Turbo C (2.0 版)使用和参考手册

[美] BORLAND 公司著

叶新恩 编译

上海科学普及出版社

引　言

Turbo C 是为这样几类人编写的：需要一个速度快，效率高的编译器的 C 程序员；为利用 Turbo C 的优点而学习 C 的 Pascal 程序员，以及那些以一个快速易用的工具为开始的 C 学习者。

C 语言是一种结构化、模块化、可编译的通用程序设计语言，它被广泛地用于系统程序设计。其良好的可移植性使得用 C 编写的程序能很容易地从一个系统转换到另一个系统上。因此，几乎所有的程序设计任务均可用 C 语言来实现。

Turbo C 软件包

Turbo C 2.0 软件包由源盘和手册（《Turbo C (2.0 版)使用和参考手册》）组成。源盘包括所有的程序、文件和用于创建、编译、连接及运行 Turbo C 的库，还包括一些样本程序、实用程序、上下文相关的帮助文件、集成调试器、及本手册没有介绍的一些附加文献。

《使用手册》对新手来说是一个指导性的东西；对老手来说是一本很有价值的更新教材。《参考手册》尽可能地列出了所有 Turbo C 的细节，解释了 Turbo C 的扩充函数，还有有关 Turbo C 编辑器、错误信息、实用工具（CPP、MAKE、TLINK、TLIB、GREP、BGIOBJ 和 OBJXREF）、命令行选项、Turbo C 语法和 TCINST。初学者最好从《使用手册》开始，然后再读《参考手册》。

Turbo C 2.0 有哪些创新之处

Turbo C 2.0 包含了许多新的特点：

* 集成调试器：单步执行（Step Over 和 Trace Into）、设置断点、监视和计算表达式。
* 一个更快的编译、连接器（快 20~30%）
* 用作编辑缓冲区的 EMS 存储区
* 更快的内存分配及串函数
* 更快的浮点计算
* 新的增加 signal 和 raise 函数
* 一个允许用户在编译时向程序插入机器代码的_emit_
* 一个高级图形库，其中新增了许多函数，包括可安装的驱动程序和字体
* 支持命令行上的通配符
* 可连接生成小模式的.COM 文件
* 支持 Borland 新增的独立的 Turbo Debugger
* MAKE 实用程序的自动依赖关系检查
* 支持 long double 常数和变量
* 能够自动进行块缩进/回退及优化填充

配置要求

Turbo C 可在 IBM PC 系列机器上运行，包括 XT、AT、PS/2 和兼容机。Turbo C 要求 2.0 或更高版本的 DOS 及至少 448K 的 RAM；监视器应为 80 列。虽然我们建议两个软盘驱动器或一个硬盘带一个软盘驱动器的机器，但是一个软盘驱动器也是可以的。

Turbo C 包含有处理 80X87 协处理芯片的浮点例程，它对 80X87 协处理芯片进行仿真。80X87 协处理芯片能大大加快程序的运行速度，但并不必要。

Turbo C 的实现

Turbo C 实现了美国国家标准局（ANSI）建议的 C 语言标准，完全支持 Kernighan 和 Ritchie 的 C 定义。此外还包括了一些混合模式程序设计任选的扩充，进一步挖掘 PC 机的能力。

第一篇：《Turbo C 2.0 使用手册》

《Turbo C 2.0 用户手册》介绍了 Turbo C，展示了如何创建和运行程序，还包括一些有关编译、连接、调试及 Project MAKE 的背景信息，下面是《使用手册》的章节安排和主要内容：

第一章：Turbo C 的安装和启动。介绍如何在具体的系统上安装 Turbo C 和使用《使用手册》的一些建议。

第二章：Turbo C 2.0 程序设计初步。介绍使用 Turbo C 集成开发环境（TC）来加载、编译、运行、编辑、存贮一简单程序的基本手段。

第三章：Turbo C 2.0 的编译和运行。展示了如何使用 Turbo C Run 命令，解释怎样利用“make”来管理一个由很多分文件构成的大程序。

第四章：调试程序。通过对一个内含错误的例子程序的调试来向读者展示 Turbo C 2.0 集成调试器的各种特征。

第五章：Turbo C 2.0 集成开发环境。解释 Turbo C 文本编辑器、集成调试器和菜单系统，并讨论 pick 文件和配置文件。

第六章：Turbo C 程序设计技术。通过一系列的例子程序介绍包括创建和运行 Turbo C 2.0 程序的一些基本步骤。

第七章：Turbo C 进一步的程序设计技术。简要地介绍另外一些 C 程序设计元素，包括数组、指针、结构和语句。

第八章：Turbo C 视频函数。首先简要地讨论了视频模式和窗口，然后讨论在文本模式下和图形模式下进行程序设计的区别。

第九章：Turbo Pascal 程序员注意事项。用例子程序来比较了 Turbo Pascal 和 Turbo C，描述了这两种语言主要的区别，以及怎样避免一些不必要的麻烦。

第十章：Turbo Prolog 与 Turbo C 的接口技术。介绍 Turbo C 的模块如何与 Turbo Prolog 程序接口，并提供了一些例子来说明这一技术。

第十一章：Turbo C 语言参考。描述与 Kernighan 和 Ritchie 提出的 C 语言定义不一致的特征和所有方面的列表，详细地解释了 Turbo C 对目前 ANSI 标准中没有给出的一些扩充的细节。

第十二章：Turbo C 高级程序设计技术。描述了有关启动不同存储模式代码的内存组织、指针运算、汇编语言接口和浮点运算的使用，并提供了一些例子来说明这一技术。

第二篇：《Turbo C 2.0参考手册》

Turbo C参考手册是专为有经验的C程序员写的，提供了语言具体实现的细节及运行的环境。此外，还按字母顺序描述了每一个Turbo C函数的功能。下面是《Turbo C 2.0参考手册》的章节安排和主要内容：

第一章：Turbo C库函数的使用。依范围列出了Turbo C的#include(*.H）文件和各个库函数，讨论了main函数及其参数，最后是Turbo C全程变量的描述。

第二章：Turbo C库：按字母顺序列出了Turbo C库函数、入口规定、include文件、功能、返回值、可移植信息以及相关函数和怎样使用该函数的例子。

附录A：Turbo C交互编辑器：较《使用手册》中第五章更详细地解释编辑命令。

附录B：编译错误信息：列出并解释了每条错误信息，简要地说明了产生信息的可能的原因。

附录C：命令行任选项：列出了与用户有关的TCC（命令行编译器）任选项入口。

附录D：Turbo C实用工具：讨论了包含在Turbo C软件包里的实用程序：预处理程序TCC、MAKE管理程序、Turbo连接程序TLINK、Turbo库管理程序TLIB、文件搜索实用程序GREP、图形驱动程序和字体转换实用程序BGIOBJ和目标模块交叉参考程序OBJXREF。

附录E：Turbo C语言语法概要：用改进的BNF形式描述Turbo C的语法(本书略)。

附录F：TCINST：通过定做自己的Turbo C环境介绍，如何使用定做程序TCINST。通过TCINST可以定做自己的键盘、修改缺省值、改变屏幕的颜色等。

附录G：MicroCalc：解释了怎样编译、运行和使用包含在Turbo C源盘中的自举程序MicroCalc(本书略)。

第 一 篇

Turbo C 2.0 使用手册

第一章　Turbo C 的安装和启动

Turbo C 软件包实际上包括 C 编译的两个不同版本：集成开发环境版本和另外一独立的命令行版本。安装 Turbo C 系统时，请将源盘上的文件拷贝到工作软盘或硬盘上，源盘没有拷贝保护，有一个安装程序(INSTALL)可以使安装 Turbo C 成为一件简单的事情，源盘被格式为双面双密度，可由 IBM PC 及其兼容机读取，作为参考之用，我们在安装盘的 READEME 文件当中列出了源文件的名称。

假设用户对 DOS 命令已经很熟悉了，例如，用 DISKCOPY 来为源盘作备份。万一不知道怎样使用 DOS 命令，那在参考了 DOS 参考手册之后再来着手建立 Turbo C 系统吧。

拿到源盘之后作一整套工作拷贝，然后将原来的那套放到一安全的地方。不要从源盘上运行 Turbo C，它们仅仅是备份形式，免得损坏工作文件。

在这章里

在这章里，我们先介绍一下 README 文件和安装 Turbo C 系统的命令。其它部分则是一些建议——根据读者的编程经验建议下一步该阅读哪些章节。

README 文件

在安装 Turbo C 之前，先花点时间看看安装盘上的 README 文件是非常重要的，这个文件包含了可能在手册中还没有的最新信息，还列出了源盘上的各个文件，及各文件所含内容的一简要描述。

为了访问 README 文件，请先将安装盘放入 A 驱动器，键入 A:，按 Enter 将当前驱动器改到 A；然后键入 README 再回车，进入 README 之后，用 Up 和 Down 光标键翻滚读文件。按 Esc 退出。

安装 Turbo C 系统

在给用户的源盘之中，包含了运行集成及命令行版本编译的所有文件，另外还有启动代码及支持 6 种存储模式和 8086/80287 协处理器的仿真库。若是首次安装 Turbo C 或是版本升级（从 1.5），INSTALL 程序将会使之很容易。

如何将 Turbo C 安装到软盘系统上

若你的系统没有硬盘，但有一两个软盘驱动器，必须先准备三张格式化了的空盘，然后再运行 INSTALL。每次运行 INSTALL，它都会让你安装一种存储模式的 Turbo C，如果想安装几种模式，就得有几套盘，每一套对应一种模式。

运行 INSTALL

INSTALL 帮助用户逐步完成安装过程，用户所要做的也就是按每一步屏幕上提示的指令去做。应仔细阅读。

运行 INSTALL:

1 将标签为 INSTALLATION DISK 的源盘插入 A 驱动器

2 键入 A:回车

3 键入 INSTALL，回车

从现在开始，只要遵循 INSTALL 显示在屏幕上的指令即可。

INSTALL 运行完之后，就可以使用 Turbo C 了。

注： 如果想在退出了 Turbo C 集成开发环境之后，还能永久保持某些任选项，那么可用 TCINST 来轻而易举地完成这工作。参见《参考手册》附录 F。

如果计算机带的是液晶或等离子显示器，那么除了要完成上一节所提到的几个步骤之外，还得设显示屏幕参数才能使用 Turbo C。在命令行上键入 MODE BW80 之后再使用 Turbo C 效果最好。

此外，也可用定做程序 TCINST 来安装黑白屏 TC，见《参考手册》附录 F。在运行定做程序时，应从屏幕模式菜单中选“Black and White”。

阅读建议

既然已安装完毕 Turbo C，现就可以阅读本手册使用 Turbo C 了。但本使用手册是为四类不同的用户写的，而只有某些章节与读者的 Turbo C 编程需要有关，所以先花几分钟来阅读一下下面的内容，会快得多。

C 初学者

假如读者正在学习 C 语言，那么就从第二三章读起，这两章介绍 Turbo C 集成开发环境，讲解怎样加载、连接一简单 Turbo C 程序，以及怎样编辑存盘创建文件。第四章介绍 Turbo C 集成调试器；接下来是第六、七章，这两章是按教材的风格写的，让你一步一步地完成创建和编译。如果不太清楚集成开发环境的使用方法，则有必要读第五章。第八章介绍 Turbo C 图形功能。

有经验的 C 程序员

如果读者是一位有经验的 C 程序员，那么可以毫不费劲地将程序和 Turbo C 接上口。我们建议读第十一章“Turbo C 2.0 语言参考”，它概述了 Turbo C 2.0 同 Kernighan 和 Ritchie 及同 ANSI 标准草案的区别；另外还需要读第三章（Turbo C 的编译和运行），第四章（调试程序），以及第十二章（Turbo C 高级编程）。如果对 Turbo C 的图形功能感兴趣，那么就应该读第八章。

Turbo Pascal 程序员

第九章（Turbo Pascal 注意事项）是特意为 Pascal 程序员写的。我们举了几个例子，将 Turbo Pascal 程序同相应的 Turbo C 程序作了一个比较，对于两种语言的重大区别之处作了详细说明。

如果用 Turbo Pascal 编过程序，那么一定对编程的七个步骤很熟悉了。为适应 Turbo C，还得读第五章、第六章、第七章（如果使用过象 SideKick，Turbo Basic 之类的 Borland 菜单驱动产品，就跳过第五章），还应看看第三章中关于编译，运行 Turbo C 程序部分及第四章中关于 Turbo C 集成调试器的说明。

Turbo Prolog 程序员

如果读者用过 Turbo Prolog，想知道如何将 Prolog 模块同 Turbo C 接口，读第十章。

第二章　Turbo C 2.0 程序设计初步

装好 Turbo C 后本可以开始编程了，但是首先得知道一些基本步骤，如怎样运行 Turbo C，怎样使用文本编辑器来创建、修改程序，以及怎样编译连接它们。

当然了，可以使用 ASCII 文本编辑器来创建程序，然后用行编译命令（Turbo C 的 TCC）从 DOS 命令行上运行之。但是你会发现，Turbo C 集成开发环境（Turbo C 中的 TC）用起来要方便得多，因为它提供编辑器、Turbo C 菜单系统命令及集成调试器于一体。

注：我们将逐步解释怎样利用 TC 菜单来完成本章中的要求。欲知 TC 系统的详情，参看本手册第五章。

在这章里

先讲几个使用 Turbo C 的基本技术：加载 Turbo C 集成开发环境（TC），加载程序到 Turbo C，编译运行之。

接下来是怎么使用编辑器来修改程序。

最后，是怎样创建新的 Turbo C 程序，怎样在运行前存盘。

HELLO.C：加工、运行一单文件程序

先从简单的开始，在编写自己的 Turbo C 程序之前，让我们先用集成开发环境（TC）中现成的程序。

在安装例子程序的目录中有一叫 HELLO.C 的文件，它是一个非常简单的程序的源文件，通过这个例子，我们来介绍加工、运行一个单文件程序的六个步骤。

第一步：加载 TC

如果用 INSTALL 程序安装的 Turbo C，那么 TC 已存在于 Turbo C 主目录下了，只要进入该目录，在 DOS 命令行上键入 TC，回车即可。

注：如果想在另一单独的工作目录下来开发程序，则应告诉 DOS 在哪里能找到 TC：

* 用 DOS PATH 命令说明 TC 所在目录（见 DOS 手册的 PATH，注意不要冲掉已存 PATH）。
* 在 DOS 3.0 以上版本中，可以在命令行上使用文件全名。如\Turbo C\TC。

第二步：选择工作目录（任选的）

若当前目录包含 HELLO.C 则可跳过此步。

选择包含 HELLO.C 的驱动器及目录，这可通过下拉文件菜单来完成（按 F10，然后再按 F，或按 Alt-F）选择 Change Dir（使用光标键，将亮条移到位再按 Enter，或直接按 C），当新目录提示框出现，输入包含 HELLO.C 的目录名，回车便可。此目录随即变成当前目录。

注：当新目录提示出现时，它列出当前目录，即是说可用 File/Change Dir 选择来查看当前所在目录；方法是选择 File/Change Dir 后提示窗口出现，按 Esc 回去，并不修改当前目录。

第三步：建立工作环境

如果用 INSTALL 程序建立 Turbo C 系统，则工作环境已经设置好。但还是可以读读此节，以确信工作环境建立无误。

为了建立和保存工作环境，按 F10，然后按 O（或 Alt-O）从主菜单中调用 Options 菜单，然后选择 Directories 进入 Directories 菜单，有两个目录与我们有关：

Include Directories 和 Library Directories。

选择 Include Directories，然后输入有 Turbo C 标准包含文件（.h）的驱动器名及目录，目录名用分号隔开。包含目录通常是：

C：\TURBOC\INCLUDE 和 C：\TURBOC\INCLUDE\SYS;

应键入：

C：\TURBOC\INCLUDE;C：\TURBOC\INCLUDE\SYS

回车。

现在选 Libary Directories;输入包含 Turbo C 库文件及启动文件的驱动器名及目录。（一般是 C：\TURBOC\LIB）。其它目录名也可输入，但是必须用分号隔开。

注：这时候如果愿意的话，还可通过 Options/Directories/Output Directory 命令来设置输出目录（编译后的程序存放的地方）。如果这样做了，则所有编译和连接输出将放到这个目录中而不是当前目录。本例这种情况没有建立输出目录。

在非常简单的情况下，这些设置也就够了。

可将设置的工作环境保存到启动 TC 会自动加载的配置文件里。按 Esc 回到 Options 菜单，然后选 Save Options 将当前设置保存到盘上的配置文件中。配置文件缺省是当前目录的 TCCONTIG.TC。

注：当 TC 启动时，它在当前目录找 TCCONFIG.TC，若找到即加载之。如果愿意，还可以给配置文件另取一名字，这只要输入新名，回车即可。如果是这样，则下次启动 TC 时必须显式地加载配置文件，命令行上 TC 后跟\C 开关（见第五章有关命令行开关），或者使用 Options/Retrive Options 命令。

注：当工作于某一个程序时，每个程序所在目录配一缺省文件是很有用的。TC 也从那个目录启动。但是若在当前目录没有找到配置文件，还会到 Turbo C 目录中寻找。也就是说可将一般的配置文件放在 Turbo C 目录中，而其它配置文件簇在使用不同设置的源文件所在目录中。

第四步：将源程序加载到编辑器中

现加载 HELLO.C，从 File 菜单中选 Load 命令，或直接按加载文件热键 F3，这时出现一包含*.c 的提示框，输入 HELLO.C，回车。

注：如果不清楚要加载哪一个文件，或者想看看当前目录中的源文件列表就直接回车，不要输入文件名。TC 将显示出当前目录中的所有.c 文件的一个列表，利用光标键进行菜单选择。

HELLO.C 现已在 TC 编辑窗口中了，如下：

```
/*HELLO.C--HELLO WORLD*/
#include <stdio.h>
main()
{
```

```
    printf("HELLO WORLD\n");
}
```

注：本可以在命令行上同时加载 TC、源文件和配置文件。这里省了，主要是考虑到 2、3、4 步的问题。集成开发环境接收两个命令行参数：欲加载到编辑器的源文件名和/C 后跟文件名，两参数顺序是任意的，以下命令：

tc HELLO /cmyconfig

将加载 HELLO.C 源文件到编辑器，并且还加载 myconfig.TC（注意/C 开关和文件名不得有空格，.C 是欲编辑的文件的缺省扩展名，.TC 是配置文件的缺省扩展名）。

第五步：编译和连接

加工程序时，先得编译源代码，生成目标文件（文件扩展名为.obj），然后将目标文件送给连接器生成.EXE 可执行文件。连接器将运行库文件中的某些过程拷贝到目标文件中。（记住在设置工作环境时告诉 Turbo C 在哪里寻找这些库文件。）

对于单个文件程序的情况，不用建立 project 文件即可完成加工运行程序（project 文件见第六章）。

虽然加工程序还有其它方法，但最简单的方法还是按 F10，然后按 C（或直接按 Alt-C），产生编译菜单，选择 Make EXE File（或按热键 F9）。注意编译菜单告知哪些目标（.OBJ）文件将被编译，生成什么样的.EXE 文件。

注意：若程序有错，则在屏幕底的 Message 窗口显示错误及警告信息。如发生这种情况。确认一下程序是否正确之后再编译。

第六步：运行程序

这时，已得到一可执行文件。

现在，从 Run 菜单中选择 Run，接着按热键 Ctrl-F9 运行程序。

发生了什么现象?可见屏幕一闪，然后又回到了 TC 屏幕。选择 Run/User Screen 或按 Alt-F5 看用户屏。

用户屏应包含下面的消息：

HELLO, WORLD

看完之后，按任一键返回 TC 屏。

小结

现退出 Turbo C（从 File 菜单选 quit 命令，或按 Alt-X）。

让我们看看都创建了些什么。

在 DOS 提示符下，dir HELLO.*回车，得到下面的文件列表：

```
HELLO      C      104    5-11-88     2: 57P
HELLO      OBJ    450    5-11-88     3: 01P
HELLO      EXE    8884   5011-88     3: 01P
```

第一个文件 HELLO.C 是程序的源文件。其中是程序的正文（源代码），在 DOS 提示符下 type HELLO.C 命令可显示其中的内容。可以看见 HELLO.C 并不长，仅有 104 个字节。

第二个文件，HELLO.OBJ，是目标文件，其中包含了由 Turbo C 编译器产生的机器指令（目标代码）如果再用 DOS 的 TYPE 命令来显示该文件，得到的将是杂乱无章的一

片。

最后一个文件，HELLO.EXE 才是 Turbo C 产生的可执行文件，它不仅包含了 HELLO .OBJ 中的代码，还包括了一些由连接器从运行库文件拷贝而来的必要的支持过程。运行的时侯，在 DOS 提示符下给出文件名即可。不用.EXE 扩展名。

在 DOS 提示下键入 HELLO，回车，运行 HELLO.EXE，HELLO,WORLD 将出现在屏幕上，然后又重新出现 DOS 提示符。

编辑一个程序

在经典著作如 kernighan 和 Richie 的《The C Programing Language》中的第一个 C 程序都是 HELLO,WORLD 程序。这就是刚才加工运行的那一个程序。

我们现在对 Turbo C 集成开发环境略熟悉一些，可来试试自己编写的程序了。我们先从修改 HELLO.C 开始。以此学习使用编辑器。

如果不在 Turbo C 中，在 DOS 提示下键入 TC HELLO，再回到 TC，这时该程序已被加载到 TC 里了。如果想输入代码，只要将光标移到右边，再键入代码即可。Ctrl-Y 删除一行，Ctrl-N 插入一行。要保证在插入模式（单词 INSERT 应出现编辑窗口上边的状态上;如没出现，按 Ins 切换）。（请参阅《Turbo C 2.0 参考手册》的附录 A）。

再编辑程序，看上去应是这样的：

```
#include <stdio.h>
main()
{
    chhar name[150];
    printf("What's your name?\n");
    scanf("%s",name);
    printf("HELLO,%s\n",name);
}
```

可见 HELLO.C 增加了三行：第一行（char name[150];）声明了一变量名，可容纳 150 个字符（字母、数字、标点等。）（150 号位留作特殊字符用，以后再说。）；增加的第二行调用 printf 打印消息"What's your name?"；第三行调用 scanf 来读入一行到 name 中。

接下来，按 Ctrl-F9 来运行程序，注意 Turbo C 很聪明，它知道源程序被修改过。因此在运行之前还要编译连接一下。

这次运行时，发生的两件事：用户屏出现，并打出消息"What's your name?"，光标停在下一行。输入名字后回车，按 Alt-F5，用户屏打印出 HELLO，<名字>。注意只读输入了第一个单词，学了第六章就知道为什么会这样。现在按任意键返回 TC 屏。

如果程序有错会收到错误和警告。错误（error）是阻止编译生成目标码的程序错，警告（warning）则是指出一可能的问题。错误和警告的类型在屏幕底端的（Message）窗口中。错误和警告的类型是各式各样的，详见《Turbo C 2.0 参考手册》附录 B。

输出到打印机

关于怎样输出到打印机，这里并不打算深入讨论打印的过程，有些有趣的问题以后再讨论，这里简要说明一下。

加载 HELLO.C 后，把它修改成这样：

```
#include <stdio.h>
main()
{
    fprintf(stdprn,"HELLO,WORLD\n);
}
```

一定要保证打印机是准备好的，跟以前一样按 Ctrl-F9，编译、连接。打印机应打印出消息 HELLO，WORLD。

注意这次我们用的是 fprintf，而不是 printf。以后随着对 Turbo C 的日益了解，慢慢就会明白我们新增的这些函数了。

编写第二个 Turbo C 程序

现在再来修改一下 HELLO.C，把它存到一个新文件。现应还在编辑器里，但如果不在（没有闪烁光标），按 Alt-E; 或激发菜单系统，后按 E 选编辑器。现作如下改变：

```
#include <stdio.h>
main()
{
    int a,b,sum;

    printf("Enter two numbers:  ");
    scanf("%d %d",&a,&b);
    sum=a+b;
    printf("The sum is %d \n",sum);
}
```

可见作了五处改变：

* 将 name 定义改为其它变量的定义（a，b 和 sum 均为整数）
* 改变了 printf 语句中的消息
* 改变 scanf 语句中格式串及变量表
* 增加了一条赋值语句 sum=a+b;
* 改变了 printf 语句中的格式串和参数表。

不要被% & \ 弄糊涂了，第六章再作解释。

写盘

现在，不要按 F2，否则又存到 HELLO.C 中了（我们想存到另一个文件中）。

按 Alt-F 进入 File 菜单，再按 W 选择 Write To 命令，Turbo C 要求输入程序的新名，键入 sum.c，回车，程序就存到盘上 sum.c 文件中。

运行 sum.c

按 Ctrl-F，Turbo C 将重新编译程序。如果有错，再回到编辑器，看看是不是和我们这里给出的一致。

一旦没有了错误，Turbo C 将连接上相应的库程序，再运行该程序。用户屏幕显示：

```
Enter two numbers.
```

程序正等待输入两个整数，用空格、制表或回车隔开。输入完第一个整数后一定要回车。程序在屏幕上打印出两者和。选 Run/User Screen(或按 Alt-F5)看结果。击任一键返回 Turbo C。

太好了，现在已用程序的几个基本结构写出了两个完整的 Turbo C 程序。欲知这些结构的情况，请阅读第六、七章。

第三章　Turbo C 程序的编译和运行

现在我们对使用 Turbo C 已经有些经验了，因此这里再介绍一点更复杂的技术——Turbo C 集成开发环境及命令行 Turbo C 更高级的特征。

Turbo C 为 C 程序的开发提供了一个灵活的环境；这就是开始时对缺省任选的设置。但可改变这些缺省值以适合个人需要。Turbo C 也提供各种支持工具来管理与开发程序有关的杂务，如错误跟踪及文件系统的管理。

如果对 Borland 的集成开发环境（TC）还不太熟悉，应仔细看看第五章，再通过菜单系统编译运行程序。这是一个有逻辑的易学的系统，花不了多少时间即会感觉自在。

在这章里

因为既可以从集成开发环境，也可以从 DOS 命令来编译、运行程序，所以这章里我们两种方式都讨论。因集成开发环境是一个完整的、强有力的、易使用的软件包，我们认为读者愿意首先了解它。

作为开始，我们先简单地回顾一下怎样通过集成开发环境来编译连接 Turbo C 源代码，最后生成可执行代码，然后再讨论 TC 的调试特征。

在解释了怎样在集成开发环境中运行程序之后，再说说使用命令行来编译、连接、MAKE 和运行 Turbo C 程序。除了集成开发环境版本的 Turbo C 之外，还有一套单独的编译器 TCC、连接器和 MAKE 实用程序。有关单独版本的详情请见《Turbo C 2.0 参考手册》附录 C。

小结

在 Turbo C 集成开发环境中建立一新程序，通常有以下几个步骤：

1. 设置目录任选项，以便编译、连接器知道在哪里查找和保存文件.
2. 加载文件到编辑器。（注：如程序有一个以上的模块组成，就必须建立 project 文件，其中列出模块文件名）。
3. 生成可执行文件。

调试

查找和修改程序中的错误通常是令人讨厌的事情。Turbo C 集成开发环境（TC）提供了一调试装置，使得这一工作容易了许多。调试达到了编译和运行级。

跟踪语法错误

使用 TC 最好的理由之一是它允许用户修改语法错误（编译时）和评估编译器给出的警告。TC 将编译器和连接器发出的消息收集到一缓冲区中，然后在消息窗口中显示，这样在访问源代码的同时，还能一下看到这些消息。

现给 HELLO.C 制造一点语法错误，将第一行包含语句中的#去掉，再去掉第五行 printf 语句中的后引号。现在程序看上去是这样的：

```
include <stdio.h>
main()
{
    printf("HELLO,WORLD \n);
}
```

按 Ctrl-F9（编译热键）重新编译之。编译窗口将显示错误和警告：应为两个错误，0 个警告。

1. 消息窗口（Message Window）

当看见编译窗口中的 Press any key 提示时，按空格键，消息窗口立刻被激活，亮条出现在第一个错误或警告上，这时编辑窗口中也会有一亮条一它标志着编译器给出的错误或警告在源代码中的相应位置。

这时可用光标键将消息窗口中的亮条上下移动，注意到编辑窗口中的亮条也随着跟踪源代码中错误发生的位置。如果将亮条置于“compling”上，则编辑器显示文件的最后位置。

如果消息窗口太长看不见，可用左、右光标水平滚动消息，为了一次能够多看点信息，可按 F5 放大消息窗口。放大后，编辑窗口不可见了，因此不进行错误跟踪。现在，保持分屏模式。

2. 改正一语法错误

为了改正错误，将消息窗口中的亮条置于第一个错误消息上，回车，光标移到编辑窗口中错误产生处，注意，编辑器状态行给出所选消息（这在放大模式下是有用的）改正之。（将第一行拿走的#放回去）。

当不只一个错误时，可用两种方法来修改下一错误。

第一种方法和前面一样，按 F6 回到消息窗口选择想修改的下一条消息。

第二种方法不用回到消息窗口，只要按 Alt-F8，编译器就会将光标移至消息窗口中列出的下一个错误 。按 Alt-F7 可移至前一个错误。

这两种方法各有长短，视情况而定。有时源代码中一个愚蠢的错误把编译弄糊涂了，产生好多消息，这时选择修改第一条消息就使得其余的一些错误消息没有什么意义了，这种情况发生时，使用方法一会方便些——修改完第一个错误之后回到消息窗口，再滚动到下一个有意义的消息上，选择之。在别的情况下，按 Alt-F8 会方便得多。

记住，Alt-F7 和 Alt-F8 是热键，TC 中无论何时均起作用。因此，在消息窗口中按 Alt-F8 得到的不是当前亮行消息，而是下一个消息。（按 Enter 选择当前消息。）但如果没别的编译消息，Alt-F8 就不起作用了。

注：可以如此法选择连接消息，但它们不跟踪源文件。在修改语法错误的过程当中，经常需要增加、删除正文，编辑器是记住的，依然能正确定位错误位置。没有必要记住行号和增加、删除的正文行。

跟踪运行错误：集成调试器

一旦修改好语法错误之后，程序编译就没什么问题了，但还是可能不按要求运行，因为可能有逻辑错误（运行错误）。这种错误跟踪就无助于发现错误位置了。

TC 有一个集成调试器可以跟踪运行错误。通过调试器可以运行，在断点处暂停，检查变量的值，甚至可以改变之，以看程序会有什么反应。第四章有一个怎样使用 TC 集

成调试器的自学教材。

Project：使用多文件程序

TC 最显著特点之一就是能进行分别编译，TC 的 Project-Make 更加有效。

在第二章中的例子里，我们涉及的都是单个文件，所以用 Compile/Make EXE File 命令即可以生成可执行文件了。对于多个文件的程序（多于一个文件），就必须告诉 TC 都有哪些文件。也就是说必须建立一个 project 文件。

创建 project 文件同.c 文件名列表一样简单，正如以后将会看到的，可以在 project 文件中列出很多不同的文件，但我们还是简化一下，就两个文件。

一个基本情况是：一个主文件和主文件将访问到的包含数据和函数的支持文件。例如，主文件 MAIN.C:

```
#include <stdio.h>
main(int argc,char *argv[])
{
  char *s;
  if (argc > 1)
     s = argv[1];
  else
     s = "the universe";
  printf ("%s %s.\n",GetString(),s);
}
```

支持文件 MYFUNCS.C：

```
ss[] = "The restaurant as the end of";
char *GetString(void)
{
  return ss;
}
```

请先建立好 MYMAIN.C 和 MYFUNCS.C。

让我们来构造这两个文件的 project 文件。project 文件仅包含将被编译、连接的文件名；本例中为：

```
mymain
myfuncs
```

扩展名.c 是不必要的，TC 假设无扩展名的文件都是.c 文件（当然了，加上.c 也是可以的）。文件的顺序是无所谓的，只是影响编译的顺序。下面的 project 文件跟前一个最终结果相同：

```
myfuncs
mymain
```

现在将它存到 MYPROG.PRJ 中（从 File 菜单中选择 Write to），project 文件就算完成了。

注意：project 文件的名称（MYPROG.PRJ）和主文件（MYMAIN.C）不一样，本来就