选择。初学者可从概论部分开始，而有经验的程序员同样可从这些程序和Turbo C高级功能演示学习中获益，因而都可提高Turbo C的编程能力，并使程序编得更加精炼、完善。

学苑出版社

计算机知识普及系列丛书

Turbo C 培训教程

石 清 小 立 编写

叶 恩 审校

学 苑 出 版 社

1993.

(京)新登字 151 号

内 容 提 要

本教程是学习编写 Turbo C 程序的最好选择。初学者可从概论部分开始，而有经验的程序员同样可从这些程序和 Turbo C 高级功能演示学习中获益，因而都可提高 Turbo C 的编程能力，并使程序编得更加精炼、完美。

本书从关于 Turbo C 的整体介绍开始，第一、二章为读者设计了一些原始程序，它们概括了 Turbo C 的所有结构和功能，以便读者能尽快地熟悉 Turbo C。

接下来，本书探讨组成 Turbo C 程序的各个部分，以下各章将详细介绍各数据结构、操作符、控制结构、函数、区域和存贮类、Turbo C 预处理程序和大量的 Turbo C 库函数。其中许多章节都包括了大量的示例程序。

欲购本书的用户，请直接与北京 8721 信箱联系，电话 2562329，邮码 100080。

计算机知识普及系列丛书

Turbo C 培训教程

编　　写：石　清　小　立
审　　校：叶　恩
责任编辑：甄国宪
出版发行：学苑出版社　　**邮政编码**：100032
社　　址：北京市西城区成方街 33 号
印　　刷：兰空印刷厂
开　　本：787×1092　1/16
印　　张：15.875　　**字数**：363 千字
印　　数：1～4000 册
版　　次：1993 年 12 月北京第 1 版第 1 次
ISBN7－5077－0821－7/TP・19
本册定价：13.00 元

前 言

目前，C语言已在专业编程中得到广泛的应用，大多数程序员在开发重大项目都喜欢采用C语言，这是由于C语言具有灵活性、便于移植和无可匹敌的易使用性。

Turbo C不仅继承了C语言的上述优点，并进一步建立了理想的开发环境，它的功能及其通用性和设计的简单性也大大提高了，程序员能比过去更快更容易地开发出较好的使用工具、实用程序和应用程序。Turbo C惊人的编译速度和所产生的代码质量都是无可匹敌的。

《高速Turbo C教程》是学习编写Turbo C程序的最好选择。初学者可从概论部分开始，而有经验的程序员同样可从这些程序和Turbo C高级功能演示学习中获益，因而都可提高Turbo C的编程能力，并使程序编得更加精炼、完美。

Turbo C是以适合于Turbo C初学者的方式介绍给读者的，因而没有C编程经验的程序员可发现大量的Turbo C程序和同类知识，因而可以较快且容易地学会Turbo C。

有经验的C程序员会发现程序有许多新的探索，以及关于Turbo C强功能演示和使用Turbo C的软件开发原则。

本书从关于Turbo C的整体介绍开始，第一、二章为读者设计了一些原始程序，它们概括了Turbo C的所有结构和功能，以便读者能尽快地熟悉Turbo C。

接下来，本书探讨组成Turbo C程序的各个部分，以下各章将详细介绍各数据结构、操作符、控制结构、函数、区域和存贮类、Turbo C预处理程序和大量的Turbo C库函数。其中许多章节都包括了大量的示例程序。

本书由我们在长期从事Turbo C程序设计基础上，经过认真编译而成，但由于时间仓促，加上我们水平有限，书中难免会有些错误和缺点，敬请广大读者批评指正。

本书出版过程中，提到了中国科学院希望高级电脑技术公司资料部秦人华经理、杨淑新老师的大力帮助和支持，在此向她们表示衷心的感谢！

目 录

第一章　概述

§1.1　C 和 Turbo C

若仅由 C 的保留字数量看，C 算小型语言，因而，一些人认为它相当容易学，然而 C 却是一门具有相当强功能的语言。用 C，可以构造大型复杂的软件系统，用少量的结构却能解决复杂的问题。C 语言和其包含许多 C 实用程序的标准库函数，为用户提供了基本工具以便用户在各层次的应用中构造软件组件。

作者认为 C 并不是初学编程者的最佳语言，一些更简单、功能较弱的语言，如 BASIC、Pascal 等，它们在介绍编程技巧时并不要求程序员懂得下属的机器。虽然 C 同样包括这些简单语言所有的编程技巧，但它还允许并鼓励程序员直接和机器内部的存贮地址和寄存器打交道，因此，它要求用户具有一定的机器常识。

与这些简单的语言相比，C 为程序员提供了高层次的灵活性，同时也付出了较高的代价。这些提高的灵活性可产生紧缩而有效的代码，但同时也引入了含糊、难读的代码。而因 C 程序员有责任写出易读和可维护的程序，所以，C 的有效使用要求学习、技能和经验。

与 C 相对的一些大型语言，如 PL/1T 和 Ada，为程序员提供了更多的预先设定的操作和结构。但这些优点常带来对编译速度、代码效益和代码大小的更高的要求和开销。

C 语言是 70 年代初期由 Bell 实验室的 Cennis Ritchie 设计的，因而它也算一门有较老资格的语言。它的首次实现是在 DEC PDP-11 上完成的，也许正是因为这个缘故，即使目前它已用于各类机器，C 仍普遍被认为用于小型计算机。

C 与 Porsche 相比，显示脆弱，低劣，却高速且效率高，C 还被认为是靠不停住的，这并不矛盾。因为恰恰是这些特点使 C 代码运行快且空间少，同时也使它易于产生被别的语言保护的错误。读者应警惕 C 本身的这种“灾难”，同时学会避免犯这些错误。

如果 C 是 Porsche，则 Turbo C 是 Turbo Porsche(也许 Turbo Nissan 300ZX 比 Turbo C 更合适，因为它的开销相对小些)。Turbo C 代码通常比别的许多 C 系统的紧缩性好，且代码运行速度快，还有较高的编译速度。这并不是通常都能兼备的。通常较快的编译是以代码优化差为代价的，而 C 却不是这样。

近几年来由于许多原因，C 变得流行起来，其中 C 代码的高速和紧缩是主要原因，尤其是面向微机和小型计算机的专业应用中，因为到目前为止，微机和小型机的硬件速度和存贮限制仍是很严重的。

C 表达式的经济性和它丰富的操作符集合也是它流行的原因之一，但这些属性也给 C 带来了一些潜在问题：缺乏可读性。缺乏可读性并不是 C 语言本身的问题，而是 C 程序员的问题，它是能够而且应该避免的。本书生成的 Turbo C 程序都格外小心，并且软件系统显示了较好的格式化技术。

虽然 C 最初，以及传统上都与 UNIX 操作系统有关，但它已经能够移到许多操作系统上，并最终能独立存在，不再依赖于别的操作系统了。Turbo C 最初创建于 MS-DOS 操作系统，所以焦点集中在该操作系统的低层应用上。将来 Turbo C 将用于 Microsoft's OS/2 或也许是别的操作系统上。

近几年常发生这样的情况，即当引入新的工作站计算机或处理机时，最先发布的两个翻译程序是针对特定处理机的汇编程序和 C 编译程序。别的高级语言编译程序往往随后产生。这既是 C 流行的结果，又是 C 流行的因素。

C 被定义为高级汇编语言，这是因为 C 语言允许程序员和计算机内部打交道——到位、字节和控制 CPU 与外设的寄存器。C 又比高级汇编语言强得多，它的块结构支持数据隐藏，同时也支持对变量域和可见性的高层控制。C 还提供了许多重要的已普遍存在于当代软件开发语言中的高层数据结构和控制结构。正如前面所指出的，大型和复杂的软件系统都采用 C。

C 是可移植性语言，C 程序的编写是针对所给的操作系统和特定的计算机系统的，但只需做很少的修改，便可移植到别的操作系统和机器。对伴随大多数 C 应用程序的库函数也是如此。由于 C 语言相对较小，所以它的大部分功能都来自于提供给 C 编译程序的库函数和用户编写的库函数，例如，简单的屏幕输入和输出并不是 C 语言的一部分，而是来自于大部分 C 系统所带的 Stdio(标准输入/输出)库。幸而，随着 C 语言的成熟和标准化，都具备 C 语言库。

Turbo C 是在 1987 年春由 Borland 公司介绍的，Turbo C 是专业水平的 C 开发系统，它包含了丰富了库函数，以支持普通的 C 编程和 8086 系列微处理机在 MS-DOS 下的编程；还有标准编译程序和连接程序，及一个完整的编辑器、编译器和连接器，和一个 make 实用程序；6 个不同的存贮模型支持 80x86 系列处理机，它们包含 64K 段边界出口。

Turbo C 是 80x86 系列微处理机上第一个主要的 C 编译程序，并完全遵守 ANSI C 标准。为了便于已熟悉 K&R C 的读者，那些在原 K&R C 系统中未实现，如今已加入 ANSI 标准中的(Turbo)C 新特性列出如下：

- 整型常量

  新关键字，signed，表明一个整型目标被说明。
- 浮点常量

  包括新的浮点类型：long double。在 1.0 版本中它被作为 double，在 1.5、2.0 版本中占 10 个字节。
- 指令

  #include 和#line 指令可包含宏(macro)。如果一个宏出现在指令行上，它将在指令执行前被扩展。
- 外部说明

  一个函数内部定义的 Extern 说明服从适当的块区域，这个声明在它所定义的块区域外不被识别。
- 函数原型

  这也许是旧的 K&R 标准中最重要的修改之一。函数原型(function prototype)是一个函数说明，它在该函数名后的一对括号中列出了该函数的所有形式参数。函数原型用于检查函数被调用时所用的参数类型和数目是否合法。想禁止这种新的类型检查，程序员可自由地采用旧的 K&R 函数。
- Macros(宏)

  在宏定义串中被嵌套的宏仅在宏本身被扩展时才扩展，而不是在宏定义中扩展。
- 预处理程序

预处理程序中支持一些新的指令、新的宏和宏中新的符号，它们有：_DATE_(宏)，_STDC_(宏)，_TIME_(宏)，_FILE_(宏)，_LINE_(宏)，#error(指令)，#pragma(指令)，##(令牌传递符号)。

- 类型转换

混合类型表达式的类型转换的缺省规则已被扩展到原 K&R 中所没有的新类型。

- 类型中断

type interrupt(类型中断)是一个函数类型修饰符，它允许中断函数返回一个值。

- 类型修饰符

修饰符 Const 和 volatile 被称为类型修饰符。Const 用来声明一个数据目标，其值不能被修改。Volatile 用来声明一个数据目标，它可以通过中断函数修改或在整体优化中可能被修改。此外，还有一些不包含在已提出的 ANSI 标准中的 Turbo C 特殊特性，它们是:

- 注释

注释可以有选择地被嵌套，这将为 Turbo C 代码的调试提供极大的帮助。

- 用于 80x86 系列的特殊关键字

asm、_cs、_ds、_es、_ss、cdecl、far、huge、interrupt、near、pascal、_AX、_AH、_AL、_BX、_BH、_BL、_CX、_CH、_CL、_DX、_DH、_DL、_BP、_DI、_SI、_SP。

- 预处理程序

预处理程序支持一些新的指令，新的宏和宏中的新符号。这些新特性有：_COMPACT、_HUGE_、_LARGE_、_MEDIUM_、_MSDOS_、_SMALL_、_TINY_、_TURBOC_。

- 汇编代码

内部汇编语言代码可以通过用关键字 asm 出现在 Turbo C 源程序模块中。

§1.2 本书简介

(Turbo)C 是一种快速灵活的语言。由于(Turbo)C 的灵活性，程序员可以写出紧凑而快速的代码，但难以读懂；也可写出可读性高的代码但效率较低。而通过练习和提高技巧，(Turbo)C 程序能编写出快速，紧凑的代码且可读性好。本书的题目正是由此而来。

本书针对想学习(Turbo)C 或想加深对 C 的理解并学习 Turbo C 的初学者或中等水平的 C 程序员，它假设读者有使用别的高级语言的编程经验，且其本身也是一种 Turbo“计算机文学作品”。

本书主要介绍 Turbo C 语言和它在构造 Turbo C 软件系统中的应用，因此本书的一部分用于学习 C。新的 ANSI C 标准特性和 Turbo C 的特殊特性贯穿本书，并在适当的时候给以介绍，关于 Turbo C 更新和专用特性的进一步学习安排在本书的后一部分。

本书列举了大量的程序和程序段来说明 Turbo C 的特性和编程方式。作者认为通过举例是学习和掌握 Turbo C 的最佳方法，尤其对 Turbo C 的初学者，而对于已有 C 经验的程序员，通过对示例的仔细分析可加深对 Turbo C 功能的理解，提高编程技巧，本书中采用的程序都保存了格式的一致性，尤其谨慎地保持代码的可读性和易理解性。

本书中列举的许多 Turbo C 程序和软件系统可解决计算机科学中一些重要应用域中的

问题。除了讲解 Turbo C 语言外，本书还给出了(Turbo)C 中软件开发的原则。通过对(Turbo)C 系统到接口和应用模型(文件)的适当分解来了解和举例说明复杂的软件系统的管理。

本书并不打算代替伴随 Turbo C 系统的《用户手册》，因此，Turbo C 编辑器的使用，和完整的编译器、连接器，或独立的编译器、连接器或 MAKE 实用程序均不属本书讨论之列。以上这些系统的使用说明请见《Turbo C 用户手册》。

§1.3 不输出“Hello World”的第一个 C 程序

在传统的关于 C 的书中，也许为了建立一个简单的 C 程序，总在第一个示例程序的第一行输出“Hello World”。本书将不按此传统，正如§1.1 节所指出的，虽然(Turbo)C 是一种小型语言，但并不是最简单的语言，第一个程序应反映出这一特点。

第一个 Turbo C 程序确实应短小但也许应脱离传统的“Hello World”程序。见程序 Listing1.1。虽然第一个程序很短小，但它使用了将在第 5、6 章介绍的一些语言特征：字符串，多维数组和布尔运算符。其目的在于对初学者体现出(Turbo)C 的风味。

Listing1.1 第一个 Turbo C 程序

```
/* Program first.c */
#include <stdio.h>
#include <string.h>
main( int argc, char* argv[] )
{
  if ( ( argc == 2 ) &&
      ( ( strcmp( argv[ 1 ], "Kahn" ) == 0 ) ||
        ( strcmp( argv[ 1 ], "Phillipe" ) == 0 ) ) )
      printf( "\n\n%s has done it again with Turbo C.\n",
              argv[ 1 ] );
   else
      printf( "\n\nBorland has done it again with Turbo C.\n" );
}
```

接下来我们来分析 listing1.1，然后读者自己再分析一遍。

/* Program first.C*/一行是注释行，它将被 Turbo C 编译器忽略。注释在 Turbo 中可嵌套，而在标准 C 中不能嵌套，它以双字符/*开始，并以双字符*/结束。

出现在程序顶部的两个#include 指令告诉编译程序在编译时用 stdio.h 和 string.h 文件的源代码代替这两条#include 指令。这些“包含”文件包括了这两个重要的且广泛应用的库文件的函数接口。

库文件 stdio.h 包含标准输入/输出(I/O)函数和宏函数，string.h 库文件包含普通的字符串处理工具。正如程序所表明的，输入/输出工具和字符串处理是(Turbo)C 语言之外的内容，因此程序员必须选择系统所提供的库函数，或者编写自己的用户库来支持输入/输出和字符串处理。

下一行，main(int argc, char*argv[])是第一个可执行模块——(Turbo)C 主程序的形式接口。所有的(Turbo)C 程序都必须包含一个函数 main。main 函数中的命令行变量 int

argc 和 char *argv[]允许用户输入一个或多个命令行字符串，并以一个或多个空白字符分隔开，它将用在程序 First 被 DOS 命令调用时的命令行中。

例如，用户可用命令“first kahn”或“first Phillipe 来调用 first 程序。这两种情况下，argv 的值(变量个数)均为 2。

程序名通常是命令行上的第一个字符串。对于 DOS3.X 版本，argv[0]指向命令行上的第一个字符串，而对于 DOS 的早期版本，程序名不作为变量，Turbo C 给 argv[0]赋值为字符串“C”。而 ANSI 草案在程序名不作为变量时，定义 argv[0]的值为""，即空字符串。

对于 DOS 3.X 版本，argv[0]所指字符串为"first"。在第一个例子中 argv[1]的值为"Kahn"，第二种情况，argv[1]字符串为"Phillipe"。

下一条语句，为了提高可读性被分解为几行，它执行一个逻辑测试，以判断 argc 为 2 与 argv[1]等于字符串"Kahn"或"Phillipe"的条件是否成立。若测试结果为真，则执行 if 测试的下一行:

```
printf( "\n\n%s has done it again with Turbo C \n",
        argv[1] );
```

这是一条输出语句，它显示用 argv[1]代替替换符%S 的字符串。控制符\n\n 表示屏幕光标在显示字符串时下移两行(即两次换行)。由 stdio.c 库函数和 Turbo C 系统提供的函数库实现的输出函数 printf，是一个多功能函数，将在第 10 章详细介绍。

如果测试结果为假，则执行以下输出语句:

```
printf( "\n\n Borland has done it again with Turbo C \n" );
```

测试条件

```
if( ( argc ==2 ) &&
  ( ( strcmp ( argv[1], "Kahn" ) == 0 )||
    ( strcmp ( argv[1], "Phillipe") == 0 ) ) )
```

用了逻辑“与”符号&&，和逻辑“或”符号||，以及库函数中的字符串比较函数 strcmp。

读者注意:

本书中大部分程序都在程序结束时用一个换行语句(回车、换行)结束，这可以通过以下输出语句完成:

```
printf( "\n" );
```

或在程序中最后一个可执行的输出语句 printf 中加入"\n"。这是因为在 Turbo C shell (完整的环境)中运行程序时，以提示"hit any key to continue"结束。如果 Turbo C 程序结束时无回车换行，则这个提示信息将写到程序输出的最后一行上。

§1.4 “开胃”的 Turbo C 编码

虽然本书的组织主要是从底自上，但它有利于初学者尽早对(Turbo)C 的风格和能力预先有一个大概了解。

本节主要完成两个任务，介绍 util.h 文件中有用的实用库函数的接口，并举例说明 util.c 文件中这个库的实现。这个实用函数库包括以下函数:清屏，定位屏幕光标，取得当前光标位置，从屏幕上移动闪烁光标，恢复闪烁光标，得到当前计算机时钟时间: 小时、分、秒和百之一秒，计算命令 begin 和 end 间有限代码块的运行时间，从带"hot-key"热键

输入的键盘输入字符，输入以 Enter 键结束的字符串，屏幕上字符串对中，判断键盘输入是否出现，判断用户用字符 y，Y n 或 N 回答的 yes/no; 冻结程序的运行直到用户键入空格，和有效地产生从 0 到 1 的实数以及从低到高的整数。其中一些实用函数将在以后的学习中用到。util.c 文件的源程序见 Listing1.3，它为初学者展现了 Turbo C 的部分风格，其中有对操作系统(DOS)的低层调用，和 8086 寄存器变量的赋值。读者可在学完第 1 到 6 章后再读 Listing1.2 和 1.3。

Listing1.2 通用的实用函数库接口

```
/*
  Interface to general-purpose utilities
  File util.h
*/
enum timetype {begin,end};
void getxy( unsigned *h,unsigned *v );
void gotoxy( unsigned h, unsigned v );
void clrscreen( void );
void remove_cursor( void );
void restore_cursor( void );
void gtime( unsigned *hour,
            unsigned *minute, unsigned *sec,
             unsigned *hund );
void rpttiming( enum timetype p );
float rand_real( void );
int random( int low, int high );
void get_key( unsigned char *ch );
void readstring( unsigned char s[] );
void centermessage( unsigned char s[] );
void spacebar( void );
int keypressed( void );
int yes( void );
```

listing1.3 通用实用函数库的实现

```
/*
  Geneal-purpose utility package
  File util.c Turbo C Version
*/
#include<stdio.h>
#include<dos.h>
#include"util.h"
static const char null='\0';
static const int max=32767;
```

```
 static int ins=0;
 static unsigned h1,h2,m1,m2,s1,s2,hund1,hund2;
 static unsigned seed1,seed2;
 static int first=1;
 void getxy( unsigned *h, unsigned *v )
 {
   union REGS regs;
   unsigned int result;
   regs.x.ax=0x300;
   regs.x.bx=0x0;
   int86( 0x10, &regs, &regs );
   result=regs.x.dx;
   *v=result/256;
   *h=result-*v*256;
 }
 void gotoxy( unsigned h, unsigned v )
 {
   union REGS regs;
   if( ( h >= 0 ) && ( h <= 79 ) && ( v >=0 ) &&
     ( v <=24 ) )
    if ( ( h !=79 ) || ( v !=24 ) )
    {
     regs.x.ax=0x200;
     regs.x.bx=0x0;
     regs.x.dx=h + 256 * v;
     int86( 0x10, &regs, &regs );
    }
 }
 void clrscreen( void )
 /* Reference ANSI.SYS, IN DOS 2.1, 3.0, 3.1,and 3.2
    clear display and home cursor.
 */ {  unsigned esc= 27;
   printf("%c[2J]%c[0;of", esc, esc );
 }
 static void convert( unsigned char *ch)
 {
 /*
```

Used for converting keys that have extended ASCII codes (such as the function keys and ALT keys) to upper ASCII values. These values can be associated with constants that refer to specific keys.

```
*/
  switch ( *ch )
  {
    case 59 : *ch = 128;
              break;
    case 60 : *ch = 129;
              break;
    case 61 : *ch =130;
              break;
    case 62 : *ch =131;
              break;
    case 63 : *ch = 132;
              break;
    case 64 : *ch = 133;
              break;
    case 65 : *ch = 134;
              break;
    case 66 : *ch = 135;
              break;
    case 67 : *ch = 136;
              break;
    case 68 : *ch = 137;
              break;
    case 71 : *ch = 167;
              break;
    case 72 : *ch = 161;
              break;
    case 73 : *ch = 165;
              break;
    case 75 : *ch = 164;
              break;
    case 77 : *ch = 163;
              break;
    case 79 : *ch = 168;
              break;
    case 80 : *ch = 162;
              break;
    case 81 : *ch = 166;
              break;
    case 83 : *ch = 169;
```