

科海培训中心系列教材

C

语言

高级实用教程

尹彦芝 编著

清华大学出版社

北京科海培训中心系列教材

C 语言高级实用教程

尹彦芝 编著

清华大学出版社

内 容 简 介

本书从比较深入的角度详细地阐述了一般 C 语言的参考书中很少提及,而实际应用中又很重要的 C 语言编程的一些难题,如中断处理程序、插入程序、pop_up 图形窗口、文本窗口及其编辑、菜单及其编辑、内存动态分配、程序管理、基指针的使用等等。在每一个问题的阐述中都给出了一个以上的实例,这些实例完整而详尽,又都是开发于目前最流行的 Turbo C、MS C、Turbo C Tools 等编译工具上,极方便于读者的调试与学习。

本书适宜于计算机专业及相关专业的本科高年级学生和研究生,以及 C 语言程序开发人员,他们已掌握了 C 语言的基本知识,进而想在 PC 机的 DOS 操作系统下开发更高级的软件,或者想更为深入地了解 C 语言的一些细节和技巧。

(京)新登字 158 号

JS318/13

C 语言高级实用教程

尹彦芝 主编

☆

清华大学出版社出版

北京 清华园

北京市朝阳区经伟印刷厂印刷

新华书店总店科技发行所发行

☆

开本:787×1092 1/16 印张:29.75 字数:757 千字

1991 年 10 月第 1 版 1992 年 10 月第 1 次印刷

印数:00001~10000 册

ISBN 7-302-01100-1 / TP · 412

定价:19.50 元

前 言

C语言是当前最流行的程序设计语言。本书是针对这样的读者而编写的：他们已熟悉和掌握了C语言的基本编程技术，现在想在PC机DOS操作系统下用C语言开发更完美的软件，或者想更为深入地了解C语言的一些细节和使用技巧。

为此，我们编写了这本《C语言高级实用教程》。

本书既不是仅仅简单地罗列C语言的语法和函数，也不是仅仅提供一些可以整块移植的实用程序，而是将重点放在对一些一般的C语言的教材和参考书中极少提及、而实际应用中又极为重要的C语言编程的技术难点上（如中断处理程序，插入程序，Pop-up图形窗口，文本窗口及其编辑，菜单及其编辑，内存动态分配，程序管理，基指针的使用等等）。为了加深读者的理解并给读者编程提供更多的方便，本书亦提供了许多演示程序和9个工具包及工具包的应用实例。这些工具包所提供的是一些针对某一类型的任务很有用的工具函数，这可作为对C语言函数的重要补充。因此称本书为“实用教程”。

同时，本书亦是一本“高级实用教程”。浏览一下本书的目录便可知道，本书不是按一般C语言教科书的“数据类型”、“表达式”、“控制语句”（假设读者已具备了这些基础知识）这样的方式来编写的，而是按照一般编程实践中所要处理的实际问题来划分章节。全书共15章，除第1章是阐述一些共同性的难点外，其余各章都是一章一个专题，依次是：编译模式和内存的组织，鼠标输入，文本屏幕输出和文本窗口，键盘输入、菜单及编辑，文件处理基本技术，字符串的处理，动态通用串的处理，文件处理高级技术，内存和程序管理，Microsoft C 6.0中的基指针技术，与DOS和BIOS的接口，中断服务程序，图形处理，混合模式与混合语言编程。

相对于其他语言来说，C语言有一个很重要的特点，这就是标准的C语言学起来并不困难，初看上去甚至很容易，但是，C语言程序的真正实现都是基于其丰富的函数的，而这些函数随C编译系统的不同而有所区别，这使得要想真正掌握C语言的编程比起仅仅掌握C语言的语法本身来说要困难得多。Turbo C 2.0大约有440多个函数，Microsoft C 6.0提供的函数更多，更不用说最新推出的面向Windows 3.0编程的Borland C++ 3.0和Microsoft C/C++ 7.0了。掌握与查询这些函数的最好办法莫过于正确地分类。Turbo C 2.0将其库函数分为16类。本书也采取分类的办法，但分类的原则与Turbo C的手册略有不同。当然，由于函数太多，本书也只能涉及其中的一部分。对于较易理解的函数，如数学函数等，本书根本未予提及，因为只要查阅手册便清楚了。而对于一些较重要而又较难掌握的函数，本书花的篇幅较多一些。书中除“动态通用串的处理”、“文件处理高级技术”这几个专题外，其余各章均从相应的C函数开始详细讲解这些函数的功能与限制，并以此为基础，提供一些演示程序或小实例或扩展出一些新的函数。

对国内为数众多的微型计算机用户来说，Turbo C 2.0和Microsoft C 6.0是大家目前较为熟悉的编译系统。本书的讲解大多以Turbo C 2.0为线索，而在某些章节，如键盘输

入、窗口输出、菜单的处理、域编辑、中断服务程序和插入程序等，Turbo C Tools 系统中有了明显的改进或补充，本书将花较多的篇幅予以介绍。全书仅在第 11 章“MS_C6.0 的基指针技术”中的例子要求使用 Microsoft C 6.0，因只在这一编译系统中才提供了这种技术。美国 Blaise Computing 公司 1989 年推出的 Turbo C 工具库 Turbo C Tools 6.0 适用于 Turbo C 的 1.0、1.5 和 2.0 版本。这个工具库提供了丰富的库函数，如字符串转换、屏幕操作、窗口、菜单、鼠标器、键盘、文件、打印机、内存管理、中断服务、插入程序、编辑器、在线帮助等函数资源，充分利用好这些函数，可使用户很方便地设计出更为精美的用户界面，使得用户编写出的应用程序在外观上和操作上达到一个新的境界。

更深入地掌握 C 语言，更“优美”地编写 C 程序，这正是我们希望读者通过本书的学习所能达到的目标。但由于作者水平有限，书中所介绍的又都是近年来不断发展变化着的新技术，在全面与准确地把握各种 C 编译的系统与环境方面，难免有疏漏，欢迎读者来函指正。

如果读者对一些常用的实用程序和算法感兴趣，可参考由清华大学出版社出版的《C 语言常用算法和子程序》一书。

作者

1992 年 6 月

目 录

| | |
|------------------------------|-------------|
| 前 言 | (1) |
| 第 1 章 几个重要问题 | (1) |
| 1.1 数据类型转换 | (1) |
| 1.1.1 各类整数之间的转换 | (1) |
| 1.1.2 实数与整数之间的转换 | (2) |
| 1.1.3 指针之间的转换 | (3) |
| 1.2 指针 | (3) |
| 1.2.1 指针说明 | (3) |
| 1.2.2 指针与地址 | (3) |
| 1.2.3 指针运算 | (4) |
| 1.2.4 指针分类 | (4) |
| 1.2.4.1 近(near)指针 | (5) |
| 1.2.4.2 远(far)指针 | (5) |
| 1.2.4.3 巨(huge)指针 | (6) |
| 1.2.4.4 基(based)指针 | (7) |
| 1.2.5 各类指针之间的转换 | (7) |
| 1.3 函数 | (7) |
| 1.3.1 有返回值的函数 | (7) |
| 1.3.2 无返回值的函数 | (8) |
| 1.3.3 修改参数的函数 | (8) |
| 1.3.4 递归函数 | (9) |
| 1.3.5 参数个数不定的函数 | (9) |
| 1.3.6 函数指针及其应用 | (10) |
| | |
| 第 2 章 编译模式和内存组织 | (14) |
| 2.1 段与偏移量 | (14) |
| 2.2 六种编译模式 | (16) |
| 2.2.1 概述 | (16) |
| 2.2.2 微模式 | (18) |
| 2.2.3 小模式 | (19) |
| 2.2.4 中模式 | (20) |
| 2.2.5 紧凑模式 | (20) |
| 2.2.6 大模式 | (20) |
| 2.2.7 巨模式 | (20) |

| | | |
|------------|--------------------------|-------------|
| 2.3 | 堆栈的组织 | (20) |
| 2.4 | 堆的组织 | (22) |
| 2.5 | 其它内存操作函数 | (24) |
| 第3章 | 鼠标输入 | (26) |
| 3.1 | 鼠标驱动程序的基本功能 | (26) |
| 3.2 | 与鼠标接口的C函数工具包 | (27) |
| 3.2.1 | 14个工具函数 | (27) |
| 3.2.2 | 工具包应用举例 | (36) |
| 3.3 | Turbo C Tools的鼠标支持函数 | (38) |
| 3.3.1 | 鼠标的初始化 | (39) |
| 3.3.2 | 询问鼠标的状态 | (39) |
| 3.3.3 | 鼠光标的位置和速度控制 | (41) |
| 3.3.4 | 鼠光标的形状和开关控制 | (42) |
| 3.3.5 | 对鼠标硬件中断的处理 | (43) |
| 第4章 | 文本屏幕输出和文本窗口 | (45) |
| 4.1 | 概述 | (45) |
| 4.2 | Turbo C的文本屏幕处理 | (46) |
| 4.2.1 | 文本输出与操作 | (47) |
| 4.2.1.1 | TTY输出规则 | (47) |
| 4.2.1.2 | 输出文本 | (47) |
| 4.2.1.3 | 对屏幕内容和光标的操作 | (47) |
| 4.2.1.4 | 屏幕与内存之间文本的移动 | (48) |
| 4.2.2 | 窗口和显示方式控制 | (48) |
| 4.2.3 | 属性控制 | (49) |
| 4.2.4 | 状态查询 | (51) |
| 4.3 | pop_up文本窗口工具包 | (52) |
| 4.3.1 | 窗口结构和窗口栈 | (53) |
| 4.3.2 | 16个工具函数 | (54) |
| 4.3.3 | 工具包应用实例 | (65) |
| 4.3.3.1 | 用“瓷砖”式窗口制作菜单 | (65) |
| 4.3.3.2 | 移动pop_up窗口 | (67) |
| 4.3.3.3 | pop_up错误信息和正常信息 | (70) |
| 4.4 | Turbo C Tools的文本屏幕处理 | (77) |
| 4.4.1 | Turbo C Tools与Turbo C的比较 | (77) |
| 4.4.2 | 显示设备和显示方式 | (78) |
| 4.4.3 | 页控制 | (80) |
| 4.4.4 | 清除和滚动 | (81) |

| | | |
|--------------|-------------------------------|--------------|
| 4.4.5 | 光标控制 | (82) |
| 4.4.6 | 显示属性和颜色 | (83) |
| 4.4.7 | 屏幕写入和读取 | (84) |
| 4.4.8 | 矩形区域写入和读取 | (85) |
| 4.4.9 | 各种文本输出函数的速度比较 | (88) |
| 4.5 | Turbo C Tools 的窗口处理 | (89) |
| 4.5.1 | 概述 | (89) |
| 4.5.2 | 建立和注销窗口 | (91) |
| 4.5.3 | 显示和关闭窗口 | (93) |
| 4.5.4 | 窗口控制和状态 | (95) |
| 4.5.5 | 清除和滚动 | (97) |
| 4.5.6 | 光标位置查询和控制 | (98) |
| 4.5.7 | 属性控制 | (98) |
| 4.5.8 | 窗口输出/输入 | (98) |
| 4.5.9 | 虚拟窗口 | (103) |
| 4.5.10 | 常驻内存的窗口程序 | (106) |
| 4.6 | Turbo C Tools 的帮助信息窗口 | (106) |
| 4.6.1 | 帮助信息源文件 | (107) |
| 4.6.2 | 缺省帮助信息窗口 | (108) |
| 4.6.3 | 帮助函数 | (109) |
| 第 5 章 | 键盘输入、菜单和编辑 | (112) |
| 5.1 | Turbo C 的键盘输入 | (112) |
| 5.1.1 | 概述 | (112) |
| 5.1.2 | 控制台级(conio)键盘输入处理 | (113) |
| 5.1.3 | 标准文件级键盘输入处理 | (116) |
| 5.1.4 | 普通文件级键盘输入处理 | (116) |
| 5.1.5 | BIOS 级键盘输入处理 | (117) |
| 5.1.5.1 | 中断 0x9 | (117) |
| 5.1.5.2 | 中断 0x16 | (118) |
| 5.1.5.3 | bioskey 函数 | (118) |
| 5.2 | <Ctrl Break> 和 <Ctrl C> | (119) |
| 5.3 | Turbo C Tools 的键盘处理 | (120) |
| 5.3.1 | 键盘输入 | (120) |
| 5.3.2 | 缓冲区处理 | (121) |
| 5.3.3 | 状态控制键 | (124) |
| 5.3.4 | 使用加强键盘 | (125) |
| 5.3.5 | 使用键控制函数 | (125) |
| 5.3.6 | 取得键码 | (126) |

| | |
|---------------------------------|--------------|
| 5.4 Turbo C Tools 的菜单处理 | (126) |
| 5.4.1 概述 | (126) |
| 5.4.2 建立、显示和注销菜单 | (129) |
| 5.4.3 定义菜单项和按键鼠标事件 | (130) |
| 5.4.4 读取用户的选择 | (132) |
| 5.4.5 菜单应用举例 | (134) |
| 5.4.5.1 [例 1]: 一个简单演示程序 | (134) |
| 5.4.5.2 [例 2]: 用菜单实现电子表格 | (135) |
| 5.5 Turbo C Tools 的域编辑 | (140) |
| 5.5.1 概述 | (140) |
| 5.5.2 域编辑 | (144) |
| 5.5.3 编辑键定义 | (146) |
| 第 6 章 基本文件处理 | (149) |
| 6.1 目录 / 文件系统概述 | (150) |
| 6.1.1 文件存取级别 | (150) |
| 6.1.2 文件属性 | (151) |
| 6.2 系统级输入 / 输出 | (152) |
| 6.2.1 文件柄 | (152) |
| 6.2.2 文件柄存取字节 | (153) |
| 6.2.3 文件柄属性字节 | (154) |
| 6.2.4 文件出错处理 | (155) |
| 6.2.5 建立文件 | (155) |
| 6.2.6 打开文件 | (157) |
| 6.2.7 读取和设置文件的特性 | (158) |
| 6.2.8 读、写和关闭文件 | (159) |
| 6.3 标准级(流式)输入输出 | (160) |
| 6.3.1 FILE 数据结构 | (160) |
| 6.3.2 建立 / 打开 / 关闭 / 删除文件 | (162) |
| 6.3.3 取文件状态和出错处理 | (164) |
| 6.3.4 控制文件缓冲区 | (164) |
| 6.3.5 移动文件指针 | (166) |
| 6.3.6 字节级的读 / 写 | (167) |
| 6.3.7 字符串级的读 / 写 | (168) |
| 6.3.8 记录级的读 / 写 | (168) |
| 6.4 基本文件处理工具包 | (169) |
| 6.4.1 10 个工具函数 | (169) |
| 6.4.2 工具包应用 | (175) |
| 6.5 驱动器和目录操作 | (175) |

| | | |
|--------------|---------------------------------|--------------|
| 6.5.1 | 驱动器 and 驱动器信息 | (175) |
| 6.5.2 | 目录操作 | (177) |
| 6.5.3 | 文件名操作 | (178) |
| 6.5.4 | 目录搜索 | (178) |
| 第 7 章 | 字符串处理 | (182) |
| 7.1 | 字符 | (182) |
| 7.1.1 | 字符数据和常数 | (182) |
| 7.1.2 | 字符输入 / 输出 | (183) |
| 7.1.3 | 字符的分类和转换 | (184) |
| 7.1.4 | 宏和宏的副作用 | (184) |
| 7.2 | 字符串 | (186) |
| 7.3 | 字符串的分析 | (187) |
| 7.4 | 字符串的综合 | (191) |
| 7.5 | 字符串的操作 | (193) |
| 7.6 | 文本字符串 | (196) |
| 7.7 | 数字字符串 | (197) |
| 7.8 | 国家和货币字符串 | (202) |
| 7.9 | 日期和时间字符串 | (204) |
| 7.10 | 文件名字符串 | (208) |
| 7.11 | 命令行字符串 | (209) |
| 7.12 | 环境字符串 | (209) |
| 7.13 | 错误级字符串 | (210) |
| 7.14 | 字符串处理工具包 | (211) |
| 7.14.1 | 13 个工具函数 | (211) |
| 7.14.2 | 工具包应用举例:PASCAL 程序翻译为 C 程序 | (215) |
| 第 8 章 | 动态通用串处理 | (223) |
| 8.1 | 动态字符串 | (223) |
| 8.2 | 动态通用串 | (225) |
| 8.3 | 动态通用串工具包 | (227) |
| 8.3.1 | 8 个工具函数 | (227) |
| 8.3.2 | 工具包应用举例:多边形表示法 | (232) |
| 8.4 | 通用串的排序与查找 | (234) |
| 8.5 | 动态通用串与链表的比较 | (242) |
| 第 9 章 | 高级文件处理 | (244) |
| 9.1 | 变长记录(VLR)文件 | (244) |
| 9.1.1 | 从文件中查找一个记录 | (244) |

| | | |
|---------------|-----------------------------|--------------|
| 9.1.2 | 插入和删除记录 | (245) |
| 9.1.3 | 碎片化问题 | (245) |
| 9.1.4 | VLR 文件格式 | (246) |
| 9.1.5 | VLR 记录格式和数据块格式 | (246) |
| 9.2 | VLR 工具包 | (247) |
| 9.2.1 | 7 个工具函数 | (247) |
| 9.2.2 | 索引处理 | (255) |
| 9.2.3 | 工具包应用举例:制作和显示幻灯片 | (256) |
| 第 10 章 | 内存和程序管理 | (264) |
| 10.1 | PSP 和环境 | (264) |
| 10.1.1 | PSP | (264) |
| 10.1.2 | 环境 | (265) |
| 10.2 | 内存管理 | (267) |
| 10.2.1 | 内存块及其控制 | (267) |
| 10.2.2 | 内存分布映像程序 memrymap | (268) |
| 10.2.3 | 内存管理函数 | (270) |
| 10.3 | 多个程序的执行及通信 | (270) |
| 10.3.1 | 程序间的通信 | (270) |
| 10.3.2 | spawn: 调用子进程 | (271) |
| 10.3.3 | exec: 转到子进程 | (275) |
| 10.3.4 | system:执行 DOS 命令 | (275) |
| 10.3.5 | signal 和 raise:事件处理 | (276) |
| 10.4 | 标准输入 / 输出重定向 | (278) |
| 10.4.1 | [例 1]:freopen.dem | (279) |
| 10.4.2 | [例 2]:dup.dem | (280) |
| 10.4.3 | [例 3]:利用 system | (282) |
| 10.5 | 程序的终止 | (282) |
| 第 11 章 | MSC 6.0 的基指针技术 | (287) |
| 11.1 | 6 种基(Based)指针 | (287) |
| 11.1.1 | 变量值基指针 | (287) |
| 11.1.2 | 变量地址基指针 | (289) |
| 11.1.3 | 不定基指针 | (289) |
| 11.1.4 | 段名基指针 | (290) |
| 11.1.5 | 指针基指针 | (291) |
| 11.1.6 | 自参照基指针 | (291) |
| 11.2 | 基指针应用于链表管理的工具包 | (292) |
| 11.2.1 | 基指针应用于链表管理 | (292) |

| | | |
|---------------|-------------------------------|--------------|
| 11.2.2 | 基指针分配函数 | (293) |
| 11.2.3 | 16 个工具函数 | (294) |
| 11.2.4 | 工具包应用三例 | (304) |
| 11.2.4.1 | [例 1] | (304) |
| 11.2.4.2 | [例 2]和[例 3] | (305) |
| 第 12 章 | 与 BIOS 和 DOS 的接口 | (309) |
| 12.1 | 中断概述 | (309) |
| 12.2 | 与 BIOS 的接口 | (312) |
| 12.2.1 | 与 BIOS 接口的函数 | (312) |
| 12.2.2 | BIOS 提供的部分服务 | (314) |
| 12.3 | 与 DOS 的接口 | (317) |
| 12.3.1 | 与 DOS 接口的函数 | (317) |
| 12.3.2 | DOS 提供的部分服务 | (318) |
| 12.4 | 标准输入 / 输出服务 | (319) |
| 12.4.1 | BIOS 提供的显示服务 | (319) |
| 12.4.2 | BIOS 提供的键盘服务 | (320) |
| 12.4.3 | DOS 提供的标准输入 / 输出服务 | (320) |
| 12.5 | 文件输入 / 输出服务 | (321) |
| 12.6 | 内存管理与程序执行服务 | (322) |
| 12.7 | 打印服务 | (323) |
| 12.8 | 时钟 / 日历服务 | (325) |
| 12.8.1 | PC 机上的时钟系统 | (325) |
| 12.8.2 | PC / AT 机上的时钟系统 | (326) |
| 12.8.3 | DOS 的时间 / 日历服务 | (326) |
| 12.8.4 | 延迟函数 | (326) |
| 12.8.5 | 声音函数 | (327) |
| 12.9 | 串行通信服务 | (327) |
| 12.10 | 错误处理服务 | (328) |
| 12.10.1 | DOS 怎样报告错误 | (328) |
| 12.10.2 | Turbo C 库函数的错误报告特性 | (329) |
| 12.10.3 | 致命错 | (331) |
| 12.10.4 | <Ctrl Break> 和 <Ctrl C> | (331) |
| 第 13 章 | 中断服务程序 | (333) |
| 13.1 | 一般概念 | (333) |
| 13.2 | 用 Turbo C Tools 写中断服务程序 | (335) |
| 13.2.1 | 工作原理 | (335) |
| 13.2.2 | 安装和驻留 | (339) |

| | | |
|---------------|-------------------------------|--------------|
| 13.2.3 | 过滤 | (342) |
| 13.2.4 | 探测和撤消 | (342) |
| 13.2.5 | 其它 | (343) |
| 13.3 | 中断服务程序实例 | (344) |
| 13.3.1 | [例 1]:周期性地发声 | (344) |
| 13.3.2 | [例 2]:检测 A 和 J 键的同时按下 | (345) |
| 13.3.3 | [例 3]:发送格式化的输出 | (349) |
| 13.4 | 用 Turbo C Tools 写插入服务程序 | (353) |
| 13.4.1 | DOS 的重入问题 | (353) |
| 13.4.2 | 插入服务技术 | (354) |
| 13.4.3 | 插入服务函数 | (358) |
| 13.4.4 | 插入服务程序举例 | (362) |
| 13.5 | 用 Turbo C 写中断服务程序 | (366) |
| 第 14 章 | 图形处理 | (369) |
| 14.1 | Turbo C 图形处理函数 | (369) |
| 14.1.1 | 概述 | (369) |
| 14.1.2 | 图形系统控制函数 | (369) |
| 14.1.3 | 画图和填充函数 | (373) |
| 14.1.4 | 屏幕和视口管理函数 | (375) |
| 14.1.5 | 图形方式下的文本输出函数 | (376) |
| 14.1.6 | 颜色控制函数 | (378) |
| 14.1.7 | 错误处理函数 | (380) |
| 14.1.8 | 状态查询函数 | (380) |
| 14.2 | Pop up 图形窗口工具包 | (381) |
| 14.2.1 | 图形窗口与文本窗口 | (381) |
| 14.2.2 | 6 个工具函数 | (382) |
| 14.2.3 | 工具包应用举例:移动窗口 | (389) |
| 14.3 | 图形方式下输出文本的若干问题 | (392) |
| 14.3.1 | 格式输出 | (392) |
| 14.3.2 | 重写 | (393) |
| 14.3.3 | 加亮 | (395) |
| 14.3.4 | 滚动 | (396) |
| 14.4 | 用 XOR 方式画旋转橡皮筋 | (396) |
| 第 15 章 | 混合模式和混合语言编程 | (400) |
| 15.1 | 混合模式编程 | (400) |
| 15.1.1 | 概述 | (400) |
| 15.1.2 | 说明一个函数为 near 或 far | (400) |

| | | |
|-------------|-------------------------------------|--------------|
| 15.1.3 | 说明一个指针为 near、far 或 huge | (401) |
| 15.1.4 | 使用库文件 | (402) |
| 15.1.5 | 不同编译模式所生成模块的连接 | (403) |
| 15.2 | C 和汇编语言混合编程 | (404) |
| 15.2.1 | 段的组合 | (404) |
| 15.2.1.1 | 汇编语言的段和组 | (404) |
| 15.2.1.2 | Trubo C 的段和组 | (405) |
| 15.2.1.3 | 段和组的连接 | (407) |
| 15.2.2 | 变量和函数名的相互引用 | (408) |
| 15.2.3 | 参数传递规则 | (409) |
| 15.2.4 | 返回值传递规则 | (410) |
| 15.2.5 | 寄存器规则 | (411) |
| 15.2.6 | 混合编程示例 | (411) |
| 15.2.6.1 | C 调用汇编 | (411) |
| 15.2.6.2 | 汇编调用 C | (412) |
| 15.3 | 行内汇编 | (414) |
| 附录 1 | 操作符表 | (417) |
| 附录 2 | 键盘码表 | (419) |
| 附录 3 | Turbo C 2.0 函数简表 | (422) |
| 附录 4 | Turbo C Tools 6.0 函数简表 | (446) |

第 1 章 几个重要问题

1.1 数据类型转换

1.1.1 各类整数之间的转换

C 语言中的数分 8 位、16 位和 32 位三种。属于 8 位数的有：带符号字符 char，无符号字符 unsigned char。属于 16 位数的有：带符号整数 int，无符号整数 unsigned int(或简称为 unsigned)，近指针。属于 32 位数的有：带符号长整数 long，无符号长整数 unsigned long，远指针。

IBM PC 是 16 位机，基本运算是 16 位的运算，所以，当 8 位数和 16 位数进行比较或其它运算时，都是首先把 8 位数转换成 16 位数。为了便于按 2 的补码法则进行运算，有符号 8 位数在转换为 16 位时是在左边添加 8 个符号位，无符号 8 位数则是在左边添加 8 个 0。当由 16 位转换成 8 位时，无论什么情况一律只是简单地截取低 8 位，抛掉高 8 位。没有 char 或 unsigned char 常数。字符常数，像 'C'，是转换为 int 以后存储的。当字符转换为其它 16 位数（如近指针）时，是首先把字符转换为 int，然后再进行转换。

16 位数与 32 位数之间的转换也遵守同样的规则。

注意，Turbo C 中的输入/输出函数对其参数中的 int 和 unsigned int 不加区分。例如，在 printf 函数中如果格式说明是 %d，则对这两种类型的参数一律按 2 的补码(即按有符号数)进行解释，然后以十进制形式输出。如果格式说明是 %u、%o、%x、%X，则对这两种类型的参数一律按二进制(即按无符号数)进行解释，然后以相应形式输出。在 scanf 函数中，仅当输入的字符串中含有负号时，才按 2 的补码对输入数进行解释。

还应注意，对于常数，如果不加 L，则 Turbo C 一般按 int 型处理。例如，语句 printf("%08lx", -1L)，则会输出 ffffffff。如果省略 l，则输出常数的低字，即 ffff。如果省略 L，则仍会去找 1 个双字，这个双字的低字就是 int 常数-1，高字内容是不确定的，输出效果将是在 4 个乱七八糟的字符之后再跟 ffff。

在 Turbo C 的头文件 value.h 中，相应于 3 个带符号数的最大值，定义了 3 个符号常数：

```
#define MAXSHORT 0X7FFF
#define MAXINT 0X7FFF
#define MAXLONG 0X7FFFFFFFL
```

在 Turbo C Tools 中，包括 3 对宏，分别把 8 位拆成高 4 位和低 4 位，把 16 位拆成高 8 位和低 8 位，把 32 位拆成高 16 位和低 16 位。

```
uthinyb(char value) utlonyb(char value)
uthibyte(int value) utlobyte(int value)
uthiword(long value) utloword(long value)
```

在 Turbo C Tools 中, 也包括相反的 3 个宏, 它们把两个 4 位组成一个 8 位, 把两个 8 位组成一个 16 位, 把两个 16 位组成一个 32 位。

```
utnybbyt (HiNyb, LoNyb)
utwdlong (HiWord, Loword)
utbyword (HiByte, LoByte)
```

1.1.2 实数与整数之间的转换

Turbo C 中提供了两种实数: float 和 double。float 由 32 位组成, 由高到低依次是: 1 个尾数符号位, 8 个偏码表示的指数位(偏值 = 127), 23 个尾数位。double 由 64 位组成, 由高到低依次是: 1 个尾数符号位, 11 个偏码表示的指数位(偏值 = 1023), 52 个尾数位。通过下列公式, 可以由尾数和指数计算出所代表的实数值:

$$X = \pm 1.尾数 * 2^{(指数-偏值)}$$

下列几种情况下, 此公式不成立:

指数 = 000...0 且尾数 = 00...0, 则 $X = \pm 0$

指数 = 000...0 且尾数! = 00...0, 则 $X = \pm 0.尾数 * 2^{(1-偏值)}$

指数 = 11...1 且尾数 = 00...0, 则 $X = \pm \infty$

指数 = 11...1 且尾数! = 00...0, 则 X 是一个无效数

在 Turbo C 的头文件 value.h 中, 相应于实数所能达到的最大最小值, 定义了如下几个符号常数:

```
#define MAXFLOAT      3.37E+38
#define MINFLOAT      8.43E-37
#define MAXDOUBLE     1.797693E+308
#define MINDOUBLE     2.225074E-308
```

实常数是按 double 格式存放的, 如果想按 float 格式存放, 则必须加后缀 F, 如: 1.23E+4F。

当把实数强制转换为整数时, 采取的是“向零靠拢的算法”, 如:

| | | |
|--------------|---------|----------|
| float值: | 65432.6 | -65432.6 |
| 转换为long: | 65432 | -65432 |
| 转换为unsigned: | 65432 | 104 |

如果不希望“向零靠拢”, 而希望“四舍五入”, 则必须由程序员自己处理。一种办法是先加上(符号位 / 2), 然后再进行类型转换。应该注意的是: 如果被转换的实数值超过了目标类型的表达范围, 则会产生错误。例如上面的 float 值 -65432.6 转换为 unsigned int 值时, 由于超过了目标范围, 所产生的 104 就是错误的。在 65432.6 转换为 unsigned int 的 65432 以后, 可以用 printf 的 %u 格式输出, 如果用 %d 格式输出, 则会得到错误的结果。

1.1.3 指针之间的转换

关于各类指针之间的转换请见下一节以及第 11 章。

1.2 指 针

1.2.1 指针说明

指针是包含另一变量的地址变量。它的一般说明形式，如 `int *fd`，其 `fd` 是一个指向整型变量的指针。比较复杂的指针说明，如 `*(* pfi) ()`，可按如下几个原则来理解：以标识符为中心，一对方括号一般表示数组，一对圆括号一般表示函数或强调某一优先顺序，方括号对和圆括号对为同一优先级，方括号和圆括号比 * 号优先级高。以下几例解释了这些原则的应用。

`int *fip()`，因圆括号优先级高，故 `fip` 先与圆括号结合，说明 `fip` 是一个函数，这个函数返回一个指向整数的指针。

`int (* pfi) ()`，因两对圆括号为同一优先级，故从左到右，`pfi` 是一个指针，这个指针指向一个函数，这个函数返回一个整数。

`int *par[]`，因方括号比 * 号优先级高，故 `par` 是一个数组，这个数组的每一个元素是指向整数的指针。

`int (* ptr) []`，因方括号与圆括号为同一优先级，故 `ptr` 是一个指针，这个指针指向一个数组，这个数组的每一个元素是一个整数。

`int *(* pfi) ()`，`pfi` 是一个指针，这个指针指向一个函数，这个函数返回一个指向整数的指针。

1.2.2 指针与地址

指针在使用之前必须初始化，给指针赋地址的方法一般有如下几种：

第一种很容易理解，通过取地址操作符取变量(包括结构变量)的地址，如：

```
char c="a",      * ptr_char;
ptr_char = &c;
```

第二种是数组，因为不带方括号的数组名等效于数组中第一个元素的地址，即数组名也可看作是一个指针，所以有两种办法。如：

```
char myname[31], * ptr;
ptr = myname;
或
ptr = &myname[0];
```

第三种是动态分配的一块内存，这时往往带有类型强制转换，但应注意当内存不够时，可能返回一个空(NULL)指针。如：