目　录

第一章　批处理文件与 DOS …… 1
1.1　什么叫批处理文件 …… 1
1.2　批处理文件的建立 …… 1
1.3　批处理文件的执行 …… 4
1.4　批处理文件的错误信息 …… 5
1.5　DOS 简介 …… 6
1.6　DOS 系统的装入 …… 6
第二章　文件系统与 DOS 指令 …… 8
2.1　文件与文件名 …… 8
2.2　树状结构的文件系统 …… 9
2.3　DOS 指令简介 …… 10
2.4　内部指令的操作 …… 13
2.5　临时指令的操作 …… 17
第三章　批处理指令 …… 22
3.1　批处理指令 …… 22
3.2　ECHO 指令 …… 23
3.3　注解(:)指令 …… 26
3.4　REM 指令 …… 28
3.5　PAUSE 指令 …… 28
3.6　GOTO 指令 …… 29
3.7　FOR 指令 …… 31
3.8　SHIFT 指令 …… 33
3.9　IF 指令 …… 36
3.10　CHOICE 指令 …… 42
3.11　批处理文件的链接 …… 44
第四章　批处理文件范例讨论 …… 48
4.1　前言 …… 48
4.2　范例讨论 …… 48
第五章　树状文件结构的指令用法 …… 60
5.1　树状文件结构 …… 60
5.2　子目录的指令用法 …… 62
5.3　树状结构的重建 …… 67
5.4　DOS 分区的建立 …… 69
5.5　DOS 分区的格式化 …… 70

5.6 硬盘的文件维护 …… 71
5.7 设计自己的文件系统 …… 75
第六章 数据的传输管道与过滤功能 …… 76
6.1 前言 …… 76
6.2 数据输入输出的改向 …… 76
6.3 数据的过滤功能 …… 78
6.4 数据管道 …… 86
6.5 DOS 5.0 和 DOS 6.0 的 DIR 指令 …… 89
6.6 DOS 的打印功能 …… 90
第七章 批处理文件的设计技巧 …… 93
7.1 前言 …… 93
7.2 批处理文件的程序结构 …… 93
7.3 程序的查错功能 …… 94
7.4 设备文件 NUL:的用途 …… 97
7.5 自动按键 …… 98
7.6 DOS 操作环境变量 …… 100
7.7 综合范例 …… 103
第八章 DOS 6.0 的新指令及内存管理 …… 107
8.1 消除疲劳的 DOSKEY 指令 …… 107
8.2 宏指令的定义 …… 109
8.3 安全可靠的保护措施 …… 114
8.4 内存的管理 …… 122
第九章 配置文件及启动文件 …… 129
9.1 定义 DOS 的配置文件 …… 129
9.2 DOS 6.0 新增的配置文件功能 …… 137
9.3 屏幕及键盘的扩充功能 …… 142
9.4 在 PROMPT 指令中使用 ANSI.SYS …… 145
9.5 ANSI.SYS 批处理文件范例 …… 147
9.6 启动文件的使用 …… 154
9.7 多配置文件与启动文件的配合 …… 156
第十章 工具程序 …… 158
10.1 前言 …… 158
10.2 工具程序的功能 …… 158
10.3 综合程序范例 …… 166
附录 A DOS 的指令索引及摘要 …… 169
附录 B EDLIN 指令的使用 …… 180
B.1 EDLIN 简介 …… 180
B.2 如何启动 EDLIN …… 180
B.3 EDLIN 的使用 …… 181

B.4 EDLIN 的命令摘要 …… 182
B.5 EDLIN 的错误信息 …… 187
附录 C EDIT 编辑程序的使用 …… 190
C.1 EDIT 编辑程序简介 …… 190
C.2 启动 EDIT 编辑程序 …… 190
C.3 EDIT 指令的操作 …… 194
C.4 EDIT 编辑目录 …… 196
C.5 Search 寻找目录 …… 198
C.6 存储和装入文件 …… 200
C.7 Options 和 Help 目录 …… 200
附录 D 美国国家标准数据码(ASCII CODE) …… 202
D.1 控制字符 …… 202
D.2 标准字符 …… 203
D.3 扩充字符 …… 204

第一章　批处理文件与 DOS

1.1　什么叫批处理文件

简单地说，批处理文件(Batch File)就是一个包含 DOS 指令而扩展名为 BAT 的文本文件。当你执行批处理文件时，DOS 就会依次执行该文件内所包含的指令，与在 DOS 提示符下自行输入各个指令所得到的结果完全相同。因此，你可以在批处理文件中加入一些常用的 DOS 指令，当你需要时只要执行这个批处理指令即可，不但可以得到相同的执行结果，而且可以避免重复输入指令而达到节省时间的目的。

批处理文件内除了可以包含任何的 DOS 指令外，还可以包括如下的文件或指令：

- 任何可以在 DOS 提示符下执行的程序文件
- 另外一个批处理文件
- DOS 所提供的批处理指令

因此，使用批处理文件除了可以节省时间外，还可以让你拓展 DOS 指令的功能，使它们更具有灵活性而能完全满足你的需要。

本书所要介绍的重点虽然是批处理文件的设计技巧，但是对于 DOS 指令的功能和用法也做了简单的介绍。因此，无论是 DOS 的初学者或是行家，本书所提供的范例都可以满足你的要求，并逐步引导你进入批处理文件的设计领域，使你成为使用 DOS 的高手。

1.2　批处理文件的建立

批处理文件本身就是一个文本文件，因此要建立批处理文件最简单的的方法，就是利用 DOS 所提供的 COPY 指令从屏幕上来建立。例如，要建立一个名为 TEST.BAT 的批处理文件，只要在 DOS 提示符下输入如下指令，然后按下 Enter 键即可：

```
COPY CON: TEST.BAT
```

然后依次输入所需的指令，接着按下 Enter 键，最后按功能键 F6 或同时按下 Ctrl 和 Z 键(用 Ctrl+Z 表示)，就可完成建立批处理文件的工作，例如下列的实际操作：

```
A:\>COPY CON: TEST.BAT
CLS
DATE
TIME
^Z
        1 file(s) copied

A:\>
```

利用 COPY 指令来建立批处理文件是最容易的建立方式，但缺点是当发现文件的内容有错误时，就必须重新建立文件。因此，这种建立文件的方式比较适合小而简单的批处理文件。如果要建立较大的批处理文件，最好利用 DOS 的行编辑程序 EDLIN 或其他文本编辑程序（例如 PE、WORDSTAR 或 WORDPERFECT 等），来建立文件会更加方便。但无论你用哪种方式建立文件，都要建立一个以 BAT 为扩展名的文件名，然后逐行输入各指令即可。

由于本书所提供的批处理文件范例大都是利用行编辑程序 EDLIN 建立的，因此特别介绍了这个指令的使用方法，以便你建立所需的批处理文件。当你要利用 EDLIN 指令来建立批处理文件时，只要在输入 EDLIN 指令后接着输入所要的文件名即可，例如下面的操作：

```
A:\>EDLIN GOOD.BAT
New file
*I ----------输入字母 I 就可开始逐行输入指令
     1:*CLS
     2:*DATA
     3:*TIME
     4:*^C ----------按 Ctrl+C 键即可结束
*E ----------输入字母 E 就可回到 DOS 提示符

A:\>
```

文件建立完成后，你就可以利用 TYPE 指令来显示所建立的批处理文件内容，例如下列的操作：

```
A:\>TYPE GOOD.BAT
CLS
DATE
TIME

A:\>
```

如果你发现批处理文件的内容不正确，或是想修改原来的内容，则可以再利用 EDLIN 指令来编辑源文件的内容，例如下列的操作就是用来修改 GOOD.BAT 文件内的 DATA 指令，并加入新的指令：

```
A:\>EDLIN GOOD.BAT
End of input file
*L ----------输入字母 L 就可显示文件内容
     1:*CLS
     2:DATA
     3:TIME
*2 ----------输入数字 2 表示要修改第二行的指令
     2:*DATA
     2:*DATE----------重新输入新的指令
*#I ----------输入 #I 表示要加入新的指令到最后一行上
     4:*DIR *.BAT
```

```
        5:*^C
*E

A:\>
```

当你再利用 TYPE 指令来显示文件内容时，就可显示如下修改过的内容：

```
A:\>TYPE GOOD.BAT
CLS
DATE
TIME
DIR *.BAT

A:\>
```

EDLIN 指令是一个非常简单而且方便的建立批处理文件的工具，总结上面的操作，EDLIN 指令所提供的编辑指令简单说明如下：

指令	功能
A	增添文字行
C	复制文字行
D	删除文字行
E	终止 EDLIN 并存储文件
I	插入文字行
L	列出文字行
M	搬移文字行
P	一次显示一个屏幕的文字行
Q	终止 EDLIN 但不存储文件
R	以一个字符串取代另一个字符串
S	搜寻一个字符串
T	从磁盘拷贝一个文件内容
W	写入文字行
line num	编辑文字行

上述的 E 和 Q 指令的差别是：E 指令会将所输入或修改的指令存储到磁盘上，如果磁盘上已含有你所指定的批处理文件，那么原来的文件内容将会变成新的内容；而 Q 指令则会放弃你所做的一切修改，也就是不会影响原来的文件内容。其他指令的详细用法请参考附录 B 的说明。

如果你使用的是 DOS 5.0 或最新的 6.0 版本，那就可以利用它所提供的 EDIT 编辑程序来进行批处理文件的编辑工作。这比前面所介绍的 COPY 指令或 EDLIN 指令来建立批处理文件要方便得多。详细的 EDIT 指令用法请参考附录 C 的说明。

1.3 批处理文件的执行

批处理文件建立完成之后，这个批处理文件好像是一个新的 DOS 指令，因此，当要执行批处理文件时，只要输入文件名即可，而不必输入 BAT 的扩展名。例如要执行上面所建立的 GOOD.BAT 批处理文件，输入如下指令即可：

```
A:\>GOOD
```

然后按下 Enter 键，DOS 就会依次执行批处理文件内的指令，并显示如下的执行结果：

```
A:\>DATE
Current date is Thu 07-02-1992
Enter new date (mm -dd--yy):        ------按 Enter 键

A:\>TIME
Current time is 7:05:09.32a
Enter new time: ---      ------      -按 Enter 键

A:\>DIR *.BAT

Volume in drive A is BATCH DISK
Directory of A:\

HELP     BAT     1157   01-01-80   12:04a
TEST     BAT       17   07-02-92    6:52a
GOOD     BAT       29   07-02-92    7:00a
       3 file(s)      1203 bytes
                    108544 bytes free

A:\>
A:\>
```

从上面的显示结果可以看出，这个执行结果与你在 DOS 提示符下自行输入如下的指令，除了所显示的时间不同外，其余都是相同的。

```
CLS
DATE
TIME
DIR *.BAT
```

其中 CLS 指令是用来清除屏幕，DATE 指令是用来显示当前的日期，TIME 指令是用来显示当前的时间，而 DIR 指令则是用来显示存储在磁盘上的批处理文件名称。

当你执行批处理文件时，DOS 将会逐一执行文件中的各个指令然后自动停止，但也可以按 Ctrl+Break 或 Ctrl+C 键中途取消批处理文件的执行，当你按下 Ctrl+Break 键时 DOS 会显示如下信息来提醒你：

Terminate batch job(Y/N)?

此时你可以回答"Y"或"N",但不能只按 Enter 键。如果你回答"N",则正在执行的指令将被取消,但 DOS 会继续执行批处理文件中的下一个指令;如果回答"Y",DOS 便会取消整个批处理指令而回到 DOS 指示符。注意:如果你是从批处理文件中执行一个程序,那就必须等到这个程序执行完毕之后才能终止批处理文件的执行。

1.4　批处理文件的错误信息

在你执行批处理文件时,除了会显示错误信息来提醒你不正确的 DOS 指令外,还会显示一些与批处理文件有关的错误信息供你参考,以便查错。这些错误信息包括:

Batch file missing

DOS 无法找到你所要执行的批处理文件。检查一下你所输入的文件名是否正确,或当前目录中是否含有这个批处理文件。

Cannot start COMMAND,existing

当 DOS 试图把另外一个指令处理程序装入内存中时,在配置文件中的 FILE 参数值太小,或是内存中没有足够的空间来存储 COMMAND.COM 文件。你可以重新装入 DOS,或是增加配置文件中的 FILE 参数,然后重新装入 DOS。

FOR cannot be nested

在批处理文件中使用了嵌套的 FOR 指令,即 FOR 指令行中又含有另外一个 FOR 指令。

Label not found

在批处理文件中,GOTO 指令所指定的标记不存在。检查一下标记名称是否正确,或在文件中加入一个适当的标记。

No free file handles/cannot start COMMAND,existing

DOS 试图装入另外一个指令处理程序到内存中时,因已打开太多的文件而无法装入。你可以增加配置文件中的 FILE 参数,然后重新装入 DOS。

Out of environment space

DOS 用来存储操作环境变量的内存空间已经不足。

Syntax error

在批处理文件中使用了不正确的 DOS 指令。检查一下你所使用的 DOS 指令语法是否正确。

1.5 DOS 简介

DOS 是磁盘操作系统(Disk Operating System)的简称,它是一个很大的程序,是沟通 IBM PC 主机与驱动器的桥梁,使数据能在两者之间正确的互相传递。因此,想要使用驱动器做正确的数据存取工作,DOS 是一种必备的操作工具。DOS 除了沟通主机与驱动器外,也是沟通用户与计算机间的桥梁,通过 DOS 用户可以简单地操作计算机,让用户在最短的时间内完成所需的工作。总之,对于想要操作 IBM PC 计算机的用户,DOS 是你首先要熟悉的操作系统。

DOS 最初是由 Microsoft 软件公司开发出来的操作系统,使用于以 CPU 8086 或 8088 为基础的个人计算机上,通过 DOS 你可以非常方便地来操作 IBM PC 主机、驱动器及打印机等计算机硬件设备。自 DOS 问世以来,已经过多次的修改而有六个主要的版本(Version),即 1.xx、2.xx、3.xx、4.xx、5.xx 或 6.xx,其中 xx 表示新版本发表之后所做的修改次数,例如 3.3 或 4.01。每次使用 DOS 系统时,它都会显示当前你所使用的版本号。修订 DOS 主要是为了增加更多的功能,以充分利用更先进的硬件优点,并修改一些先前版本的错误(Bugs)。虽然新版的功能比旧版强,但其间的改良是渐进的,因为各个版本都是兼容的(Compatible),所以用户可轻易地适应和熟悉新的 DOS。

DOS 6.0 是目前最新的版本,不但可使用于以 80286 为 CPU 的 IBM PC/AT 上,也适用于以 80386 甚至是以 80486 为 CPU 的 IBM/PS2 更高级机型上,在速度、效率及功能上提高了很多,其中有一个重大的改进,就是提供了目前最流行的多窗口操作菜单,使初学者再也不必地去熟记任何指令,只需通过键盘和鼠标的指引就能轻易地控制整个操作系统。DOS 6.0 可以说是目前唯一能在 IBM PC 各型机种上使用的版本,加上友好而简单的操作方式,DOS 6.0 将成为以后最普及的版本。

DOS 系统主盘(System Master Diskette)是由 Microsoft 公司所提供,我们以上介绍的 DOS 6.0 就包含在这些磁盘上,利用其中的 DOS 指令(Command),我们就能轻易地命令计算机做我们想做的事以及数据存取的工作。除了 DOS 6.0 以外,系统主盘还包含了许多应用程序和实用程序,以供用户做更有效的利用,本书稍后也会做详细的介绍。

DOS 6.0 系统磁盘共有五张,其中标签为 Disk1、Disk2 及 Disk3 等三张磁盘是作为安装 DOS 6.0,或更新 DOS 旧版本用,而标签为 Wordgrp1 的磁盘是专门安装网络功能用,另外标签为 Suppdisk1 的磁盘则提供其他的支持功能。

要完整地使用 DOS 6.0 所提供的功能,你的主机必须配备硬盘,而且至少要有 2.5MB 的剩余空间可用。DOS 6.0 系统安装的过程非常简单,只要输入 SETUP 指令按照菜单上所显示的步骤,依次进行就可以完成安装的步骤。如果你是计算机初学者,则需通过计算机经销商为你进行或指导安装的步骤。

1.6 DOS 系统的装入

要使用 DOS 的指令,必须先将 DOS 系统装入计算机内存中,这个装入的过程称为 Booting DOS。如果你要从软盘驱动器装入 DOS 系统,只要在打开主机电源前,将具有启动 DOS 的系统磁盘插入驱动器 A 中并关好门栓,随着电源的接通,驱动器的指示灯会亮起,主机自动做

装入 DOS 的工作，在几秒钟内屏幕就出现提示的文字。

如果你是从硬盘驱动器来装入 DOS 系统，直接打开主机电源就可自动地装入 DOS 并出现提示的文字。但是要注意，开机时软盘驱动器的门栓要开着，以免主机又从软盘驱动器装入 DOS。

如主机已处于开机状态而要装入 DOS 时，则可执行系统复位(System Reset)，即同时按 Ctrl、Alt、Del 三个键，此时驱动器指示灯会亮起，装入操作开始，最后屏幕也会出现提示文字。

总之，装入 DOS 的方法有两种，采用前者的方法即执行主机由关到开的冷启动或采用后者系统复位的热启动。但无论是冷启动或热启动屏幕上都会显示如下的提示文字：

```
A)      (从软盘驱动器装入 DOS 系统)
C)      (从硬盘驱动器装入 DOS 系统)
```

如果你的屏幕没有直接显示 A)或 C)符号，而是显示要你输入日期和时间的信息，那么你只要按两次 Enter 键即可。

A)称为 DOS 系统的提示符(Prompt)，它表示 IBM PC 主机此时已进入 DOS 系统状态，而且一切准备就绪可以接受任何 DOS 指令了。A)的 A 为系统预设的驱动器 A，即启动主机时装入 DOS 的第一台驱动器。如果想指定第二台驱动器则可用 B:来选择，用户如装有硬盘驱动器则可以 C:来选择到硬盘驱动器上，如下所列：

```
A)              (表示存取操作在当前的驱动器 A 上)
A)B:(Enter)     (表示输入 B:之后再按 Enter 键)
B)              (表示存取操作选定在当前的驱动器 B 上)
A)C:(Enter)     (表示输入 C:之后再按 Enter 键)
C)              (表示存取操作选定在当前的驱动器 C 上)
```

如果直接从硬盘驱动器装入 DOS 系统，则直接显示 C)提示符，此时如要指定驱动器 A 或 B，则可以 A:或 B:来选择即可。当你操作完毕后，只要关掉主机电源，然后关掉显示器和打印机等外围设备的电源即可。

学习计算机最有效的方式，就是多实践上机操作。一边看书，一边操作，再观察计算机执行的结果，比始终埋头书本，光看不练，印象要深刻得多。本书所提供的批处理文件范例，都值得你实机上机练习，以提高学习的效率。这些范例虽然都是在 DOS 6.0 版本上编写的，但是只要使用 DOS 3.1 以上的任何版本都可正确地执行。如果范例中使用到的指令需要特别的版本才能执行时，将会加以注明。

第二章 文件系统与 DOS 指令

2.1 文件与文件名

文件是指一组相关信息的集合。一张磁盘上的文件可以比喻为抽屉中的文件夹，用来存储所有的程序、文字或数据。而每一个文件都被赋予一个单独的名称，以便借助文件的名称来查阅各个文件的内容。典型的 DOS 文件名其格式如下：

XXXXXXXX. XXX

文件名通常包括两部分：文件名和扩展名(其间以句点隔开)。文件名可由 1 到 8 个字符组成，扩展名则由 3 个或少于 3 个的字符组成，但扩展名为选择性的，可有可无。除了文件名和扩展名外，也可以指定文件所在磁盘的驱动器位置，只要在文件名前加上驱动器设备名称即可。如不指定的话，DOS 就当作是在“当前所选定的驱动器”上。因此一个完整文件名称包括三部分：驱动器的设备名称、文件名及扩展名。

文件名可以用大写或小写字母来表示，DOS 都会将其转换成大写字母的形式。除了下列的符号不能使用外，其余都是组成文件名和扩展名的正确字符：

. " / \ [ ] :

| < > + = ; ,

ASCII 码小于 33 的控制字符

下列的文件名都是有效的：

A:GOOD. BAT	参考驱动器 A 内的 GOOD. BAT 文件。
TEMP. BAS	参考当前驱动器，也就是当前所选定驱动器内磁盘的 TEMP. BAS 文件。
B:MY. NEW	参考驱动器 B 内的 MY. NEW 文件。
TEST	参考当前磁盘的 TEST 文件。

文件名结构中也可以利用通配符(Wild Card)"?"或"*"来参考一个以上的文件，也就是当搜寻磁盘文件目录来找出相同的文件名时，"?"字符可以取代相关位置上的任何字符，而"*"则可代表文件名的任何字符。例如：

B:TEMP. ??? 或 B:TEMP. *

可用来参考驱动器 B 上所有以 TEMP 为文件名的文件，下面还有一些例子来说明通配符的用法：

A:*.COM	参考驱动器 A 内所有扩展名为 COM 的文件。
B:*.*	参考驱动器 B 内所有的文件。
B:????????. ???	与 B:*.* 相同，其实"*"相当于一串"?"。
DUMP. *	参考当前磁盘内所有以 DUMP 为名称的文件。
C*.*	参考当前磁盘内所有以字母 C 开头的文件。

DISK.　　参考当前磁盘内所有以 DISK 字母结尾的文件。

文件的扩展名虽然可有可无，但是当你配合通配符来使用时可以让你很容易地找出相同性质的文件。例如：

*.BAS

就是指全部以 BASIC 语言所编写的程序文件。有一些特定的扩展名是保留给 DOS 使用的，例如下列的扩展名与其文件性质：

COM —— 指令文件
EXE —— 指令文件
SYS —— 系统文件
BAT —— 批处理文件
BAS —— BASIC 程序文件
CPI —— 特殊设备文件
PIF —— 特殊设备文件

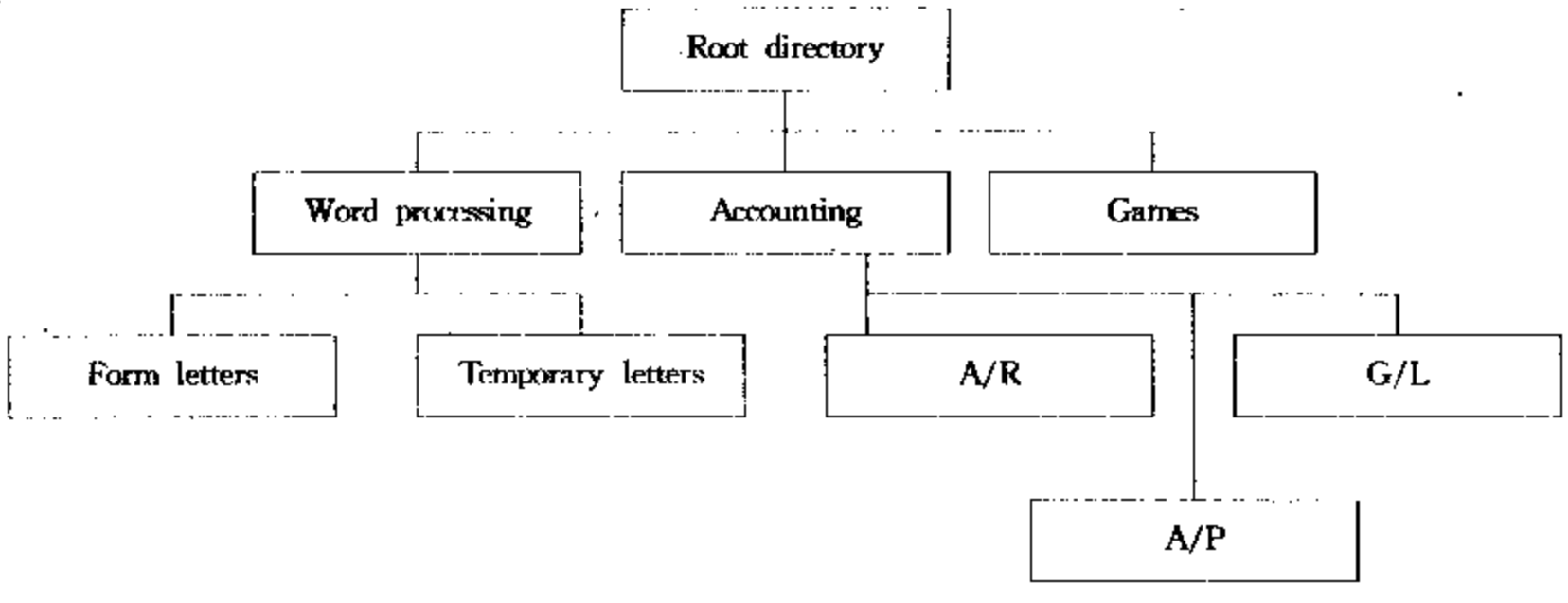
就像文件有名称一样，IBM PC 的计算机外围设备也是有名称的。在 DOS 指令中使用设备名称就好像使用文件名一样，但是这些设备名称都是由 DOS 赋予的，你并不能为设备命名，下表为组成典型系统所需的设备及 DOS 赋予它们的名称：

名称	设备
LPT1:(PRN:)	第一部打印机
LPT2:	第二部打印机
LPT3:	第三部打印机
CON:	显示器或键盘
COM1:(AUX:)	第一组通信口(Communication Port)
COM2:	第二组通信口
COM3:	第三组通信口
COM4:	第四组通信口
NUL:	不存在的设备(测试用)

上述这些名称为 DOS 所保留下来作为特殊用途，所以你就不能用这些 DOS 的设备名称来作为文件名，否则会造成错误。CON:及 PRN:是我们最常用的输入及输出设备，因此稍后在介绍 DOS 指令时会详细说明其用途。

2.2　树状结构的文件系统

磁盘文件的目录像一本书的目录，你可以从目录中检查磁盘的内容(也就是它所存储的文件名、各个文件所占的空间、建立文件的日期和时间等信息)。虽然任何一张磁盘都至少含有一个目录，但也能含有几个目录，而一个目录则包括一些相关的文件。当一张磁盘被设定以后，DOS 会自动在磁盘上建立一个主目录，这个目录也称为根目录。你可以在主目录下另外建立额外的子目录来存储你所需的文件，而形成树状结构，像一棵倒立的树，如下图：



你也可以在子目录下再建立另外一个子目录，只要磁盘的空间足够，DOS 并不限制你能建立的子目录层数。例如上图的根目录(Root Directory)含有三个子目录分别为：

WORDPROC —— 有关文本处理(Word Processing)的目录

ACCOUNTS —— 有关会计(Accounting)的目录

GAMES —— 有关计算机游戏(Games)的目录

而 WORDPROC 子目录又含有 FORMLET 和 TEMP 两个子目录，分别用来存储有关信件(Form Letters)和临时信件(Temporary Letters)的文件。另外 ACCOUNTS 子目录包含了 AR、AP 和 GL 三个子目录，分别用来存储有关应收帐款(Accounts Receivable)、应付帐款(Accounts Payable)和总帐(General Ledger)的文件。但 GAMES 子目录不含任何的子目录。

使用树状结构文件系统的最大好处是可以根据文件的性质来分门别类，以便存储在不同的子目录上，而把每一个子目录视为独立的单位来使用，使得磁盘文件的管理更加方便。

树状结构中的每一个子目录都有它单独的路径(Path)。因此当你要寻找某一个文件时，一定要指出该文件所属子目录的路径名称(Path Name)，DOS 才能依此路径去找出你所要的文件。例如上图中的子目录 GL，其路径从根目录开始依次为 ACCOUNT 然后为 GL，那么它的路名称则是：

\ACCOUNT\GL

第一个"\"表示根目录，后面的"\"则是用来分隔子目录和文件名用的。

磁盘文件的路径名称的使用格式为：

[d:]\ path\filename

如果上图的树状结构是存储于驱动器 A 中的磁盘文件系统，则子目录 AP 的路径名称可表示如下：

A:\ACCOUNT\AP

如果没有指定驱动器的代号，那么 DOS 就当作是当前所选定的驱动器。

树状结构文件系统的操作指令，请参考第五章的详细说明。

2.3 DOS 指令简介

DOS 指令一般都非常简单，可分为两种类型：内部指令(Built—in Command)及临时指令

(Transient Command)。内部指令就是随着 DOS 的装入而存在于计算机内存中，要使用时不必再从 DOS 的系统主磁盘中装入。而临时指令则相反，它是要使用时才到磁盘上去将它寻找出来，然后再装入内存中。

DOS 操作系统内用来解释指令的部分称为指令处理程序(Command Processor)，当你装入 DOS 后，指令处理程序就从系统主磁盘上的 COMMAND.COM 文件内被装入计算机内存中，以便处理用户所选用的指令。如果输入的是内部指令，DOS 的指令处理程序就会直接执行该指令；如果输入的临时指令，那么 DOS 的指令处理程序就会到当前磁盘上寻找该指令以便执行它，如果找不到该指令，DOS 就会显示如下的错误信息：

Bad command or filename

来提醒用户注意。因此，在你执行 DOS 的临时指令时，当前所使用的磁盘上一定要含有这些指令文件才能正常的执行。另外，你所输入的 DOS 指令不但要正确而且语法(Syntax)也要完全符合规定才能被接受。

一般 DOS 指令的使用语法都以如下的几项规则来说明其格式：

1. 以大写表示的指令必须完全正确的输入，当然输入时以大小写的方式都可接受。

2. 以中括号([ ])括住的部分是选择性的，也就是可有可无，但选择使用时中括号本身不需输入。

3. 格式中所出现的标点符号，例如，．＝？都必须正确的输入，除非是在中括号内的标点符号，那些是选择性的。

4. 以小写表示的部分则是可以由用户替换的，例如：

d：	表示驱动器的代号
filename	表示文件名
ext	表示扩展名

5. 符号"|"则表示可任选其中一种使用。

6. 符号"..."则表示可按用户所需的次数重复使用。

例如 DIR 指令的使用语法为：

DIR [d:] [filename [.ext]] [/P]

用户选择使用如下的任意一种方式都是正确的：

```
DIR
dir
DIR A:
DIR MYFILE
DIR B:TEST.BAS
DIR *.BAS
DIR /P
DIR C:*.BAS/P
```

但是 DIR.BAS 就错了。ECHO 指令的使用语法为：

ECHO [ON|OFF]

因此，用户选择使用如下的任意一种方式都是正确的：

```
ECHO
```

```
ECHO ON
ECHO OFF
```

用户在输入 DOS 指令后就可按 Enter 键以便执行。如在按 Enter 键之前发现所输入的指令格式有错误，则可以按倒退键 BackSpace 退回去修改或按 ESCAPE 键删除整个指令行后重新输入，假如是在按下 Enter 键后才发现错误，也可以按 Ctrl+Break 或 Ctrl+C 键来停止指令的执行。

如输入的指令不正确，DOS 在执行时就会显示错误的信息来提醒用户，为了用户的方便，DOS 5.0 和 6.0 各个指令的使用格式都摘录在附录 A 中以便参考。当然遇到下列的情况：像驱动器的门未关、打印机的电源未开或纸已用完、磁盘损坏等等，DOS 都会检查这些错误而显示不同的错误信息来提醒用户注意，甚至会以如下的文字来询问用户要采取哪一种步骤：

Abort, Ignore, Retry, Fail?

如果你知道问题的所在，解决之后即可按 R 请 DOS 再试一次(Retry)；如果不是这种情况那就按 A 来取消指令的执行(Abort)。如果你按 I 那么 DOS 就会忽略(Ignore)这个错误而继续执行。如果你是按 F，那么 DOS 就会取消当前的指令并送出一个错误信息，然后继续执行这个指令，这时候你可以输入正在使用的驱动器代号(像 A: 或 C:)即可。

要注意的是：回答时按 A、I、R 或 F 即可，不必按 Enter 键。另外，如果这个信息是在使用驱动器时出现的，那么在回答之前千万不要更换磁盘，以免磁盘受损。

操作 DOS 时，最常遇到的错误信息有：

General Failure

造成的原因可能是磁盘与驱动器的类型不符，磁盘没有正确地插入驱动器中或是门栓没关好，或是磁盘没有正常地格式化。

Insufficient Disk Space

磁盘空间不足。

Invalid Drive Specification

所指定的驱动器代号不正确。

Nonsystem Disk or Disk Error

装入 DOS 时，显示你所使用的磁盘并非系统主磁盘或磁盘本身有问题。

Sector Not Found

表示在磁盘上无法找到存储数据的扇区，这意味着你的磁盘可能已经损坏而必须进行格式化了。

Seek Error

表示驱动器无法定位到磁盘中的正确磁道上，请确定磁盘是否正确地插入驱动器中，或者用另一部驱动器试试。

其他与各个指令有关的错误信息，请参考各章中介绍该指令时的说明。

2.4　内部指令的操作

1. CLS

用来清除屏幕，使提示符显示在屏幕的左上方。

2. DIR

用来显示磁盘上的文件目录，内容包括文件名、文件长度、建立文件日期和时间。例如下列的 DIR 指令：

DIR	显示当前磁盘内所有的文件名。
DIR B:	显示驱动器 B: 内所有的文件名。
DIR A:*.BAS	显示驱动器 A 内所有扩展名为 BAS 的文件名。
DIR A:A*.*	显示驱动器 A 内所有以 A 开头的文件名。
DIR /W	选择"宽"显示，即每行只包含五个文件名，其他信息则不显示出来。
DIR /P	选择"页"显示，即屏幕每显示满一屏就停止，如要继续则按下任何键即可。

3. TYPE

用来显示所指定文件的内容。此指令一般都用来显示以 ASCII 格式存储的文本文件内容。例如：

TYPE TEXT.DAT

可将 TEXT.DAT 的内容显示在屏幕上。如指定文件的内容并非 ASCII 文本文件时，屏幕上会出现一些乱七八糟的字符。

4. COPY

用来拷贝文件，例如：

COPY MYFILE.BAS YOURFILE.BAS

将 MYFILE.BAS 拷贝到同一磁盘上面文件名为 YOURFILE.BAS 的文件中，如 YOURFILE.BAS 文件已存在，则原文件内容将被 MYFILE.BAS 覆盖掉。注意，拷贝一个文件成其自身是不允许的。

COPY A:MYFILE.BAS B:/V

将 MYFILE.BAS 由驱动器 A 拷贝到驱动器 B 上面文件名仍然相同。使用选项/V 的目的是用来检查拷贝过程中是否有任何的错误发生。虽然这个检查的工作需要额外的时间，但在拷贝重要的文件时这是值得的。

COPY A.BAS+B.BAS+C.BAS TOTAL.BAS

将各个文件连接以后拷贝到同一驱动器上文件名为 TOTAL.BAS 的文件中，如未指明目标文件名则自动以第一个被连接的文件名 A.BAS 为文件名。

COPY A:*.BAS B:

将驱动器 A 上的所有以 BAS 为扩展名的文件名拷贝到驱动器 B 上，而各个文件名都与原来的相同。注意，目标磁盘一定要有足够的空间接受拷贝的文件，否则造成"Insufficient disk space"(磁盘空间不足)的错误。

COPY A:*.* B:

将驱动器 A 上所有的文件全部拷贝到 B 驱动器上，而各个文件名都与原来相同。

COPY 指令除了上述的文件拷贝功能外，还具备下列功能：

● 建立文本文件

COPY 指令可以让你用键盘来建立一文本文件(ASCII File)，例如输入下列指令行后：

COPY CON: TEST.BAT

DOS 将会等待你从键盘输入数据，每输入一行后按 Enter 键即可继续输入下一行，当全部输入完成后按 Ctrl+Z 键，DOS 就会将你输入的数据拷贝到指定的 TEST.BAT 文件内。如果磁盘内已含有 TEST.BAT 文件，那么文件内原来存储的数据就被"覆盖"。

利用这种方式虽然可以建立你所要的文本文件，但有一个问题：如果你要修改文件的部分内容时必须重新输入整个文件的数据，因此在建立较大的数据文件时，还是利用 DOS 所提供的 EDLIN 指令或 EDIT 指令来进行编辑的工作比较理想。

● 显示文件内容

COPY 指令可用来显示一个文本文件的内容到屏幕上或打印机上，这与 TYPE 指令的功能完全一样。例如：

COPY TEST.BAT CON:

可以将批处理文件 TEST.BAT 的内容显示在屏幕上，而如下的指令：

COPY TEST.BAT PRN:

则直接将文件内容输出到打印机上。

● 沟通键盘与打印机

COPY 指令可以将键盘所输入的数据送到打印机上打印。例如：

COPY CON:PRN:

以这种方式输入的数据并不会存储到文件中，当你全部输入完，按 Ctrl+Z 键后，所有的数据就被送到打印机上打印。利用这种方式来打印数据虽然很容易，却不是最好的办法，如发现数据有误时一定要全部重打，因此还是建议你先将数据以文本编辑工具如 EDLIN 来建立文件，当确定无误后再将它输出到打印机上。

5. REN(或 RENAME)

用来重新命名一个文件，例如：

REN OLD.BAS NEW.BAS

将 OLD.BAS 改名为 NEW.BAS。

REN *.BAK *.BAS

则会将所有以 BAK 为扩展名的文件名全部更改为以 BAS 为扩展名的文件名。请注意，若企图

将一文件名改为已存在目录中的名称时，将造成错误。

6. DEL(或 ERASE)

用来删除目录中所指定的文件，例如：

DEL A:TEMP.BAS	删除驱动器 A 中的 TEMP.BAS 文件。
DEL B:*.BAK	删除驱动器 B 所有以 BAK 为扩展名的文件。
DEL *.*	删除当前磁盘中所有的文件，此时为了慎重起见，系统会显示"Are you sure?"的提示，如确定要清除磁盘上所有的文件，按 Y，如不想清除按 N 即可。
DEL *.EXE/P	使用选项/P 可以在删除文件之前，先给你一个提示信息以确定是否删除。

文件一经删除将永远消失，所以使用此指令时要特别小心，以免造成损失。

7. DATE

用来设定系统日期，如果未包括任何的日期参数(也就是只下达 DATE 指令时)，DOS 就会显示系统当前所设定的日期，并提示你输入新的日期。和装入 DOS 时的情况一样，DOS 会先显示当前的日期(即先前所设定的日期)，然后提示你输入新的日期。如果你直接按 Enter 键，DOS 将维持原来日期。如果输入了新的日期，DOS 会把系统日期设定为新的日期。

8. TIME

用来设定系统时间，如果未包括任何的时间参数(也就是只下达 TIME 指令时)，DOS 就会显示系统当前所设定的时间，并提示你输入新的时间。和 DATE 指令一样，TIME 指令也可以让你检查和更改时间。日期和时间一经设定后只要不关闭电源或做系统复位，该日期和时间就会持续的更新。假如你的主机安装了以电池为电源的计时设备，那就不受关机或开机的影响，计算机会一直更新下去，不必再设定日期和时间，除非日期和时间有误时，才必须重新设定。

日期和时间的输入格式说明如下：

日期格式：mm－dd－yy 或 mm/dd/yy(月－日－年)

mm 代表月份，范围为 1～12。

dd 代表日期，范围为 1～31。

yy 代表年代，范围为 80～99 或 1980～2099 都可。

例如：6－1－86 或 06/01/86 都可被接受。

时间格式：hh:mm:ss.xx(时:分:秒)

hh 代表小时，范围为 0～23。

mm 代表分钟，范围为 0～59。

ss 代表秒，范围为 0～59。

xx 代表百分之一秒，范围为 0～59。

例如：9:05 或 13:30 或 10:10:10.10 都可被接受。

无论是在输入日期或时间时，如输入格式错误系统会提示你再一次地输入，直到正确或直

接按 Enter 键为止，DOS 就是利用这个日期和时间来记录最近一次你在系统上建立或更新一个文件的时间。

9. VER

用来显示当前正在使用的 DOS 版本号(Version)。

10. VOL

用来显示磁盘名称(Volume Label)。所谓磁盘名称是指磁盘在格式化(Formatting)时为磁盘所设定的名称，其目的只是用来辨认磁盘而已。例如：

VOL　　显示当前磁盘的磁盘名称。

VOL B:　显示驱动器 B 内的磁盘名称。

其实，DIR 指令、CHKDSK 指令及 TREE 指令也可用来显示磁盘名称。

11. VERIFY

用来确认数据是否被正确地存储到磁盘上。通常 DOS 对于每一个向磁盘输出的操作，都会自动地检验其正确性，因为这个检验的工作需要额外的时间，所以当你的程序要把数据存入磁盘时，整个系统执行的速度就会比较慢。如果不必确认时可以利用 VERIFY OFF 来停止确认的工作，以节省时间，需要时可再利用 VERIFY ON 来确认。

12. PROMPT

DOS 的提示符原设定为〉符号，PROMPT 指令可以用来改变它。其使用格式为：

PROMPT prompt

上列格式中的 prompt 提示符可从下表任选一个或组合几个使用：

代码	提示符
$$	钱字号($)
$b	符号|
$d	系统日期
$e	符号←
$g	符号〉
$h	自动按倒退键
$l	符号〈
$n	当前磁盘代号
$p	当前目录的路径
$q	符号=
$t	系统时间
$v	DOS 版本号
$	自动按 Enter 键

上表中的提示符都是以钱字号为首，而各代码的字母可以是大写或小写。最常用的一种提示符为显示当前目录的路径并紧跟着符号〉，例如：

PROMPT PG

而下列 PROMPT 指令则可分别改变提示符如下：

PROMPT d_t_pg	显示日期、时间及当前目录的路径并紧跟着符号〉
PROMPT thhhhhhg	只显示时间的时和分并紧跟着符号〉
PROMPT #	改变提示符为类似 UNIX 操作系统的 # 符号
PROMPT .	改变提示符为类似 dBASE 的点提示符
PROMPT SYC$g	显示作者的英文姓名字首并紧跟着符号〉

上列的 PROMPT thhhhhhg 指令，其中的六个 $h 是用来删除时间的秒数，因为整个时间的显示格式为：

hh:mm:ss.xx

为了要删除格式中的秒数(即":ss.xx"部分)，一共是六个字符，因此使用了六个 $h 来删除这六个字符，而保留时和分的部分。

2.5　临时指令的操作

再次提醒你，在操作如下的临时指令时一定要确定当前磁盘上含有这些指令，才能正常操作。

1. FORMAT

用来格式化指定驱动器内的磁盘，使之具有适当的格式以便存放 DOS 文件。在使用新磁盘前必须先进行格式化，否则不能使用。如果在进行磁盘复制工作时磁盘未先 FORMAT，则 DISKCOPY 指令会先将磁盘格式化，然后再拷贝。FORMAT 指令的使用格式如下：

FORMAT A:	将驱动器 A 内的磁盘格式化。
FORMAT B:	将驱动器 B 内的磁盘格式化。
FORMAT B:/V	将驱动器 B 内的磁盘格式化，并可设定一个磁盘名称(Volume Label)用来辨认磁盘。
FORMAT B:/S	将驱动器 B 内的磁盘格式化，并将操作系统文件(IBMBIO.COM、IBMDOS.COM 及 COMMAND.COM)拷贝到格式化的磁盘上。利用这种方式格式化的磁盘可作为以后开机时的引导盘。

请注意，FORMAT 指令在格式化过程中会将原磁盘上的内容全部销毁，因此在使用时要特别小心。

2. SYS

当使用 FORMAT d:/S 或 FORMAT d:/B 格式化磁盘时，可以保留磁盘上的某些特殊位置，以备存放 DOS 系统文件，然后再利用 SYS 指令将当前磁盘上的 DOS 系统转移到所指定的磁盘上，以建立系统磁盘。

3. DISKCOPY

用来拷贝整个磁盘，以作备份用。DISKCOPY 是唯一能够直接使用未格式化磁盘的 DOS 指令。若目标磁盘在拷贝前没有格式化，则 DISKCOPY 在复制操作进行时会自动将它格式化。

上述的 COPY 指令与 DISKCOPY 指令都可用来拷贝磁盘上的文件，DISKCOPY 指令一般都是用来拷贝整个磁盘，并且会将目标磁盘的原有文件全部删除掉。但是 COPY 指令一般都是用来拷贝部分的文件到目标磁盘上，不会删除原有的文件，如果要利用 COPY 指令来拷贝整个磁盘也可以用如下的指令行来完成：

COPY A:*.* B:

当拷贝完成后，最好利用如下的 DISKCOMP 指令与 COMP 指令来检验一下是否拷贝正确。

4. DISKCOMP

用来比较两张磁盘以确定是否拷贝正确。例如：

DISKCOMP A: B:　用来比较驱动器 A 与驱动器 B 中的磁盘是否完全相同。

DISKCOMP A: A:　在驱动器 A 中比较两张磁盘是否相同，此时 DOS 会提示你插入适当的磁盘，当你放好磁盘后按下任一键就可继续比较的工作。

DISKCOMP 会比较所有的磁道，如果有的磁道的内容不同，就会显示一条信息，此信息包括具有不一样内容的磁道号码，以及它是在磁盘的哪一面（0 或 1）。当比较完成后，DISKCOMP 会显示如下提示信息：

Compare another diskettes(Y/N)?

来询问你是否要继续比较磁盘，如要按 Y 继续，否则按 N 结束。DISKCOMP 指令只能用来比较以 DISKCOPY 指令拷贝过的磁盘，如果是以 COPY 指令拷贝的磁盘则需利用如下的 COMP 指令。

5. COMP

用来比较两个文件以确定内容是否相同。例如：

COMP C:YOUNG.XXX C:YOUNG.YYY

用来比较硬盘 C 上的 YOUNG.XXX 和 YOUNG.YYY 文件。

COMP A:MYFILE1 C:MYFILE2

用来比较驱动器 A 上的 MYFILE1 和硬盘驱动器 C 上的 MYFILE2 文件。

COMP A:MYFILE C:

用来比较驱动器 A 和硬盘驱动器 C 的 MYFILE 文件。

COMP A:*.BAK C:*.TXT

用来比较驱动器 A 上扩展名为 BAK 与硬盘驱动器 C 上扩展名为 TXT 而文件名相同的文件。

COMP A:*.* C:*.*

用来比较驱动器 A 与硬盘驱动器 C 上文件名与扩展名完全相同的文件。

在下达 COMP 指令时如果没有指明任何参数，那么 COMP 指令在比较之前会提示你输入所要比较的文件，而在比较过程中，如果两个文件对应的位置中内容不同，就会显示错误信

息指出对于无法吻合的字节在该文件中的位移量(Offset),以及这两个字节的实际内容(用十六进制表示)。显示的格式如下所示:

File 1 = XX

File 2 = XX

其中 File 1 是第一组文件,而 File 2 则是第二组文件。当 COMP 指令发现有十几个不相同的地方后,它就假设其他进一步的比较都是没有用的,就会停止比较的工作而显示如下信息:

10 Mismatches - ending compare

如果比较成功了,那么 COMP 指令就会显示:

Files compare OK

当比较完两个文件后,比较的工作就会继续到下一组能吻合的两个显示文件上,直到找不出进一步吻合的文件为止,于是 COMP 指令就会显示:

Compare more files(Y/N)?

如果要比较另外两组文件就需按 Y 继续,于是又会提示你输入所要比较的文件;否则按 N 停止比较的工作。

在文件的比较工作中,COMP 指令都会寻找正在被比较的文件最后一个字节,以确定其是否为一个文件终结标记(EOF——End of File),也就是 Ctrl+Z。如果找到了,COMP 指令就不会有其他的操作,如果找不到这个文件终结标记,COMP 指令就会显示如下信息:

Eof mark not found

出现这个信息就指出在文件中可以使用的数据部分仍然是正确的,因此 COMP 指令最后还是显示:

Files compare OK

表示比较的结果没有问题。

6. CHKDSK

用来检查磁盘目录及文件信息,并列出磁盘及内存容量的状态信息,内容如下:

● 磁盘的识别名称和磁盘格式化的日期,当格式化磁盘时,假如没有指定磁盘名称,则上述两者都不会显示。

● 磁盘的序号(Volume Serial Number)。

● 磁盘上的总存储空间。

● 隐藏文件(Hidden files)数量及其所占用的磁盘空间。

● 文件目录(Directory)的数目及其所占用的空间。

● 用户文件(User files)的数目及其所占用的空间。

● 磁盘上剩余空间的总字节数(Byte)。

除了上述磁盘的状况外,CHKDSK 也可用来查看主机当前内存的状态,并显示如下信息:

● 所安装内存的总字节数。

● 未使用的内存的字节数。

另外 CHKDSK 指令还可以配合参数/F 和/V 来达到不同的目的,例如 CHKDSK/F 除了可显示类似如上的信息外,并且会自动修复在文件配置表(File Allocation Table)的目录中发生的任何错误。在你结束 DOS 操作前,最好利用这个指令来检查你的磁盘是否一切正常?如果

发生任何问题，则可以尽早修复它。因此当 CHKDSK 指令显示如下提示信息时只要按 N 即可。

X lost clusters found in Y chains.

Convert lost chains to files(Y/N)?

而 CHKDSK/V 除了可以显示当前磁盘的状态信息外，还可显示所有的文件名以及它们的路径。

7. RECOVER

用来将受损的磁盘文件恢复。如受损的部分位于存储某文件的扇区，可能就无法正常地读取该文件了。如要恢复文件可以在 RECOVER 指令后输入该文件名即可。例如：

RECOVER YOUNG. XXX

当文件被恢复后，存储在受损部分附近的数据将会消失，同时也可能在文件后面加上一些多余的数据，因此，用户必须自行检查恢复后的文件，以便删除那些不必要的数据。

使用 RECOVER 指令时也可以配合通配符来指定所要恢复的文件，但 RECOVER 指令只会恢复第一个符合的文件。

如果受损的部分位于存储目录的扇区，你也可以用 RECOVER 指令把磁盘中的所有文件恢复。例如：

RECOVER A:

此时 RECOVER 指令就会假定整个目录都是不正确的，然后检查在驱动器 A 中的文件配置表以及一连串的配置单位，并且在目录中产生如下格式：

FILE0000. REC, FILE0001. REC, FILE0002. REC,...

每一个 FILEnnnn. REC 都表示一个磁盘中被恢复的文件名。RECOVER 指令只能将存储在受损扇区的文件恢复，并不能恢复其中所存储的数据。当然 RECOVER 指令不能恢复那些已被删除的文件。使用 RECOVER 指令后，如要检查某一恢复的文件是否正确必须逐一确认，然后再利用 REN 指令将文件名更改为原来的文件名。因此，在使用 RECOVER 指令时要非常谨慎。

8. LABEL

用来建立、改变或删除磁盘名称。磁盘名称一般都利用 FORMAT d:/V 来赋予，而 LABEL指令却可以在磁盘已经格式化后来建立、改变或删除磁盘名称，因此在使用上具有灵活性。磁盘名称是由长度不超过 11 个字符所组成的字符串，其命名方式完全由用户决定，只要名称中不含下列字符即可：

. " / \ [ ] : | < > + = ; , ? *

如果赋予的磁盘名称超过 11 个字符，那么 DOS 只会接受前面 11 个字符，例如：

LABEL A:SAMPLE DISKS

即赋予驱动器 A 的磁盘名称为 SAMPLE DISK 而不是 SAMPLE DISKS。如果在下达LABEL 指令时未指定磁盘名称，屏幕就显示如下提示信息：

Volume in drive A is XXXXXXXXXXX

Volume label(11 characters. ENTER for none)?

如果要赋予磁盘一个磁盘名称，可以在此输入它。如果要改变一个已经存在的磁盘名称，所输入的新磁盘名称将会取代原来旧有的名称。如果要删除一个磁盘名称，就不要输入任何磁盘名称，直接按 Enter 键即可，接着屏幕显示如下提示信息：

Delete current volume label(Y/N)?

如确定要删除，按 Y 即可，否则按 N。

第三章 批处理指令

3.1 批处理指令

批处理指令(Batch Command)是用于批处理文件内的 DOS 指令，批处理指令共有九个而且都是 DOS 的内部指令。像一般的 DOS 指令一样，批处理指令也可以直接在 DOS 提示符下直接输入使用，只是批处理指令通常都配合批处理文件才能真正发挥其作用。现在，我们将以实例依次来说明这九个批处理指令的用法。为了配合实际的操作，请先检查一下你的驱动器 A 的磁盘上，是否含有 EDLIN.COM 的指令文件以便编写批处理文件，并在驱动器 B 内插入任何一张 DOS 系统磁盘即可。但是，本书所显示的执行结果与屏幕上的结果可能不一致，这完全要看你所使用的磁盘内容是什么而定。例如下列的批处理文件 TEST.BAT：

```
A:\>TYPE TEST.BAT
:PROGRAM NAME:TEST.BAT
DIR *.BAT
DIR *.COM

A:\>
```

当你执行这个批处理文件时，将会列出驱动器 A 中全部以 BAT 和 COM 为扩展名的文件，如下所示：

```
A:\>TEST (输入 TEST)
A:\>DIR *.BAT

Volume in drive A has no label
Volume Serial Number is 214F-1100
Directory of A:\

TEST        BAT        48   04-29-92    10:02a
        1 file(s)          48 bytes
                      270336 bytes free

A:\>DIR *.COM

Volume in drive A has no label
Volume Serial Number is 214F-1100
Directory of A:\
```

```
EDIT        COM      413   04-09-91    5:00a
DISKCOPY    COM    11793   04-09-91    5:00a
COMMAND     COM    47845   04-09-91    5:00a
        3 file(s)          60051 bytes
                       270336 bytes free

A:\>
```

由执行结果可看出，所有扩展名为 BAT 和 COM 的文件都被显示出来，就像你自己输入 DIR 指令一样。然而上面所显示的执行结果可能和你所执行的结果不一样。

另外，建立批处理文件时要特别注意下面的情况，以免造成错误：

● DOS 使用"<"和">"来改变系统的标准输出和输入，因此不要在批处理文件内使用这两个符号，除非你有特别目的。

● 在批处理文件中符号"%"也有特别的用途，如果你想要执行的指令行中含有"%"就必须用"%%"代替。例如，你想显示文件名 FILE%1 的目录时，平常你会输入：

```
DIR FILE%1
```

但是在批处理文件中，你必须输入：

```
DIR FILE%%1
```

来告诉 DOS 第一个"%"是不具有任何意义的符号。"%"在批处理文件中的用法请参考下面批处理指令的说明。

3.2　ECHO 指令

ECHO 指令可以让你决定当 DOS 执行批处理文件时，是否要把各指令在屏幕上显示出来，但它并不会与指令执行时所产生的信息相互干扰。其使用格式如下：

```
ECHO [ON|OFF|message]
```

正常情况下，当 DOS 在执行批处理文件时都会在屏幕上显示所执行的各个指令，也就是 ECHO 原来都被设定为 ON 的状态，如要停止显示这些指令可以设定 ECHO 为 OFF，例如：

```
ECHO OFF
```

如果在批处理文件执行的同时要显示提示信息，可在 ECHO 后加上所要显示的信息，例如：

```
ECHO YOUR FILE HAS JUST BEEN COPIED
```

那么，无论 ECHO 是 ON 或 OFF，DOS 都会将提示信息显示在屏幕上。如果在下达 ECHO 指令时并未加上任何参数，例如：

```
ECHO
```

DOS 将会显示当前 ECHO 被设定的状态为 ON 或 OFF。

建议你在开始设计批处理文件时最好先设定 ECHO 为 ON，当完成测试后再将 ECHO 指令放在批处理文件的第一行并设定成 OFF 的状态，以避免指令的显示。DOS 将会保持当前 ECHO指定的设定状态直到你改变它为止。

实例说明：如下范例中，批处理文件内容包含了下面几条指令：

```
;PROGRAM NAME:ECHOTEST.BAT
ECHO    ECHO TEST SAMPLE
ECHO OFF
DIR *.BAT
ECHO
ECHO ON
ECHO
DIR *.BAT
```

当执行这个批处理文件时,屏幕显示如下:

```
A:\>ECHOTEST (输入 ECHOTEST)

A:\>ECHO    ECHOTEST SAMPLE
ECHO TEST SAMPLE

A:\>ECHO OFF

Volume in drive A has no label
Volume Serial Number is 214F  1100
Directory of A:\

TEST            BAT         48   04-29-92   10:02a
ECHOTEST        BAT         78   10-10-86   10:17a
        2 file(s)       126 bytes
                     269312 bytes frec

ECHO if off

A:\>ECHO
ECHO is on
A:\>DIR *.BAT

Volume in drive A has no label
Volume Serial Number is 214F-1100
Directory of A:\

TEST                BAT         48   04   29   92   10:02a
ECHOTEST            BAT         78   10   10   86   10:17a
        2 file(s)       126 bytes
                     269312 bytes free

A:\>
```

在上面的实例中,ECHO ECHO TEST SAMPLE 及 ECHO OFF 会被显示出来,而 DIR

*.BAT 及 ECHO 指令就不会显示了，因为在这之前 ECHO 已被设定成 OFF 状态，但是 DIR 指令的输出仍然会显示，接着又把 ECHO 恢复为 ON，因此你又可以看到 ECHO 及 DIR *.BAT 这两个指令了。

如果要让 ECHO 指令本身在执行时不显示出来，DOS 3.3 以上的版本可以利用@符号来抑制它，例如下列的批处理文件：

```
: PROGRAM NAME:NOECHO.BAT
@ECHO    ECHO TEST SAMPLE
@ECHO OFF
DIR *.BAT
@ECHO
@ECHO ON
@ECHO
DIR *.BAT
```

执行的结果显示如下：

```
A:\>NOECHO (输入 NOECHO)
ECHO TEST SAMPLE

 Volume in drive A has no label
 Volume Serial Number is 214F-1100
 Directory of A:\

TEST         BAT       48   04-29-92   10:02a
ECHOTEST     BAT      107   04-29-92   10:39a
NOECHO       BAT      110   04-29-92   10:47a
        3 file(s)        265 bytes
                      268288 bytes free
ECHO is off
ECHO is on

A:\>DIR *.BAT

 Volume in drive A has no label
 Volume Serial Number is 214F-1100
 Directory of A:\

TEST         BAT       48   04-29-92   10:02a
ECHOTEST     BAT      107   04-29-92   10:39a
NOECHO       BAT      110   04-29-92   10:47a
        3 file(s)        265 bytes
                      268288 bytes
```

```
A:\>
```

配合@符号来抑制 ECHO 指令的显示，屏幕上所显示出来的效果是否比较理想呢？

3.3 注解(:)指令

批处理文件内虽然只是一系列的 DOS 指令，但是慢慢地你将发现批处理文件的设计其实还是需要一些程序设计的技巧，才能发挥批处理文件的最大效果。因此，在批处理文件内适当地加上相关的注解来说明程序是必要的，注解只是一行含有文字说明的语句，DOS 并不会执行这些注解文字。如果要在批处理文件内加入注解，其使用格式如下：

```
: COMMENT
```

换句话说，只要在注解文字前加上一冒号(:)即可，DOS 在执行批处理文件时将会跳过这些注解而不予执行。如下范例就是加上了充分说明的批处理文件：

```
:PROGRAM NAME:LIST.BAT
:*
:* BATCH FILE TO LIST COMMAND FILES AND BATCH FILES
:*
:* LIST THE COMMAND FILES ON ALL THE DISKS
   DIR A:*.EXE
   DIR B:*.EXE
   DIR C:*.EXE
:*
:* LIST THE BATCH FILES ON ALL THE DISKS
   DIR A:*.BAT
   DIR B:*.BAT
   DIR C:*.BAT
:*
:* FINISHED
:*
```

这个批处理文件用于显示驱动器 A、B 及 C 中扩展名为.EXT 和.BAT 的所有文件，如果你的主机还没有安装硬盘驱动器，执行这个批处理文件时除了显示错误信息外，并不会造成其他的错误。当然你也可以在相关的 DIR 指令前加入冒号，就可避免执行这些指令。

上面的批处理文件其执行结果显示如下：

```
A:\>LIST    (输入 LIST)

A:\>        DIR A:*.EXE

Volume in drive A has no label
Volume Serial Number is 214F-1100
Directory of A:\
```

```
CHKDSK      EXE      16200    04－09－91    5:00a
EDLIN       EXE      12642    04－09－91    5:00a
          2 file(s)          28842 bytes
                             267264 bytes free

A:\>      DIR B:*.EXE

Volume in drive B has no label
Volume Serial Number is 2859－14D6
Directory of B:\

File not found

A:\>      DIR C:*.EXE

Volume in drive C has no label
Volume Serial Number is 1751－A237
Directory of C:\

TEST        EXE      154624    04－12－92    10:56a
          1 file(s)          154624 bytes
                             77518848 bytes free

A:\>      DIR A:*.BAT

Volume in drive A has no label
Volume Serial Number is 214F－1100
Directory of A:\

TEST        BAT        48    04－29－92    10:02a
ECHOTEST    BAT       107    04－29－92    10:39a
NOECHO      BAT       110    04－29－92    10:47a
LIST        BAT       412    04－29－92    10:53a
          4 file(s)          677 bytes
                             267264 bytes free

A:\>      DIR B:*.BAT

Volume in drive B has no label
Volume Serial Number is 2859－14D6
Directory of B:\

File not found
```

```
A:\>    DIR C:*.BAT

Volume in drive C has no label
Volume Serial Number is 1751-A237
Directory of C:\

AUTOEXEC      BAT      211   04   11   92   4:57a
        1 file(s)          211 bytes
                     77518848 bytes free

A:\>
```

由执行结果可以看出，批处理文件内所加入的注解不但不会影响指令的执行，而且可以增加程序的阅读性。

3.4 REM 指令

利用 REM 指令可以从批处理文件内显示说明的信息，其使用格式如下：

```
REM message
```

说明的信息(Message)可以是长度不超过 123 个字符的任何字符串，例如：

```
REM YOUR FILE HAS JUST BEEN COPIED
```

那么，DOS 将会显示这个说明的信息在屏幕上。REM 指令和 ECHO 指令的差别是：在ECHO 为 OFF 的状态下，DOS 将不会显示 REM 后的任何信息，但 ECHO 后的信息仍会被显示出来。

任何指令前只要加入 REM，DOS 就视为是说明的信息将它显示在屏幕上而不会执行，例如：

```
REM ERASE *.*
```

这种方式在测试批处理文件时非常方便，因为你可以暂停某些指令的执行，当想要执行时再将 REM 指令删除即可。

3.5 PAUSE 指令

PAUSE 指令可用来暂停批处理文件的执行，其使用格式如下：

```
PAUSE [message]
```

指定说明的信息其字符串长度不可超过 121 个字符。当 DOS 遇到 PAUSE 指令时便会先显示该信息，再显示如下提示文字：

```
Press any key to continue...
```

然后暂停下来，直到你按下任一键后再继续下一个指令的执行。如果 ECHO 被设定为 OFF 状态，那么 DOS 就显示如上的提示信息。

使用 PAUSE 指令的目的是：

1. 用来提醒用户某些需要注意的重要事项，例如：

```
:PROGRAM NAME:PAUSE1.BAT
ECHO THIS BATCH FILE IS ABOUT TO ERASE YOUR FILE
PAUSE
ERASE *.*
```

执行后屏幕上将会显示如下：

```
A:\>PAUSE1  (输入 PAUSE1)

A:\>ECHO THIS BATCH FILE IS ABOUT TO ERASE YOUR FILE
THIS BATCH FILE IS ABOUT TO ERASE YOUR FILE
A:\>PAUSE
Press any key to continue...          (按 Ctrl+Break 键)
^C
Terminate batch job(Y/N)? Y           (按 Y 表示要终止执行)

A:\>
```

这时用户如不想将文件删除，可以按 Ctrl+Break 键来停止批处理文件的执行，以免磁盘上的文件全部被删除掉。

2. 用来要求用户做一些必要的操作，例如：

```
:PROGRAM NAME:PAUSE2.BAT
ECHO OFF
ECHO PLACE THE DISKETTE IN DRIVE A:
PAUSE
DIR A:
```

执行后屏幕上将显示如下：

```
A:\>PAUSE2  (输入 PAUSE2)

A:\>ECHO OFF
PLACE THE DISKETTE IN DRIVE A:
Press any key to continue...
```

以提醒用户将磁盘插入驱动器 A 中，然后按下任一键就可显示驱动器 A 上的文件目录。

3.6　GOTO 指令

GOTO 指令可将执行的流程转移到某一标记(Label)上，其使用格式如下：

```
GOTO LABEL
```

在批处理文件中所谓的标记就是在一行中以冒号(:)为首，其后紧跟一个最长八个字符的名称，其格式如下：

```
:LABEL
```

如果标记名称的长度超过 8 个字符，DOS 仍以前面的 8 个字符为名称。GOTO 指令会使 DOS 去执行：LABEL 那一行下方的指令，如果：LABEL 没有被定义，那么这一个批处理文件的执行操作就会停止，并且显示如下信息：

Label not found

下面就是一个 GOTO 指令的范例程序：

```
:PROGRAM NAME:LOOP.BAT
:* ----------
:* BATCH FILE TO DISPLAY A MESSAGE REPEATEDLY
:*

:LOOP
    ECHO THIS LINE IS DISPLAYED OVER AND OVER
    GOTO LOOP
    :*
    :* FINISHED
    :* ----------
```

在上例中，我们用 LOOP 及 GOTO LOOP 形成一个连续的循环，因此在执行时会在屏幕上连续不断地以 ECHO 指令显示如下信息，直到你按下 Ctrl+Break 键为止。

```
A:\>LOOP   (输入 LOOP)

A:\>
A:\>    ECHO THIS LINE IS DISPLAYED OVER AND OVER
THIS LINE IS DISPLAYED OVER AND OVER

A:\>    GOTO LOOP

A:\>    ECHO THIS LINE IS DISPLAYED OVER AND OVER
THIS LINE IS DISPLAYED OVER AND OVER

A:\>    GOTO LOOP

A:\>    ECHO THIS LINE IS DISPLAYED OVER AND OVER
THIS LINE IS DISPLAYED OVER AND OVER

A:\>    GOTO LOOP

A:\>    ECHO THIS LINE IS DISPLAYED OVER AND OVER
THIS LINE IS DISPLAYED OVER AND OVER

A:\>    GOTO LOOP
```

```
A:\>      ECHO THIS LINE IS DISPLAYED OVER AND OVER
THIS LINE IS DISPLAYED OVER AND OVER

A:\>      GOTO LOOP

A:\>      ECHO THIS LINE IS DISPLAYED OVER AND OVER
THIS LINE IS DISPLAYED OVER AND OVER

^C        (按 Ctrl+Break 键)

Terminate batch job(Y/N)? Y      (按 Y 终止执行)

A:\>
```

由于冒号(:)可被用来当成注解指令也可作为一个标记的符号,因此 DOS 都把紧接在冒号后的 8 个字符看成标记名称,即使你认为这是一个注解行。为了避免 DOS 造成混淆,注解的前八个字符最好不要与标记名称相同。另外要小心的是:DOS 无法区别大小写的字母,而且也会把紧接在冒号后的空白忽略掉。因此,如下的两行指令:

```
:finished
:FINISHED WITH THE FILES FOR NOW
```

DOS 都认为是相同的,而把二者当作是标记名称"finished"。

如果你在一个批处理文件中同时指定了一个以上的标记名称,DOS 将会使用头一个而把其他的忽略掉。

3.7　FOR 指令

利用 FOR 指令可以反复地执行 DOS 的指令,其使用格式如下:

```
FOR %%variable IN(set) DO command
```

%%variable 这个变量会逐次地以 set 集合中指定的元素来作为变量值。当得到一个值后,DOS 就执行 command 这一指令。例如:

```
FOR %%B IN (A: B: C:) DO DIR %%B
```

DOS 将执行"DIR %%B"三次,而每一次 DOS 会以集合中的各元素来取代"%%B":先是"A:",然后是"B:",最后是"C:",也就是 DOS 将会依次执行如下的指令:

```
DIR A:
DIR B:
DIR C:
```

所谓变量是指其值可以改变的名称,在 FOR 指令中所使用的变量名称是在%%之后紧接着一个字母,大小写都可。如果是在 DOS 提示符下输入 FOR 指令,那么只需一个%就够了,例如你可以输入:

```
A>FOR %A IN (FILE1) DO DIR %A
```

格式中所指定的 command 可以是除 FOR 指令以外的任何 DOS 指令,也就是说,FOR 指令不

能使用嵌套(Nested)的写法。格式中所用的 set 集合，你可以指定任何的项目，如集合中包含了文件名称，你也可以使用通配符"?"和"*"，那么%%variable 就会设定成磁盘中能够匹配的每一个文件名。例如，下面 FOR 指令将会显示磁盘中所有以.TXT 为扩展名的文件内容：

```
FOR %%B IN (*.TXT) DO TYPE %%B
```

当然，在 SET 集合中也可使用路径名称。

现在，我们就来介绍一下参数的用法。

为了执行一个批处理文件，只要在 DOS 提示符下输入文件名即可。但在输入文件名时，也可在文件名后指定某些字符串作为参数(Parameters)使用。例如，想要执行一个批处理文件，其文件名为 A:YOUNG.BAT，如果输入：

```
A:YOUNG XXX YYY ZZZ
```

表示你指定了三个参数：XXX、YYY 及 ZZZ。而在批处理文件中可用如下变量：

```
%1  %2  %3  %4  %5  %6  %7  %8  %9
```

来参考第一到第九的参数，在此例中变量%1 将被"XXX"所取代，%2 被"YYY"所取代，%3 被"ZZZ"所取代，其余的为空的变量。

你可以在一个批处理文件的指令中加入一些变量，使得批处理文件在执行时可用你所指定的参数来取代它，这种参数的使用方法，可让你以不同的数据来执行一个批处理文件，例如下面的范例：

```
:PROGRAM NAME:TYPES.BAT
:*
:* BATCH FILE TO DISPLAY THREE FILES
:*
:* DISPLAY THE FIRST FILE
   TYPE %1
:* DISPLAY THE SECOND FILE
   TYPE %2
:* DISPLAY THE THIRD FILE
   TYPE %3
:*
:* FINISHED
:*
```

如果这个批处理文件的文件名为 A:TYPES.BAT，而你输入：

```
A:TYPES NAMES.TXT ADDRESS.TXT PHONE.TXT
```

DOS 将执行如下的指令：

```
TYPE NAMES.TXT
TYPE ADDRESS.TXT
TYPE PHONE.TXT
```

如果你是输入：

```
A:TYPES CLASS1.TXT CLASS2.TXT CLASS3.TXT
```

那么 DOS 就执行如下的指令：

```
TYPE CLASS1.TXT
TYPE CLASS2.TXT
TYPE CLASS3.TXT
```

在一个批处理文件中最多只能使用十个变量(即%0到%9),但在指令行上却可以指定十个以上的参数,请看看下面的SHIFT指令的介绍即可了解。%0是一个特殊的变量,这个变量通常是被批处理文件本身的驱动器名称(如果指定的话)或文件名取代。如果你不需要使用到批处理文件本身,则变量从%1开始即可。例如下列的范例:

```
:PROGRAM NAME:TYPEIT.BAT
:*
:* BATCH FILE THAT DISPLAYS ITSELF
:*
:* DISPLAY THIS FILE
   TYPE %0.BAT
:*
:* FINISHED
:*
```

如果这个批处理文件的文件名为TYPEIT.BAT.而你输入:

```
A:TYPEIT
```

DOS便会执行如下的指令:

```
TYPE A:TYPEIT.BAT
```

但要注意批处理文件内的TYPE指令,必须要在%0后加上.BAT的扩展名,这是因为变量%0并没有指定扩展名。执行结果如下:

```
A:\>TYPEIT     (输入 TYPEIT)

A:\>    TYPE TYPEIT.BAT
:PROGRAM NAME:TYPEIT.BAT
:*
:* BATCH FILE THAT DISPLAYS ITSELF
:*
:* DISPLAY THIS FILE
   TYPE %0.BAT
:*
:* FINISHED
:*
A:\>
```

可见,执行这个批处理文件后,就会自动地显示批处理文件本身的内容。

3.8 SHIFT指令

SHIFT指令可让你在指令行上使用超过十个以上的参数,以便改变%0到%9的变量值。

其使用格式非常简单，就是：

SHIFT

执行一次 SHIFT 指令，就会把所有在指令行上的参数向左移一个，因此%0 这个变量值就会被%1 所取代，而%1 的变量值也就被%2 所取代，依此类推。其后的其他 SHIFT 指令同样也会造成每一个参数全部向左移位。例如你要执行一个批处理文件其文件名为 YOUNG.BAT，当你输入：

YOUNG A B C

那么各变量值分别是：

%0	%1	%2	%3	%4	%5	%6	%7	%8	%9
YOUNG	A	B	C						

注意，%4 到%9 为空的变量值。当执行一个 SHIFT 指令后，各变量值变成：

%0	%1	%2	%3	%4	%5	%6	%7	%8	%9
A	B	C							

请注意，变量%0 的原值已消失而被参数 A 所取代。

当你执行一个批处理文件时，如果使用的参数超过了九个，那么只有前面九个参数会先被指定到变量上，然而剩余的参数仍会被 DOS 保留下来。当你使用到 SHIFT 指令时，第一个被保留的参数就被左移到变量%9 上，因此 SHIFT 指令允许你使用九个以上的参数。例如，当你输入如下的文件名及参数时：

YOUNG A B C D E F G H I J K

各变量值先被设定为：

%0	%1	%2	%3	%4	%5	%6	%7	%8	%9
YOUNG	A	B	C	D	E	F	G	H	I

当执行一个 SHIFT 指令后，各变量值变成：

%0	%1	%2	%3	%4	%5	%6	%7	%8	%9
A	B	C	D	E	F	G	H	I	J

再执行一个 SHIFT 指令后，各变量值又变成：

%0	%1	%2	%3	%4	%5	%6	%7	%8	%9
B	C	D	E	F	G	H	I	J	K

如果又执行一个 SHIFT 指令，各变量值则变成：

%0	%1	%2	%3	%4	%5	%6	%7	%8	%9
C	D	E	F	G	H	I	J	K	

要注意，%9 现在已变成空的变量值了，同时也要注意，只要你执行一个 SHIFT 指令，变量%0 的原值立即消失。如果你想使用变量%0 的原值，那就必须在执行第一个 SHIFT 指令之前完成。

SHIFT 指令的第一种用途是：当你无法确定有几个参数时，利用 SHIFT 指令来处理。例如下面的范例：

```
:PROGRAM NAME:FILES.BAT
:*  ----------  ----------
:* BATCH FILE TO DISPLAY ANY NUMBER OF FILES
```

```
:*
:LOOP
:* DISPLAY THE FILE WITH NAME %1
   TYPE %1
:* SET %1 TO THE NEXT FILE
   SHIFT
:* REPEAT THE WHOLE PROCESS
   GOTO LOOP
:*
:* FINISHED
:* ---  ---  ---  ---  ---  ---  ---  ---
```

当 DOS 执行这个批处理文件时，变量%1 将逐次地被每一个参数所取代，而被用来显示各文件的内容，如果你输入：

```
FILES  FILE1  FILE2  FILE3
```

DOS 将会依次地显示各文件的内容。但你执行这个批处理文件时却有一个问题：它并不会停止！如你要停止它，可以在最后一个变量被处理过后按 Ctrl+Break 键来终止它的执行。例如：

```
A:\>FILES FILE1 FILE2 FILE3      (输入此指令)

A:\>     TYPE FILE1
* * * FILE1 * * *

A:\>     SHIFT

A:\>     GOTO LOOP

A:\>     TYPE FILE2
* * * FILE2 * * *

A:\>     SHIFT

A:\>     GOTO LOOP

A:\>     TYPE FILE3
* * * FILE3 * * *

A:\>     SHIFT

A:\>     GOTO LOOP

A:\>     TYPE
Required parameter missing
```

```
A:\>    SHIFT

A:\>    GOTO LOOP

A:\>    TYPE
Required parameter missing

A:\>
Terminate batch job(Y/N)? Y    (按 Ctrl+Break 键或 Y 终止)
A:\>
```

如果要避免这种执行不停而需以按键方式来终止的问题，请利用下面介绍的 IF 指令及参考下一章范例三的程序。

3.9 IF 指令

IF 指令让你可以有条件的执行 DOS 指令，其使用方式有如下三种：

1. 比较两个字符串，其使用格式如下：

IF [NOT] string1 == string2 command

格式中的字符串(string)中也可使用%0 到%9 的变量，而指令(command)可以是除 IF 指令外的任何 DOS 指令，也就是 IF 指令不允许嵌套(Nested)的用法。DOS 将比较两个字符串，如果相等就执行所指定的指令，否则 DOS 就跳过这些指令。假如你在格式当中使用了 NOT，那么在两个字符串不相等时 DOS 才会执行这些指令，下面的例子是用来显示其文件名为%2 的文件内容，但条件是变量%1 的值一定要与"DISPLAY"相同：

```
IF %1 == DISPLAY TYPE%2
```

而下面的例子则是用来删除驱动器 A 上的全部文件，除非变量%1 的值不是"stop"。

```
IF %1 == stop ERASE A:*.*
```

IF 指令最大的用处是可以配合 GOTO 指令来使用，下面的例子可用来删除两个文件，除非变量%1 的值为"save"。

```
IF %1 == save GOTO CONTINUE
ERASE FILE1
ERASE FILE2
:CONTINUE
.
.
.
```

请特别注意：条件中两个字符串比较时，两者的英文字母和大小写必须是完全相同条件才会成立。换句话说，大小写的英文字母在比较时是不一样的。

2. 检查文件是否存在，其使用格式如下：

IF [NOT] EXIST [d:] [path] filename[.ext] command

格式中的指令(command)可以是除 IF 指令外的任何 DOS 指令。如果文件名(filename)在你所指定的磁盘目录中，条件 EXIST 就成立，那么 DOS 就执行你所指定的指令，否则 DOS 就跳过这些指令。假如你在格式中使用了 NOT，那么就变成文件名不存在时才执行这些指令。

下面的例子是用来显示 FILE1 文件(如果该文件存在)：

```
IF EXIST FILE1 DIR FILE1
```

而下面范例则是一个比较大的批处理文件，其功能是以变量%1 为文件名，如果该文件存在，就将它从驱动器 A 拷贝到驱动器 C 上。但不论文件是否存在，执行这个批处理文件时都会显示适当的信息：

```
:PROGRAM NAME:MYCOPY.BAT
:* ----------
:* BATCH FILE TO COPY A FILE FROM A: TO C:
:*
:* IF THE FILE DOES NOT EXIST DISPLAY A MESSAGE
      IF EXIST %1 GOTO FOUND
      ECHO YOUR FILE WAS NOT FOUND
      GOTO STOP
:*
:* IF THE FILE EXISTS,COPY IT AND DISPLAY A MESSAGE
:FOUND
      COPY A:%1 C:%1
      ECHO THE FILE %1 HAS BEEN COPIED FORM A: TO C:
:*
:STOP
:* FINISHED
:* ----------
```

首先输入 MYCOPY FILE6 以便拷贝 FILE6 文件，执行结果如下：

```
A:\>MYCOPY FILE6  (输入此行指令)

A:\       IF EXIST FILE6 GOTO FOUND

A:\       ECHO YOUR FILE WAS NOT FOUND
YOUR FILE WAS NOT FOUND  (表示 FILE6 文件不存在)

A:\       GOTO STOP

A:\>
```

如果再输入 MYCOPY FILE2 则执行结果显示如下：

```
A:\>MYCOPY FILE2 (输入此行指令)

A:\       IF EXIST FILE2 GOTO FOUND
```

```
A:\>      COPY A:FILE2 C:FILE2
          1 file(s) copied

A:\>      ECHO THE FILE FILE2 HAS BEEN COPIED FROM A: TO C:
THE FILE FILE2 HAS BEEN COPIED FORM A: TO C:
          (表示 FILE2 文件已成功地被拷贝到驱动器 C:上)
A:\>
```

3. 测试 ERRORLEVEL 值,其使用格式如下:

IF [NOT] ERRORLEVEL number command

格式中的指令(command)可以是除 IF 指令外的任何 DOS 指令。DOS 将会测试 ERRORLEVEL这个特殊的变量,如果 ERRORLEVEL 的值等于或大于你在格式中所指定的数值(number),那么 DOS 就执行你所指定的指令(command),否则 DOS 就跳过这些指令。假如你在格式中使用了 NOT,ERRORLEVEL的值就变成小于 number 的值时才执行这些指令。

有经验的程序设计员对于 IF 指令的这种使用方式会特别有兴趣,因为它根据某个程序的执行结果来决定是否执行某些指令。而程序的执行结果值是由 DOS 所提供 ERRORLEVEL 这个变量值来取得,一个程序如果能正常地完成,ERRORLEVEL 就被设定成 0,若不能正常地完成,ERRORLEVEL 就被设定成比 0 大的值,因此,利用 ERRORLEVEL 值我们可以测试程序的执行结果以决定你所指定的条件是否成立。请看下面的例子:

```
IF ERRORLEVEL 2 GOTO ABORT
```

如果 ERRORLEVEL 的值等于或大于 2,DOS 就执行 GOTO 指令。在 DOS 指令中有 BACKUP、FORMAT、GRAFTABL、KEYB、REPLACE 及 RESTORE 指令能够设定 ERRORLEVEL,各个指令的设定值如下表说明:

BACKUP 指令:

设定值	执行结果
0	正常完成
1	未找到要做备份的文件
2	因文件共享发生冲突而造成文件未做备份
3	用户按 Ctrl+Break 键终止备份的工作
4	由于错误所造成的终止

FORMAT 指令:

设定值	执行结果
0	正常完成
3	用户按 Ctrl+Break 键终止磁盘格式化的工作
4	由于错误所造成的终止
5	当用户被询问是否格式化硬盘时,按下"N"键

GRAFTABL 指令:

设定值	执行结果
0	正常执行
1	原先装入的符号数据表已被新表所取代
2	原先未装入符号数据表也没有装入新表
3	使用了不正确的参数
4	不正确的 DOS 版本

KEYB 指令：

设定值	执行结果
0	正常执行
1	语法错误
2	未找到键盘定义文件或是该文件已损坏
3	无法建表
4	CON：设备有问题
5	指定一个未准备好的代码页
6	未找到翻译表
7	不正确的 DOS 版本

REPLACE 指令：

设定值	执行结果
0	正常执行
1	不正确的 DOS 版本
2	源文件找不到
3	路径找不到
5	文件的存取属性不正确
8	内存空间不足
11	使用了不正确的参数

RESTORE 指令：

设定值	执行结果
0	正常完成
1	没有找到文件
2	因文件共享发生冲突而造成备份文件未被存回
3	用户按 Ctrl+Break 键终止文件存回的工作
4	由于错误所造成的终止

其他可以设定 ERRORLEVEL 的指令，将在以后各章中进一步的说明。请看如下的范例

及操作情况：

```
:PROGRAM NAME:MYBACKUP.BAT
:* ---- ---- ---- ---- ---- ---- ----
:* BATCH FILE TO TEST ERRORLEVEL
    ECHO OFF
    CLS
    BACKUP C:*.BAK A:
    IF ERRORLEVEL 1 ECHO !! WRONG BACKUP !!
:* END
:* ---- ---- ---- ---- ---- ---- ----
```

在执行此批处理文件前，请先确定驱动器 A 中的磁盘上含有 FORMTA.COM 和 BACKUP.COM 两个文件以便进行 BACKUP 的工作。

```
A:\>MYBACKUP      (输入 MYBACKUP)

Insert backup diskette 01 in drive A:

WARNING! files in the target drive
A:\root directory will be erased
Press any key to continue...

* * * Backing up files to drive A: * * *
Diskette Number:01

\BATCH5\TEST.BAK
\BATCH5\YOUNG.BAK
A:\>
```

执行 MYBACKUP 批处理文件后，可看到 BACKUP 指令制作备份成功而未显示错误信息：WRONG BACKUP。但如果事先把硬盘 C 上的 *.BAK 文件全部删除后再执行 MYBACKUP批处理文件，就会造成 BACKUP 指令无法找到要制作备份文件而显示如下的错误信息：

```
A:\>C:

C:\BATCH5>ERASE *.BAK

C:\BATCH5>DIR *.BAK

Volume in drive C has no label
Volume Serial Number is 1751  A237
Directory of C:\BATCH5
```

```
File not found                    (表示已无扩展名为 BAK 的文件)

C:\BATCH5)A:

A:\)MYBACKUP                      (再输入 MYBACKUP 执行一次)

A:\)      ECHO OFF

WARNING! No files were found to backup
!! WRONG BACKUP !!      (显示错误信息)

A:\)
```

当你在执行 BACKUP、FORMAT、GRAFTABL、KEYB、REPLACE 及 RESTORE 指令时，可以设计如下的批处理文件来显示这些指令的执行结果，以便掌握这些指令的操作情况：

```
A:\)
:PROGRAM NAME:BACKUPS. BAT
@ECHO OFF
REM Backup the hard drive
ECHO Backing up the hard drive...
ECHO Have a stack of disks handy,then
BACKUP D:\*. * A:/S/M
IF ERRORLEVEL 4 GOTO ERROR4
IF ERRORLEVEL 3 GOTO ERROR3
IF ERRORLEVEL 2 GOTO ERROR2
IF ERRORLEVEL 1 GOTO ERROR1
ECHO Backup was successful
GOTO END
:ERROR4   —error,program stopped
ECHO Some kinda error took place
GOTO END
:ERROR3——Control  Break pressed,stopping the backup
ECHO Backup halted! (You pressed Control—Break)
GOTO END
:ERROR2 ——File sharing conflicts
ECHO Three are file sharing conflicts.
ECHO Please contact the network administrator for assitance
GOTO END
:ERROR1—  Nothing to backup
ECHO No files found to backup,no need to procede
Bye!
ECHO
:END
```

3.10 CHOICE 指令

CHOICE 指令是 DOS 6.0 新增的交互式批处理指令，它可以显示指定的提示信息并暂停，以便你按键选择。其使用格式为：

CHOICE [/C:keys][/N][/S][/T:c,nn][message]

单独使用 CHOICE 指令，将会显示预设的提示信息[Y,N]?，然后暂停下来让你输入大小写的字母 Y 或 N。如果要区别大小写字母，则可以使用/S 选项，你甚至可以使用/N 选项来取消预设的提示信息，而自行指定提示信息。例如：

```
A:\>CHOICE
[Y,N]? Y
A:\>CHOICE ARE YOU SURE
ARE YOU SURE [Y,N]? Y
A:\>CHOICE /S /N Are you sure? (Y)es or (N)o →
Are you sure? (Y)es or (N)o =>Y
```

从上例可以看出，CHOICE 指令不但可以显示你所要的提示信息，而且也可以要求用户输入字母 Y 或 N。当然你还可以利用/C 选项来指定其他的按键，例如：

```
A:\>CHOICE /C:1234 CHOOSE ONE
CHOOSE ONE [1,2,3,4]? 3
A:\>CHOICE /N /C:1234 CHOOSE ONE (1 - 4) =>
CHOOSE ONE (1—4) =>3
```

在/C 选项后指定你所要的按键，显示时将会在中括号内以逗号隔开。你也可以使用/T 选项来指定暂停的时间，例如：

```
A:\>CHOICE /N /C:1234 /T:1,10 ARE YOU SURE →
ARE YOU SURE =>1
```

当显示提示信息后，如果在十秒钟内未输入选择项，那么 CHOICE 指令就会自动输入所指定的选择项(即数字 1)。

用户输入选择项后，CHOICE 指令就会依次设定 ERRORLEVEL 参数的值，也就是选择第一项(即设定数值 1)，选择第二项(即设定数值 2)，依此类推。如果输入非指定的选择项，CHOICE 指令会发出响声以提醒用户。如果 CHOICE 指令检测到错误的状况，则会设定 ERRORLEVEL的值为 255，如果按了 Ctrl+Break 或 Ctrl+C 键，则设定 ERRORLEVEL 的值为 0。

因此，你可以利用 IF 指令来测试 ERRORLEVEL 参数的设定值，就可知道用户所输入的选择项，并依此执行各选择项的功能。例如下列的范例：

```
:PROGRAM NAME:CHOICES.BAT
:* — - ------ ---- ------ ---- ------ --
@ECHO OFF
ECHO
ECHO 1. DIR
ECHO 2. DIR /W
```

```
ECHO 3. DIR /P
ECHO.
CHOICE /N /C:123 CHOOSE ONE =}

IF ERRORLEVEL 3 GOTO DIR1
IF ERRORLEVEL 2 GOTO DIR2
IF ERRORLEVEL 1 GOTO DIR3

:DIR1
DIR
GOTO END

:DIR2
DIR /W
GOTO END

:DIR3
DIR /P
GOTO END

:END
:* ------------------------------------
```

当你执行这个批处理文件时，屏幕上会显示一个功能目录供你按键选择，以执行不同的 DIR 指令功能。例如：

```
A:\>CHOICES

1. DIR
2. DIR /W
3. DIR /P

CHOOSE ONE =}2        (按 2 选择第二项功能)

Volume in drive A is BATCH
Volume Serial Number is 095F-1DE7
Directory of A:\

COMMAND.COM   AUTOEXEC.BAT   CURRENT     HELP.BAT     ABOUT.ALL
README        CONFIG.SYS     [UTILITY]   [HELPFILE]   [SAMPLES]
[OTHERS]      CHOICES.BAT
        12 file(s)      55820 bytes
                       838656 bytes free
```

3.11 批处理文件的链接

你可以在批处理文件中执行另一个批处理文件，这种方式称为批处理文件的链接(Chaining)。但在正常的情况下，当第二个批处理文件执行完成后，流程并不会回到原来的批处理文件上。因此，你只能从一个批处理文件的最后一行来执行另一个批处理文件。

如果你在完成第二个批处理文件的执行后想要回到原批处理文件上，有一种方法就是利用 DOS 所提供的 COMMAND 指令与 EXIT 指令即可达到批处理文件链接的真正效果。COMMAND 指令的使用格式如下：

COMMAND [/C command]

COMMAND 指令是 DOS 的内部指令，其目的是用来使用第二个指令处理程序(Secondary Command Processor)。如果你输入：

COMMAND

DOS 将会装入第二个指令处理程序，以便让你执行你想要执行的指令。如要结束第二个指令处理程序然后回到原来的指令处理程序，可以利用 EXIT 指令来完成，EXIT 指令也是 DOS 的内部指令。其使用格式就是单纯的：

EXIT

虽然只有一些有经验的程序设计员才会想到用第二个指令处理程序，但你还可以利用"/C"参数来装入第二个指令处理程序以便让你执行某一个指令。例如，当你输入：

COMMAND /C DIR A:

此时第二个指令处理程序将会执行"DIR A:"指令后自动停止，然后 DOS 便回到原来的指令处理程序。这也就是如何从一个批处理文件的中途来链接另一个批处理文件的答案：利用 COMMAND 指令配合"/C"参数来启动另一个批处理文件即可，当第二个批处理文件执行完成后，便会再回到原来的批处理文件并继续完成执行的工作。

现举例说明这种链接的技巧：假如你有两个批处理文件，其文件名为 TEST1.BAT 与 TEST2.BAT，你如果想要从 TEST1.BAT 来链接 TEST2.BAT，只要在 TEST1.BAT 的批处理文件中下达"TEST2"即可。例如下列的链接实例：

```
:PROGRAM NAME:TEST1.BAT
REM TEST1.BAT
TEST2
ECHO OFF
ECHO GOOD -BYE FROM TEST1.BAT

:PROGRAM NAME:TEST2.BAT
REM TEST2.BAT
ECHO OFF
ECHO SEE YOU FROM TEST2.BAT
```

假如从 TEST1.BAT 来链接 TEST2.BAT，那么执行结果为：

```
A:\)TEST1      (输入 TEST1)
```

```
A:\>REM TEST1.BAT

A:\>TEST2

A:\>REM TEST2.BAT

A:\>ECHO OFF
SEE YOU FROM TEST2.BAT

A:\>
```

从上面执行结果可以看出，TEST2.BAT 执行完成后，DOS 的流程并未再回到 TEST1.BAT 继续执行下去。如果你想要在 TEST2.BAT 执行完成后让 DOS 流程再回到 TEST1.BAT 上，那就非使用下面的指令不可了：

```
COMMAND /C TEST2
```

现在，用如下的实例来链接就可达到这种效果：

```
:PROGRAM NAME:TESTA.BAT
REM TESTA.BAT
COMMAND /C TESTB
ECHO OFF
ECHO GOOD-BYE FROM TESTA.BAT

:PROGRAM NAME:TESTB.BAT
REM TESTB.BAT
ECHO OFF
ECHO SEE YOU FROM TESTB.BAT
```

假如从 TESTA.BAT 来链接 TESTB.BAT，那么执行结果为：

```
A:\>TESTA      (输入 TESTA)

A:\>REM TESTA.BAT

A:\>COMMAND /C TESTB

A:\>REM TESTB.BAT

A:\>ECHO OFF
SEE YOU FROM TESTB.BAT

A:\>ECHO OFF
GOOD-BYE FROM TESTA.BAT
A:\>
```

其实，批处理文件的链接方式也可以直接利用 DOS 3.3 版本以上所提供的批处理指令 CALL 来完成调用的工作。因此，下列的技巧仍然可以达到一样的链接效果：

```
:PROGRAM NAME:MYCALL.BAT
REM MYCALL.BAT
CALL MYSUB
ECHO OFF
ECHO GOOD--BYE FROM MYCALL.BAT

:PROGRAM NAME:MYSUB.BAT
REM MYSUB.BAT
ECHO OFF
ECHO SEE YOU FROM MYSUB.BAT
```

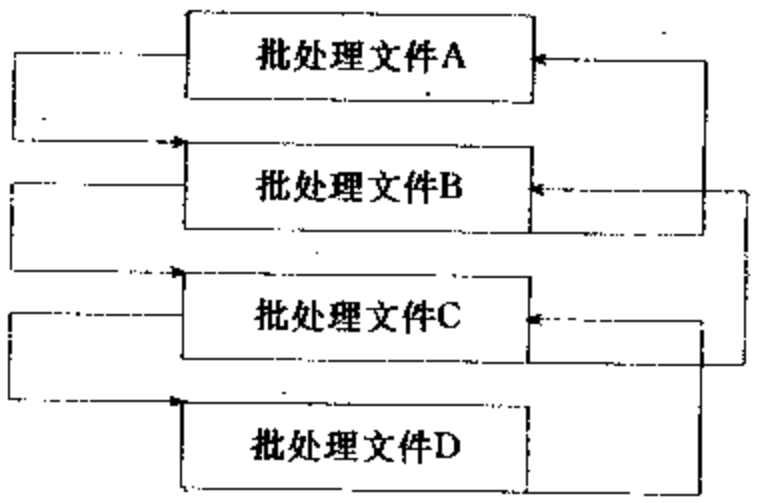
```
A:\>MYCALL          (输入 MYCALL)

A:\>REM MYCALL.BAT
A:\>CALL MYSUB
A:\>REM MYSUB.BAT

A:\>ECHO OFF
SEE YOU FROM MYSUB.BAT
A:\>ECHO OFF
GOOD-BYE FROM MYCALL.BAT


A:\>
```

批处理文件链接时可以形成如下的嵌套结构，也就是一个批处理文件调用(Call)另一个，而另一个批处理文件又调用第三个……，依此一直调用下去，而流程返回时则由最后一个被调用(Called)的批处理文件按序回到上一个调用它的批处理文件上，依此而回到第一个批处理文件上(如下图所示)。



使用这种嵌套结构的链接方式时，由于每一个新的指令处理程序都会占用一部分的内存空间直到它结束为止，如果你所链接的批处理文件太多那就很可能造成内存空间不足的错误。这时候你一定要修改原设计，使每一个批处理文件都从原批处理文件的最后一行上被链接起来，这样你就可以不必使用到 COMMAND 或 CALL 指令了。

第四章 批处理文件范例讨论

4.1 前言

在此，我们将综合上一章所介绍的各种批处理指令，来设计几个完整的批处理文件，让你可以真正了解批处理指令的功能和用法，以奠定你设计批处理文件的基础，并且培养你的实际经验，以便往后能设计功能更强更大的批处理文件，来满足各种需求。

4.2 范例讨论

1. 范例一

这个程序是利用一个批处理文件来更改DOS 指令的名称。由于CHKDSK 这个 DOS 指令最容易拼写错，因此我们就先选择 CHKDSK 指令来试试看，程序如下：

```
:PROGRAM NAME:CH.BAT
:* ------------ Batch file example:1 ------------
:*
:* Batch file to rename the CHKDSK Command
:*
:* Execute the CHKDSK command with up to three parameters
      CHKDSK %1 %2 %3
:*
:* ------------ ------------ ------------ ------------
```

我们赋予这个批处理文件一个比较不易拼错的文件名如 CH.BAT，那么当你要使用 CHKDSK 指令时便可输入"CH"而不必输入"CHKDSK"。

但根据 CHKDSK 指令的使用格式，你最多可以指定三个参数，注意参数是如何被传送到真正的 CHKDSK 指令上的，如果你输入：

```
CH A:
```

这个批处理文件会执行：

```
CHKDSK A:
```

如果在批处理文件内的指令使用了一个空变量，那么 DOS 就不会以任何的值来取代这个变量，因此，只要在这个指令上指定 0～3 个参数，它都可以接受。例如，当你输入：

```
CH A:/F /V
```

这个批处理文件将会执行：

```
CHKDSK A:/F /V
```

请看如下的操作情况：

A:\>CH　　(输入 CH)

A:\> CHKDSK
Volume Serial Number is 214F－1100

362496 bytes total disk space
146432 bytes in 33 user files
216064 bytes available on disk

1024 bytes in each allocation unit
354 total allocation units on disk
211 available allocation units on disk

655360 total free bytes memory
588544 bytes

A:\>
A:\>CH :A/V　　(输入此行指令)

A:\> CHKDSK A:/V
Volume Serial Number is 214F－1100
Directory of A:\

A:\TEST. BAT
A:\EDIT. COM
A:\CHKDSK. EXE
A:\EDLIN. EXE
A:\DISKCOPY. COM
A:\COMMAND. COM
A:\ECHOTEST. BAT
A:\NOECHO. BAT
A:\LIST. BAT
A:\PAUSE1. BAT
A:\PAUSE2. BAT
A:\LOOP. BAT
A:\TYPES. BAT
A:\TYPEIT. BAT
A:\FILES. BAT
A:\FILE1
A:\FILE2
A:\FILE3
A:\MYCOPY. BAT

```
A:\MENU.BAT
A:\MYBACKUP.BAT
A:\BACKUPS.BAT
A:\TEST1.BAT
A:\TESTA.BAT
A:\TEST2.BAT
A:\TESTB.BAT
A:\SAMPLE.BAS
A:\QB.BAT
A:\MYCALL.BAT
A:\MYSUB.BAT
A:\SAMAPLE.EXE
A:\CTL.BAT
A:\SAMPLE.OBJ
     362496 bytes total disk space
     146432 bytes in 33 user files
     216064 bytes available on disk

       1024 bytes in each allcation unit on disk
        354 total allocation units on disk
        211 available allocation units on disk

     655360 total bytes memory
     588528 bytes free

A:\>
```

2. 范例二

当你使用 IF 指令来测试两个字符串是否相同时，可能会遇到一个问题，那就是如果其中一个字符串为空字符串则会造成错误。例如下面的指令行：

```
IF %1 == /D DIR
```

当第一个参数等于字符串"/D"时便能执行 DIR 指令以显示当前预设驱动器的文件目录，现假设%1 的值等于"AAA"，那么 IF 指令就变成：

```
IF AAA == /D DIR
```

显然条件不成立，DOS 就不会执行 DIR 指令了。但如果没有指定参数而造成空的变量时，那么 IF 指令就变成：

```
IF == /D DIR
```

这种情况将造成错误，因为 IF 指令的语法规定在"=="的两边都要有一个字符串。为了要克服这个问题，你可以在每一个字符串前加上一个字符，如下例：

```
IF X%1 == X/D DIR
```

如果%1 的值为"/D",IF 指令变成:

IF X/D == X/D DIR

那么 DIR 指令就被执行。但如果%1 的值不是"/D"而是"QQQ",那么 IF 指令变成:

IF XQQQ == X/D DIR

条件不成立,当然不会执行 DIR。再假设%1 是空字符串,IF 指令就变成:

IF X == X/D DIR

虽然条件不成立,却不会造成错误,因为"=="两边永远都会有两个字符串存在。

利用下面的范例来说明这个技巧:

```
:PROGRAM NAME:EQUAL.BAT

:* ---------- Batch file example:2 ----------
:*
:* Batch file to test if %1 is empty
:*
:* Test if %1 is empty
        if x%1 == x goto empty
:*
:* Display message - %1 is not empty
        echo %%1 is not empty
        goto stop
:*
:* Display message - %1 is empty
:empty
        echo %%1 is empty
:*
:stop
:* ----------
```

如果将这个批处理文件的文件名定为 EQUAL.BAT,当你执行这个批处理文件时如果指定了参数,例如:

EQUAL QQQ

那么 DOS 就会显示如下信息:

%1 IS NOT EMPTY

表示%1 不是空的变量。如果在执行时未指定任何参数,例如:

EQUAL

那么 DOS 将显示如下信息:

%1 IS EMPTY

表示%1 是空的变量。

请注意,为了要显示"%",在 ECHO 指令内一定要使用"%%",但在不被执行的注解行中就不需要了。

操作情况如下:

```
A:\>EQUAL A      (输入 EQUAL A)
A:\>     if xA == x goto empty
A:\>     echo %1 if not empty
%1 is not empty
A:\>     goto stop
A:\>
A:\>EQUAL      (输入 EQUAL)
A:\>     if x == x goto empty
A:\>     echo %1 is empty
%1 is empty
A:\>
```

3. 范例三

在批处理文件的程序设计中，IF 指令常常会与 GOTO 指令配合来达到控制流程的目的。然而，当你在设计一个较大的批处理文件时如果滥用了太多的 GOTO 指令，不但无法提高执行的效率，反而增加程序修改时间。因此，为了避免使用 GOTO 指令时造成困扰，请你遵守如下的使用规则：

除了以 GOTO 指令来形成一个循环外，流程往上跳的 GOTO 指令绝对不要使用。

利用下面所提供的范例来说明这个规则：

```
:PROGRAM NAME:DISPLAY.BAT
:* ---------- Batch file example:3 ----------
:* Batch file to display files
:*
:* if there are no files to display,stop
:loop
      if x%1 == x goto stop
:*
:* Display the file %1
      echo Display:%1
      type %1
:*
:* Shift th next file name into %1
      shift
:*
:* Process the next file
      goto loop
:*
:* Display message there are no more files—and stop

:stop
echo No more files to display
:* ----------
```

这个批处理文件的文件名为 DISPLAY.BAT，而它的目的是用来显示你指定的任何文件的内容，如果你输入：

DISPLAY FILE1 FILE2 FILE3 FILE4

那么 DOS 会依次显示各个文件的内容。请看如下的操作情况：

```
A:\>DISPLAY FILE1 FILE2 FILE3 FILE4      (输入此行指令)
A:\>      if xFILE1 == x goto stop
A:\>      echo Displaying:FILE1
Displaying:FILE1
A:\>      type FILE1
* * * FILE1 * * *
A:\>      shift
A:\>      goto loop
A:\> if xFILE2 == x goto stop
A:\>      echo Displaying:FILE2
Displaying:FILE2
A:\>      type FILE2
* * * FILE2 * * *
A:\>      shift
A:\>      goto loop
A:\>      if xFILE3 == x goto stop
A:\>      echo Displaying:FILE3
Displaying:FILE3
A:\>      type FILE3
* * * FILE3 * * *
A:\>      shift

A:\>      goto loop
A:\>      if x == x goto stop
A:\>      echo No more files to display
No more files to display
A:\>
```

在执行过程中，屏幕上除了显示执行的结果外同时也会把批处理文件内的指令一起显示出来，这是由于 ECHO 的状态未被指定为 OFF 的缘故。如果你觉得屏幕上这样的显示可能会影响执行的效果，可以在执行批处理文件前先设定 ECHO 为 OFF 状态，就会在屏幕上只显示执行的结果。例如：

```
A:\>ECHO OFF
DISPLAY FILE1 FILE2 FILE3      (输入 ECHO OFF 后，DOS 提示符会消失，请直
Displaying:FILE1                接输入 DISPLAY FILE1 FILE2 FILE3 即可)
* * * FILE1 * * *
Displaying:FILE2
* * * FILE2 * * *
```

```
Displaying:FILE3
* * * FILE3 * * *
No more files to display
ECHO ON                                    (再输入 ECHO ON 以便显示 DOS 提示符)
A:\>
```

4. 范例四

很多的应用程序常需要在屏幕上显示一个功能表(MENU),以供用户选择操作。现在,我们就提供一个程序范例来说明如何在批处理文件中设计这种具有交互式(Interactive)效果的菜单。假如你经常使用到如下四个程序:

1. 文字处理(Word Processor)
2. 试算表(Spreadsheet)
3. 数据库管理(Database Management)
4. 电传通信(Telecommunication)

而每一个程序都有一个批处理文件来启动它,分别是:

1. BAT
2. BAT
3. BAT
4. BAT

因此为了方便起见,可以设计一个批处理文件来显示如下的功能,让用户以按键的方式来选择所要执行的程序:

```
* * * Program Menu * * *
1 -- Word Processing
2 -- Spreadsheet
3 -- Database Management
4 -- Telecommunication
Please enter your choice:
```

显示这个功能表的批处理文件的文件名为 MENU.BAT,程序内容如下:

```
:PROGRAM NAME:MENU.BAT
:* ------------ Batch file example:4 ------------
:* Batch file to display a menu
      echo off
:*
:* Clear the screen
      cls
:*
:* Display the various choices
echo * * * Program Menu * * *
      echo 1 -- Word Processing
      echo 2 -- Spreadsheet
```

```
echo 3 — — Database Management
echo 4 — — Telecommunication
echo Please enter your choice:
:* ————————————————————————————————
```

执行结果显示如下：

```
A:\)MENU　　(输入 MENU)
A:\)　　echo off
* * * Program Menu * * *
1 — — Word Processing
2 — — Spreadsheet
3 — — Database Management
4 — — Telecommunication
Please enter your choice:
A:\)
```

执行这个批处理文件后只要按"1"、"2"、"3"或"4"就可启动你所要的程序。如果你想在每次装入 DOS 时自动执行 MENU 这个批处理文件而显示如上的功能表，可在 AUTOEXEC. BAT 文件(请参考第九章启动文件的使用说明)的最后一行加上"MENU"。

在你执行 MENU. BAT 的批处理文件之前，要先设计好所要执行的批处理文件(例如 1. BAT、2. BAT、3. BAT 和 4. BAT 等)，否则在用按键选择功能表的项目时，并不会达到预期的功能。另外，以这种方式来要求用户用按键选择时，也没有防止错误的效果。

如果你拥有 DOS 6. 0，那么就可以利用 CHOICE 指令来改良 MENU. BAT 程序如下：

```
:PROGRAM NAME:CMENU. BAT
:* ———————————— Batch file example:4—1 ————————————
:* Batch file to display a menu
    echo off
:*
:* Clear the screen
    cls
:*
:* Display the vaious choices
    echo * * * Program Menu * * *
    echo 1 — — Word Processing
    echo 2 — — Spreadsheet
    echo 3 — — Data Management
    echo 4 — — Telecommunication
    echo
    choice /c:1234 Please enter your choice

:* Execute the menu function depend on choice
    if errorlevel 4 goto menu4
    if errorlevel 3 goto menu3
```

```
        if errorlevel 2 goto menu2
        if errorlevel 1 goto menu1

:menu1
        echo You enter 1
        goto end

:menu2
        echo You enter 2
        goto end

:menu3
        echo You enter 3
        goto end

:menu4
        echo You enter 4
        goto end

:end
:* ---------- ---------- ---------- ----------
```

执行结果显示如下：

```
A:\>CMENU

A:\>  echo off
* * * Program Menu * * *
1 -- Word Processing
2 -- Spreadsheet
3 -- Data Management
4 -- Telecommunication
Please enter your choice[1,2,3,4]? 1
You enter 1
```

你只要输入所需的选择项，就可显示相关的提示信息。你可以自行加入你所要执行的指令，以便配合目录上所显示的功能。

5. 范例五

这个程序范例是提供给熟悉 QuickBASIC 程序设计语言的用户参考的。由于在操作编译式的 QuickBASIC 语言时，必须按照如下的三个步骤来进行，那就是：

1. 编辑 QuickBASIC 的源程序(Editing)
2. 将源程序编译成目标程序(Compiling)
3. 目标程序的连接及执行(Linking & Executing)

但在实际操作这三个步骤时都要依次输入相关的指令来进行，因此有时候感到相当不便。如果要省掉这些按键的时间，利用批处理文件来解决这个问题是最理想的。下面就是这个批处理文件的程序内容：

```
:PROGRAM NAME:RUNQB.BAT
:*
:* EDIT QuickBASIC SOURCE PROGRAM
EDIT A:\%1.BAS
:*
@ECHO * MAKE SURE ALL QuickBASIC FILES *
@ECHO * ARE IN THE DIRECTORY: C:\QB45 *
PAUSE
C:
CD\QB45
:* COMPILE SOURCE PROGRAM
BC A:%1 /O/V;
IF ERRORLEVEL 1 GOTO NOGOOD
:*
:* LINK OBJECT PROGRAM
LINK %1;
:
:* RUN EXECUTEABLE PROGRAM
%1
A:
:*
:NOGOOD
:*
:* END
```

实际的操作如下：

```
A:\)RUNQB SAMPLE      (输入此行指令，以便编辑 SAMPLE 程序文件)
A:\)EDIT A:\SAMPLE.BAS
```

此时屏幕会显示 EDIT 程序的编辑菜单，如果你使用本书所附的磁盘，编辑菜单内会显示如下 QuickBASIC 程序范例，否则你可以编辑自己的程序：

```
'PROGRAM NAME:SAMPLE.BAS
I = 1
A$ = "TOTAL IS:"
DO WHILE I <= 100
    T = T+I
    I = I+1
LOOP
PRINT A$;T
```

```
END
```

编辑完成 EDIT 程序后，就开始进行编译、连接及执行的工作。由于本书的操作范例是将 QuickBASIC 的全部程序安装在硬盘的 C:\QB45 目录下，因此屏幕会显示如下信息来提醒你注意：

```
* MAKE SOURE ALL QuickBSIC FILES *
* ARE IN THE DIRECTORY: C:\QB45 *

A:\>PAUSE
Press any key to continue...
```

如果你还没有准备好，可以按 Ctrl＋C 键来终止批处理文件的执行，以便安装 QuickBASIC。否则按下任一键就可继续执行下去。

```
A:\>C:

C:\>CD\QB45
C:\QB45>BC A:SAMPLE /O/V;
Microsoft (R) QuickBASIC Compiler Version 4.50
(C) Copyright Microsoft Corporation 1982 - 1988.
All rights reserved.
Simultaneously published in the U.S. and Canada.

43901 Bytes Available
43588 Bytes Free

   0 Warning Error(s)
   0 Severe Error(s)

C:\QB45>IF ERRORLEVEL 1 GOTO NOGOOD

C:\QB45>LINK SAMPLE;
Microsoft (R) Overlay Linker Version 3.69

Copyright (C) Microsoft Corp 1983—1988. All rights reserved.

C:\QB45>SAMPLE

TOTAL IS:5050      (QuickBASIC 程序的执行结果)

C:\QB45>A:
A:\>
```

在进行上面的操作时，如果有任何的错误出现，都可以按 Ctrl＋C 键来终止批处理文件的

执行，以便修改 QuickBASIC 的源程序。如果要再回到 EDIT 编辑程序，只要再输入如下的指令即可：

RUNQB SAMPLE

修改完成后，批处理文件就可继续处理编译、连接及执行的工作。如果又有错误出现，则再重复如上的步骤，直到执行正确为止。

6. 范例六

这个范例与上一个范例的功能相似，它们的差别是：这个范例是用来进行 COBOL 语言的程序编辑、连接及执行。你可以根据你的需求来修改这个批处理文件的内容，甚至将它改成其他程序设计语言（例如 Pascal 或 C）的批处理文件也可以。完整的程序显示如下，供你参考：

```
:PROGRAM NAME:RUNCOBOL.BAT
:* ---------- Batch file example:5 ----------
:* Batch file for editing,compiling,linking &
:*      executing a COBOL program

:* Editing a COBOL source program in drive B:
:*
        pause Insert the SYSTEM diskette in drive B:
        b:
        edit %1.cob
:*
:* Compiling source program to object program
        pause Insert the COBOL diskette in drive A
:*
        a:
        cobol b:%1;
:*
:* Linking object program & executing
        pause Insert the LIBRARY diskette in drive A
:*
        link b:%1;
        %1
:*
:* Finished
:* ----------
```

第五章 树状文件结构的指令用法

5.1 树状文件结构

当你使用 FORMAT 指令对磁盘格式化时，DOS 会在磁盘上建立一个能说明各文件特征的目录。由于 PC 主机所安装的驱动器(尤其是硬盘驱动器)，其存储容量都非常大，为了使磁盘建立文件的系统更具有灵活性，DOS 允许你在磁盘上建立额外的目录，这些目录称为子目录(Subdirectory)。子目录将磁盘分成若干个不同的存储区，每个存储区在使用上就如同一张独立的磁盘。而整个磁盘主目录(Main Directory)就是由各子目录组成，主目录也就是在磁盘格式化时，由 DOS 自动建立的目录，而子目录则是由用户借助 DOS 所提供的有关指令建立的任意目录。

当你在文件结构上增加多层分支时，就好像树枝向外延伸分叉一样，因此这种文件结构常被称为树状结构(Tree－Structured)的文件系统，主目录就是它的根，因此主目录也称为根目录(Root Directory)，分支就是子目录。树状结构文件系统的关系如图 5－1 所示：

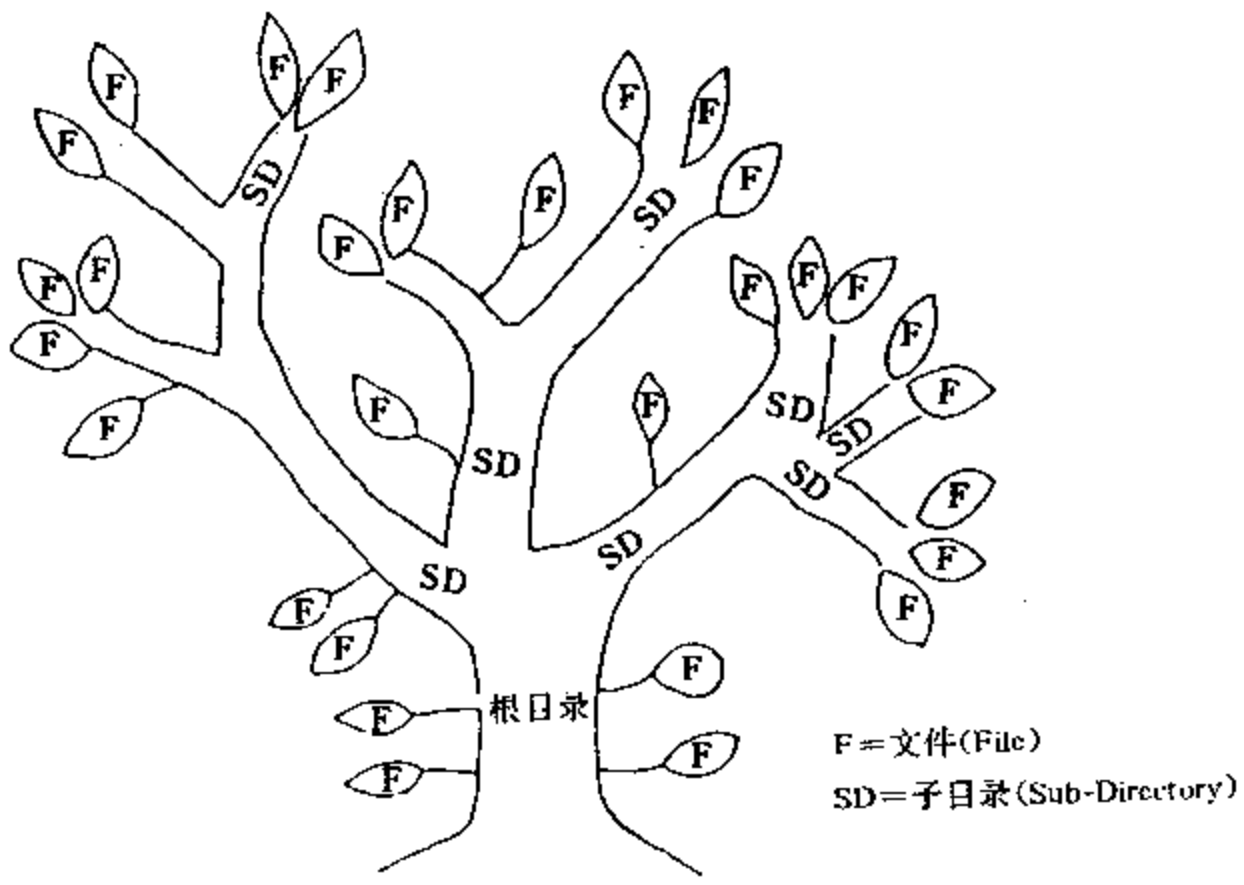


图 5－1

现在利用一个实例来说明树状的文件系统。例如有一个地区性的棒球联盟，该联盟有三个俱乐部：Red、Green 及 Blue。Red 和 Green 俱乐部各有一个球队，而 Blue 俱乐部则有男性和女

性两支球队。而且每支球队都拥有自己的球迷和与比赛有关的球员、赛程及成绩。这个棒球联盟与球队的附属关系如图 5－2 所示：

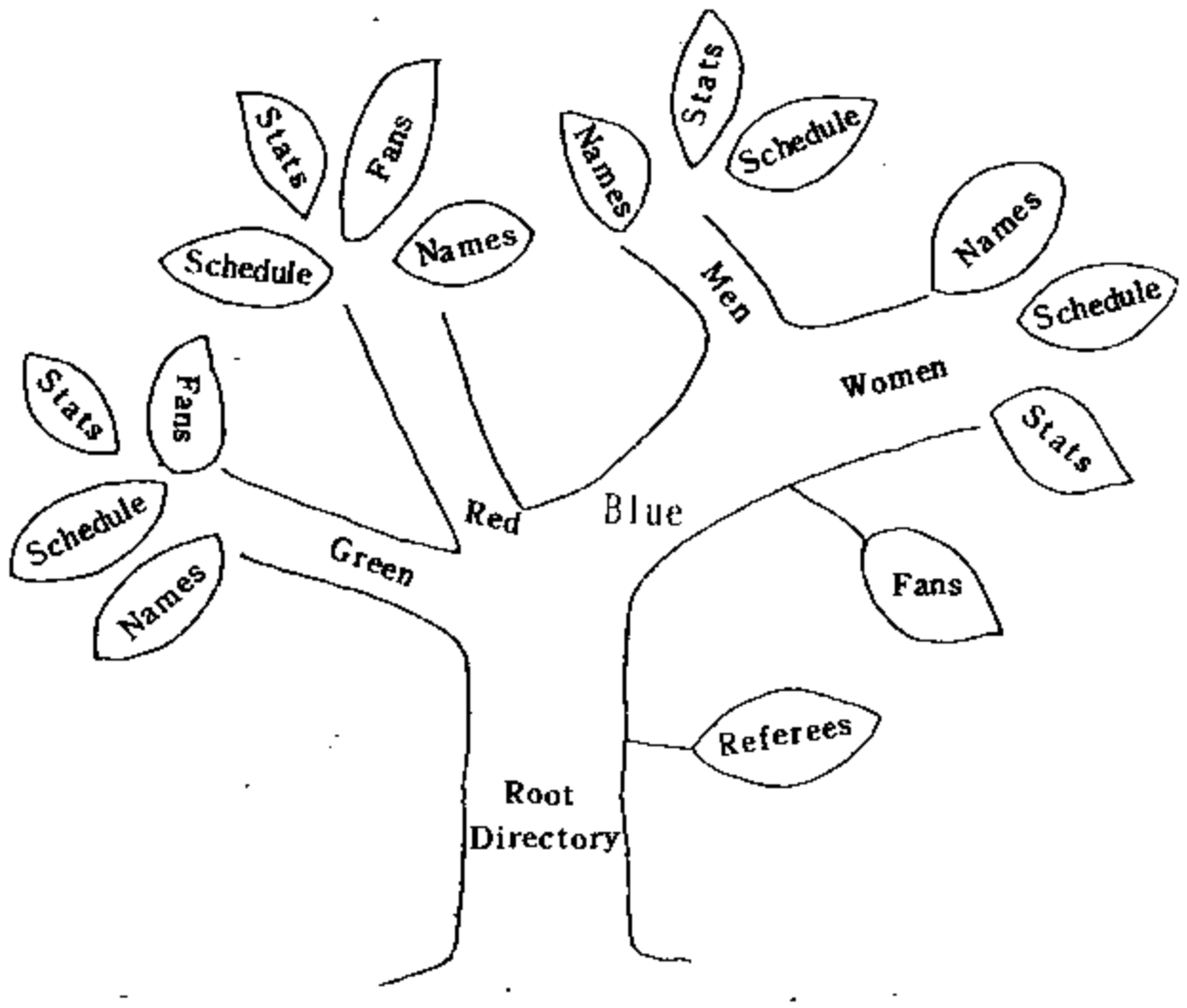


图 5－2

棒球联盟的主要工作是管理如下数据：

每一个俱乐部的球迷(Fans)动态；

每一个俱乐部的球员姓名(Names)、胜负(Stats)及赛程(Schedule)。

另外，也需要掌握各裁判(Referees)的姓名。

上述的数据分别存储在不同的文件中，由上图的树状结构可知根目录包含一裁判文件(Referees)和三个子目录(俱乐部 Red、Green 及 Blue)，而子目录 Blue 本身又有两个目录(Men 及 Women)。每一个俱乐部都有一个球迷文件(Fans)，而每一个球队子目录包括三个文件(Names、Stats 及 Schedule)。

在树状结构中，由于每一个文件所属的目录不尽相同，因此当你要指明某一个文件时，必须要告诉 DOS 这个文件属于哪一个目录。因此，你可以指定一条到达文件的路径(Path)以便 DOS 去寻找所要的文件，如果未指明寻找的路径，DOS 就只会在当前所选定的磁盘目录上寻找。当装入 DOS 时，当前目录(Current Directory)就是启动 DOS 磁盘的根目录，除非你使用 CHDIR 指令把当前目录改变到其他子目录上。例如：

\BLUE\MEN\NAMES

就是用来指定 Blue 俱乐部男性(MEN)球队的 NAMES 文件。其中\BLUE\MEN\便是用来寻找 NAMES 文件的路径。如果路径是以"\"符号为首，则 DOS 会从根目录以下开始寻找，

否则就从当前目录找起。假设当前目录是\BLUE，那么只要：

MEN\NAMES

就可以找到 NAMES 文件了。如果当前目录是\BLUE\MEN，当然只要：

NAMES

就可以找到该文件了。如果要找到 REFEREES 文件，使用：

\REFEREES

即可，这里的路径就是单纯的"\"符号，也就是根目录。但无论如何都要记住：指定路径时最多不能超过 63 个字符。

5.2 子目录的指令用法

子目录其实就是一个包含目录索引的文件，这些索引在形式上与主目录的索引完全相同，只是子目录的索引数目并没有限制，唯一的要求就是只要在磁盘中有足够的空间存放得下即可。

子目录命名的方式与文件命名的方式相同，但由于子目录定义其他文件，因此你不能使用一般的文件指令来拷贝(COPY)或删除(DELETE)子目录。

现在分别说明有关处理子目录的指令。

1. CHDIR 指令— 改变或显示当前目录(Change Directory)

正如 DOS 会看着当前所选定的驱动器一样，它也会看着当前的目录。当启动 DOS 时，装入 DOS 的驱动器就是当前驱动器，而当前驱动器上的主目录就是当前使用的目录。就像你可以变换当前驱动器一样，你也可以用 CHDIR 指令来改变当前目录，使你每次用到别的子目录时，不必输入路径名称。CHDIR 指令的使用格式如下：

CHDIR | CD [d:][path]

CHDIR 或 CD 是 DOS 的内部指令，使用时如果没有指定驱动器名称将会改变当前所选定驱动器上的当前目录。例如，要改变驱动器 A 上的当前目录到\BLUE\MEN，可以输入：

CHDIR A:\BLUE\MEN

如果要改变当前所选定驱动器上的当前目录到根目录，可以输入：

CHDIR\

如果要显示磁盘的当前目录，可用不指定路径名称的 CHDIR 指令来完成。例如，假设当前驱动器 A 的当前目录为\BLUE\MEN，当输入：

CHDIR A:

DOS 将会显示：

CHDIR A:\BLUE\MEN

当然，要显示当前所选定驱动器上的当前目录，则只要输入：

CHDIR

即可。

2. MKDIR 指令——建立一个新的子目录(Make Directory)

MKDIR 指令可以建立一个新的子目录。其使用格式如下：

MKDIR | MD [d:][path]

MKDIR 或 MD 是 DOS 的内部指令，使用时需指定一路径名称，DOS 将会按路径最后所指定的名称建立子目录。如果没有指定驱动器名称则以当前驱动器来建立子目录。

例如想要在根目录中建立子目录 BLUE，可以输入：

MKDIR \BLUE

而要在子目录 BLUE 中建立子目录 MEN 时，可以输入：

MKDIR \BLUE\MEN

记住：如果所指定的路径未以"\"为首，甚至省略了路径，那么子目录将被建立在当前目录中。因此，如下两个指令行所建立的子目录与上例相同：

CHDIR \BLUE

MKDIR MEN

建立子目录时，只能在该目录不存在或所指定的路径名称已存在时才能建立。例如，当子目录\YELLOW 不存在时输入：

MKDIR \YELLOW\MEN

DOS 将会显示如下错误信息：

Unable to create directory

子目录建立完成后，DOS 会自动在子目录上产生两个特殊记录。例如，你想看看刚才所建立的子目录内容时，可以输入：

DIR \BLUE

屏幕将显示：

```
A:\>DIR \BLUE
Volume in drive A has no label
Directory of A:\BLUE

.          <DIR>       3-01-90   12:29a
..         <DIR>       3-01-90   12:29a
        2 File(s)     86016 bytes free

A:\>
```

每一个子目录都含有这两个特殊记录，可用来指定其他的目录。"."表示当前目录。例如，要删除当前目录中的所有文件时，可以输入：

ERASE .

就可轻易地全部删除。".."则表示当前目录的上一层目录，也就是母目录，你可以利用它回到当前目录的上一层目录。假设当前目录是\BLUE\MEN，那么你可以输入：

DIR ..

来列出母目录\BLUE 中的内容。如要列出\BLUE\WOMEN 中的内容，则可输入：

DIR ..\WOMEN

如要列出根目录中的全部内容，只要输入：

DIR ..\..

即可。

3. RMDIR 指令——删除一个子目录(Remove Directory)

RMDIR 指令可以用来删除一个子目录。其使用格式如下：

RMDIR | RD [d:][path]

RMDIR 或 RD 是 DOS 的内部指令，使用时要指定一路径，DOS 将会删除所指定路径名称中的最后一个子目录。如果没有指定驱动器名称则会删除当前驱动器上的子目录。例如，要删除子目录\BLUE 中的目录\MEN，可以输入：

RMDIR \BLUE\MEN

删除文件时要特别小心，删除目录时更要特别慎重。为了避免文件的丢失，DOS 在执行 RMDIR 指令时应遵循如下原则：

(1) 一次只能删除一个子目录，也就是删除所指定路径中最后的一个。

(2) 只有在子目录中是空的时候才能删除该子目录，也就是在目录中只有"."和".."两个特殊记录时才能删除。

(3) 不能删除根目录和当前目录。

可见，这些规则都是用来维护树状结构的完整性和保障文件的安全，以免你误将根目录和当前目录间的子目录删除掉，如果使用 RMDIR 指令时违反了这些规则，DOS 将会显示如下错误信息，以提醒用户注意：

Invalid path,not directory,

or directory not empty

如果要删除子目录中的全部文件，可以使用 ERASE 指令。例如要删除\BLUE\MEN 中的全部文件时，输入：

ERASE \BLUE\MEN*.*

即可，如果在下达 ERASE 指令时只指定路径而未指明所要删除的文件，DOS 将认为是*.*的文件，而删除该子目录中的全部文件。因此，如下的指令其效果与上例是相同的：

ERASE \BLUE\MEN

但 ERASE 指令并不能用来删除子目录，只有 RMDIR 指令才能删除子目录。

4. DELTREE 指令——删除一个子目录

DELTREE 指令是 DOS 6.0 新增的临时指令，它的功能与 RMDIR 指令相同，只是使用起来比较方便而已。其使用格式如下：

DELTREE [/T][d:][path]

DELTREE 指令可以用来删除你所指定的子目录，以及该子目录内的所有文件和子目录，而不必像 RMDIR 指令必须遵守删除的原则，才能进行删除子目录的工作。

由于 DELTREE 指令在删除子目录时，使用起来非常方便，但是为了避免文件的丢失，使用时要格外小心。因此，DELTREE 指令在删除子目录之前，都会显示提示信息来要求确认。例

如下列的操作：

```
A:\>DELTREE OTHERS
Delete directory "others" and all its subdirectories? [yn]n

A:\>DELTREE A:\*.*
Delete file "a:\io.sys"? [yn]n
Delete file "a:\msdos.sys"? [yn]n
Delete file "a:\command.com"? [yn]n
Delete file "a:\cblspace.bin"? [yn]n
Delete file "a:\autoexec.bat"? [yn]n
Delete file "a:\current"? [yn]n
Delete file "a:\help.bat"? [yn]n
Delete file "a:\about.all"? [yn]n
Delete file "a:\readme"? [yn]n
Delete file "a:\config.sys"? [yn]
Delete directory "a:\utility" and all its subdirectories? [yn]n
Delete directory "a:\helpfile" and all its subdirectories? [yn]n
Delete directory "a:\samples" and all its subdirectories? [yn]n
Delete directory "a:\others" and all its subdirectories? [yn]n
Delete file "a:\choices.bat"? [yn]n
Delete file "a:\cmenu.bat"? [yn]n
Delete file "a:\dos6.bat"? [yn]n

A:\>
```

上面的实例操作中，输入的全部都是字母 n，表示不想删除任何文件或子目录，尤其是第二个 DELTREE 指令的功能与 DEL *.* 指令是完全一样的，这是用来删除整个磁盘内容的指令。除非你确定要删除，否则不要轻易使用这个指令。

如果你使用 DELTREE 指令格式中的/T 选项，那么在删除之前就不会再显示提示信息来要求确认，使用这种方式就更加要小心了。无论如何，使用 DELTREE 指令时，一定要考虑清楚后再做。

5. MOVE 指令——不同子目录之间的文件搬移

MOVE 指令也是 DOS 6.0 新增的临时指令，它可在不同的目录间进行文件的搬移工作，省去旧版 DOS 先拷贝再删除的麻烦。其使用格式如下：

MOVE [d:][path] filename [...] destination

其中的 filename 就是你所要搬移的文件名称，而 destination 就是用来指定你所要移入的子目录名称。例如下列的指令：

MOVE \BLUE\MEN\T.TXT \TEMP

就是将\BLUE\MEN 子目录中的 T.TXT 文件搬移到\TEMP 子目录中。如果所要移入的子目录不存在，那么 DOS 就会将 T.TXT 文件拷贝到根目录下，并改名为 TEMP。

如果所要搬移的文件在一个以上，而所要移入的子目录又不存在，那么 DOS 将会显示提示信息，来询问你是否建立一个子目录以便移入文件，例如下列的实际操作：

```
C:\MOVE \BLUE\MEN\*.* \BLUE\TEST
Make directory "C:\blue\test"? [yn]y
C:\blue\men\t.txt => c:\blue\test\t.txt[ok]
C:\blue\men\tt.txt => c:\blue\test\tt.txt[ok]

C:\>
```

如果你回答"n"不建立新的子目录，DOS 便会显示如下错误信息：

Cannot move multiple files to a single file

表示多个文件不能搬移到单一的文件上。

MOVE 指令也可以用来更改子目录的名称，例如下列的指令：

MOVE \BLUE\MEN \BLUE\BOY

可以将\BLUE\MEN 子目录改名为\BLUE\BOY。子目录的改名只限定在同一层的目录区，因此类似下列的改名将造成错误：

```
C:\>MOVE \BLUE\TEST \WHITE
C:blue\test => c:\white [Unable to open source]

C:\>
```

与 COPY 指令一样，使用 MOVE 指令搬移文件时，也要注意所要移入的子目录中如有文件名与搬移的文件同名，那么这些同名的文件内容会被移入的文件覆盖，因此使用时要小心。

6. PATH 指令——指定路径以便 DOS 去寻找所需的文件

PATH 指令可用来指定一个寻找路径(Search Path)以便 DOS 去找出所要的指令文件、程序文件或批处理文件。其使用格式如下：

PATH [[d:]path[[;[d:]path]...]] | [;]

PATH 指令是 DOS 的内部指令，其使用格式看起来复杂，其实是一个非常简单的指令。你可以在一个指令内指定几个路径，各路径间只要以分号隔开即可，例如：

PATH \;\COMMANDS;A:\WANG

将会提示 DOS，如果在当前目录中无法找到所要的文件时，可以按如下的路径去寻找：

- 当前所设定驱动器上的根目录。
- 当前所设定驱动器上的子目录\COMMANDS。
- 驱动器 A 上的子目录\WANG。

如果所指定的路径有误，当 DOS 寻找到错误路径时才会发现错误。而 DOS 在寻找时也会按照路径中所指定的每一个子目录依次寻找下去，例如：

PATH \COMMANDS\EXTRAS

DOS 将会依次检查\COMMANDS 和\COMMANDS\EXTRAS。寻找时如该路径不存在，DOS 将会予以忽略而继续寻找下一个路径。到最后如在任一路径中仍未找到所要的文件，DOS 将会显示如下信息：

Bad command or filename

如果要显示当前的寻找路径，只要输入 PATH 指令而不需指定任何参数即可。假设当前所设定的驱动器为 C，当你输入：

PATH \;\COMMANDS;A:\WANG

PATH

将会显示：

PATH=C:\;\COMMANDS;A:\WANG

如要重新设定寻找路径，也就是取消前面所定义过的所有路径，只要输入：

PATH;

即可。

7. TREE 指令—— 显示磁盘上所有的路径

TREE 指令可用来显示当前所选定的驱动器上的树状文件结构，其使用格式如下：

[d:][path]TREE[d:][/F]

TREE 指令是 DOS 的临时指令，使用时可以指定驱动器的名称以便显示磁盘上的目录结构。例如：

TREE A:

如要指令中加上"/F"选项时将可显示所有路径及其文件目录，例如：

TREE A:/F

5.3 树状结构的重建

我们将存储在磁盘上的文件系统看成类似树状的结构。为了促进树状结构的操作方便，可以利用如下两个指令来达到这个效果。

1. JOIN 指令——连接两个树状结构

JOIN 指令可以把一张磁盘的文件系统连接到另外一张磁盘上，因此就可从两个不同的目录而获得单一的目录结构。JOIN 指令是 DOS 的临时指令，其使用格式如下：

[d:][path]JOIN[d: d:\directory] | [d:/D]

使用时要在第二部驱动器代号之后指定目录，而这个目录必须在根目录中，且只能有一层的深度。JOIN 指令会按你所指定的目录名称，将第一部驱动器上树状结构连接到第二部驱动器的树状结构上。如果需要的话，也可以使用虚拟驱动器(Virtual Disk)，虚拟驱动器的用法请参考第九章的介绍。

现举例说明 JOIN 指令的用法：假设有一个程序需要读取存储在硬盘上的会计数据，其文件目录为\ACCOUNTS，但实际上，这些数据文件存储在驱动器 B 上。此时，就可利用 JOIN 指令使 B:的文件好像是在 C:\ACCOUNTS 的目录中。例如：

JOIN B: C:\ACCOUNTS

此指令行将会以 C:\ACCOUNTS 的目录从 B:的树状结构连接到 C:的树状结构上。例如在 B:上有一文件名为 JANUARY，我们将以 B:JANUARY 的名称参考此文件。但只要从 B:

连接到 C:\ACCOUNTS 后就需以 C:\ACCOUNTS\JANUARY 的名称来参它。又如 B:有一文件名为\DATA\SALES，你就要以 C:\ACCOUNTS\DATA\SALES 的名称来参考这个文件。

只要一个驱动器被连接起来后，在未中断前你只能以所指定的新名称来参考它。比如上例中，你不能以 B:JANUARY 来参考 JANUARY 文件，而需以 C:\ACCOUNTS\JANUARY 的名称参考它。如果以原来的名称参考一个已经被连接起来的驱动器，DOS 将会显示如下错误信息：

Invalid drive specification

如要中断一个已被连接的驱动器，可以在驱动器名称后使用/D 参数。例如要中断 B:，可以使用：

JOIN B:/D

你如果想要知道目前有哪一部驱动器被连接起来，可以使用不带任何参数的 JOIN 指令，即：

JOIN

DOS 将会显示如下信息：

B: =〉C:\ACCOUNTS

此信息表示 B:被连接到 C:\ACCOUNTS 上。

【记住】你只能连接一部驱动器到另一部驱动器的根目录上。例如，你可以连接 B:到 C:\ACCOUNTS 上，而不能连接到 C:\ACCOUNTS\DATA 上。如果你所指定的目录并不存在，DOS 就会在所指定的磁盘中建立一个目录，但如果这个目录存在，它就必须是空的（也就是只含有"."及".."两个特殊记录），如试图以 JOIN 指令连接起来的目录并不是空的，DOS 就会显示如下错误信息：

Directory not empty

另外，所指定的目录也不能是当前目录，如果是当前目录 DOS 将会显示：

Invalid parameter

连接的状态会一直持续到你中断它为止，当然关掉主机电源或做系统复位时也会中断连接的状态。当中断一个已被连接的驱动器时并不会删除被连接上的目录，例如你已连接 B:到 C:\ACCOUNTS 目录上，当执行系统复位后 B:将被中断，但目录 C:\ACCOUNTS 将仍然存在。

【注意】当一部驱动器被连接以后，千万不要使用如下的 DOS 指令：BACKUP、RESTORE、FORMAT、DISKCOPY 及 DISKCOMP。另外也不要试图去连接一个已被使用过的 SUBST 指令的驱动器，以免造成无法检测的错误。

2. SUBST 指令——取代一路径名称

SUBST 指令可以用不同的驱动器代号来取代另外的驱动器或路径名称。其使用格式如下：

[d:][path]SUBST[d: d:][d: d:path] | [d:/D]

SUBST 指令是 DOS 的临时指令，此指令最简单的用法是以一个指定的代号来取代某一驱动器，在这种情况下只要指定两个驱动器名称即可：第一个是你想要使用的代号，而第二个

是要被取代的真正名称。例如，有一个程序必须使用到存储在C:的文件，但事实上主机根本没有安装硬盘驱动器，而且所要的文件是存储在B:上，为了解决这个问题只要下达如下的SUBST指令即可：

SUBST C: B:

下达之后，所有参考到C:的文件，DOS都会自动以存储在驱动器B上的文件来使用。

SUBST指令的另一种用法是可以让你用一个简单的驱动器代号来代表一个路径名称，这种情况下，必须在第二个参数中指定所要被取代的路径名称，而所指定的路径必须是从根目录开始。例如，当你要使用附属于B:\CATALOG\CAR\PARTS目录下的很多文件时，为了使用方便起见你可以利用如下的SUBST指令，简单的以代号E:来取代B:\CATALOG\CAR\PARTS：

SUB E: B:\CATALOG\CAR\PARTS

现在假如你参考文件名E:ENGINE，DOS将会认定是B:CATALOG\CAR\PARTS\ENGINE，这种取代方法使用起来不是很方便吗？其实，你只要回想一下，SUBST指令与上述JOIN指令的作用正好相反。

如果你想删除一个取代的作用，只要在取代名称后使用/D参数即可。比如要删除上例中的E:可以使用如下指令：

SUBST E:/D

如果下达SUBST指令时未使用任何参数时，DOS就会显示当前的取代状况。例如：

SUBST

DOS就显示如下信息：

E: =〉B:\CATALOG\CAR\PARTS

此信息表示E:是用来取代B:\CATALOG\CAR\PARTS目录的代号。

当使用SUBST指令时所指定的驱动器代号绝对不可以一样，而且所指定的第一部驱动器也不可以与当前所选定的驱动器一样，否则DOS会显示如下错误信息：

Invalid parameter

如果指定的路径不存在，DOS会显示如下的错误信息：

Path not found

取代的作用将会持续到你中断它为止，关掉主机或做系统复位也会中断取代的作用。

你可以选择字母A:到E:作为取代的符号，如果想用其他的字母(例如F:到Z:)作为代号，则需在CONFIG.SYS文件中将LASTDRIVE=E的设定值改为Z;CONFIG.SYS配置文件请参考第九章的介绍。

【注意】当SUBST指令有作用的时候，千万不要使用如下的DOS指令：BACKUP、RESTORE、FORMAT、DISKCOPY、DISKCOMP、FDISK、LABEL和JOIN。同时使用CHDIR、MKDIR、RMDIR和PATH指令时也要特别小心。

5.4　DOS分区的建立

如果你的IBM PC机安装有硬盘驱动器，在你使用它来执行DOS之前必须先以FDISK指令来建立分区，然后才能以FORMAT指令将硬盘驱动器格式化，以便将DOS

系统主磁盘拷贝到硬盘上。

完成了上述的格式化过程后，就可直接从硬盘驱动器来装入 DOS。要想从硬盘驱动器来装入 DOS，必须确定在冷启动或热启动时，驱动器 A 的门是开着的，否则系统还会从驱动器 A 上装入 DOS。

由于硬盘的空间非常大，因此在使用时可以分成一到四个独立的区域，以供不同的操作系统共享一个硬盘。为了让 DOS 能够使用硬盘，就必须为 DOS 产生一个 DOS 分区(Partition)，如果你还使用了其他的操作系统(如 OS/2、UNIX 或 ZENIX 等)，那么就应该为每一个操作系统建立一个分区。要建立一个分区可以通过 DOS 的 FDISK 指令来完成。另外 FDISK 指令也可以处理一些有关硬盘的工作，例如：

1. 建立一个分区。
2. 改变作用中的分区(Active Partition)。
3. 删除一个分区。
4. 显示各个分区的信息。

FDISK 指令是一个交互式的指令，使用时会在屏幕上显示各种功能目录及提示信息来指引用户完成工作，因此使用起来非常简单。但在建立分区时，DOS 会破坏整个磁盘的数据，所以使用时应特别小心。如果无法确定你所使用的硬盘是否已被分区，可以先通过 FDISK 指令来显示当前的状况，当确信还没有建立分区时就可进行分区的工作，以免损坏硬盘上原有的数据。

5.5 DOS 分区的格式化

当你的硬盘已建立好 DOS 分区后，就可进行 DOS 分区的格式化以便存储数据。其实，DOS 分区的格式化与软磁盘的格式化方法是一样的，都是利用 DOS 的 FORMAT 指令来进行格式化的工作。其操作过程如下：

请将 DOS 系统主磁盘放在驱动器 A 中，然后在 DOS 提示符下输入如下的 FORMAT 指令即可：

```
A:\>FORMAT C:/S/V
```

屏幕就会显示：

```
WARNING,ALL DATA ON NON-REMOVABEL DISK
DRIVE C: WILL BE LOST!
Process with Format(Y/N)? Y
```

由于格式化过程会销毁存储在 DOS 分区中的所有数据，因此在你继续工作之前必须确定是否真的要执行 FORMAT 指令。如不想格式化按 N 即可结束工作，如确定要格式化按 Y 即可进行格式化的工作，此时硬盘驱动器上的指示灯会亮起，并且在屏幕上显示 Formatting... 的信息，以提醒你正在格式化中。如果要将一部容量为 20MB 的硬盘驱动器格式化，整个过程大约要三分钟左右。当格式化完成后，屏幕会显示如下信息：

```
Formatting...Format complete
System transferred
```

```
Volume label(11 characters,ENTER for none)? HARDDISK
```

因为在下达 FORMAT 指令时使用了/S 参数，在格式化时就会把操作系统的三个文件（即 IBMBIO. COM、IBMDOS. COM 及 COMMAND. COM）从 DOS 的主磁盘拷贝到硬盘的 DOS 分区上，因此在上面的菜单中你可以看到 System transferred 的信息。接着要求你输入一个不超过 11 个字符的磁盘名称(Volume Label)，例如 HARDDISK 以作为辨认硬盘用，如不想为你的硬盘驱动器取一个磁盘名称，直接按 Enter 键就可完成格式化的工作。最后屏幕会显示有关硬盘使用空间的各种统计数据，如下所示：

```
21204992 bytes total disk space
   63488 bytes used by system
21141504 bytes available on disk

A:\>
```

在格式化过程中如发现硬盘上有损坏的地方，FORMAT 指令将会跳过这些损坏的部分，而显示如下的统计信息：

```
21204992 bytes total disk space
   63488 bytes used by system
   71680 bytes in bad sectors
21069821 bytes available on disk
```

当你看到 DOS 提示符时，就表示 DOS 分区不但已经建立完成而且也已经格式好了，你就可以准备使用这个 DOS 分区了。请输入如下的 DIR 指令显示当前硬盘的文件目录：

```
A:\>DIR C:
Volume in drive C is HARDDISK
Directory of C:\

COMMAND      COM      25307   3-17-92   12:00p
        1 file(s)     21141504 bytes free

A:\>
```

如果要把 DOS 系统主磁盘全部拷贝到硬盘的 DOS 分区上，需以如下的 COPY 指令来完成拷贝的工作：

```
A>COPY A:\*.* C:
```

当完成拷贝后，你可以先将驱动器 A 的门打开，然后再按 Ctrl+Alt+Del 以热启动的方式从硬盘装入 DOS，然后在提示符 C>的状态下以 DIR 指令来检查硬盘上的文件目录是否正确。如果一切无误，那就表示你以后可直接用硬盘来启动主机或装入 DOS 了。

5.6　硬盘的文件维护

硬盘的存储容量不但是软盘的几十倍，其存取速度更是软盘望尘莫及，因此在数据处理上

是一项必备的设备，而且在价格日渐降低的情况下，一般用户拥有一部硬盘驱动器已不是难事了。由于硬盘的存储容量非常大，其中所存储的大量数据可能都是长期日积月累而成，如果不做适当的管理和维护可能会因意外而失去原存储的数据，因此适时的为文件制作备份是需要也是必要的工作。

通常你可以把硬盘看成一张很大的软磁盘，利用 DOS 指令也可以进行拷贝(COPY)、删除(ERASE)、重新命名(RENAME)和其他与文件有关的维护工作，但是 DISKCOPY 和 DISKCOMP 指令却不能使用于硬盘，因为它们只专用于软盘。硬盘的备份工作一般都借助 BACKUP 和 RESTORE 指令来完成。

1. BACKUP 指令——从硬盘制作备份文件

BACKUP 指令是 DOS 临时指令，可用来拷贝所需的文件到指定的备份磁盘上，其使用格式如下：

[d:][path]BACKUP d:[path][filename[.ext.]]d:[/S][/M][/A]
[/D:mm-dd-yy][/T:hh:mm:ss][/F][/L[:[d:[path]filename[.ext]]]]

BACKUP 与 COPY 指令都是用来拷贝文件的指令，但 BACKUP 与 COPY 指令的差别是：(1)使用 BACKUP 指令所拷贝的备份文件不能直接使用，必须以 RESTORE 指令重新存回后才能使用。(2)BACKUP 指令会将备份磁盘上的原有文件全部删除掉，除非你使用参数/A来避免它。使用 BACKUP 指令时只要指定你所要拷贝的文件及存储备份文件的驱动器代号即可，但备份磁盘一定要事先格式化先能使用(或选用参数/F)。例如：

BACKUP C:YOUNG.XXX A:

可以将硬盘上的 YOUNG.XXX 文件拷贝到驱动器 A 上。如果使用通配符的文件名则可拷贝一组相关的文件，例如：

BACKUP C:*.* A:

就可将硬盘上的所有文件全部拷贝到驱动器 A 上，而屏幕上会逐一显示被拷贝的文件名。

使用 BACKUP 指令时可以配合如下的七个参数来完成不同的拷贝：

/S　除了要把指明的或是当前的目录中文件进行备份外，还要把该目录所属的子目录文件全部进行备份。因此，如下的指令行：

BACKUP C:*.* A:/S

就可将硬盘上的所有文件(包括全部子目录文件)拷贝到驱动器 A 上。

/M　只拷贝从上一次进行备份后，到目前为止曾经被修改过的文件。例如将硬盘 C 的全部文件进行备份之后，只修改了 YOUNG.XXX 而加入一个新的 YOUNG.YYY 文件，那么下达如下的指令行：

BACKUP C:*.* A:/M

就可拷贝 YOUNG.XXX 和 YOUNG.YYY 两文件到驱动器 A 上。

/A　指明要把将拷贝的文件全部扩展到备份磁盘的原有文件中。由于 BACKUP 指令会删除备份磁盘上的原有文件，如果只要拷贝从上一次进行备份后到目前为止曾被修改过的文件，且要保留原有备份磁盘的文件，就可利用如下指令行来完成：

BACKUP C:*.* A:/M/A

当使用 BACKUP 指令如未指明参数/A，DOS 会在屏幕上显示备份磁盘的原有文件将被删除的警告信息，以提醒你注意。

/D　用来指出在某一天或某一天以后曾经被修改过的文件，才需要进行备份，日期的格式是日－月－年(即 mm－dd－yy)都是以两位数字表示。例如：

BACKUP C:*.* A:/A/D:10－31－91

即表示要将从 1991 年 10 月 31 日以后所修改过的文件进行备份并附加到驱动器 A 上。

/T　只拷贝在指定的日期时间之后修改过的文件。例如：

BACKUP C:*.* A:/D:10－31－91/T:12:00

即表示要将从 1991 年 10 月 31 日 12 点以后所修改过的文件备份到驱动器 A 上。

/F　如果备份磁盘没有事先格式化，可以利用此参数来格式化备份磁盘。这种格式化功能是通过 FORMAT 指令来完成，因此在当前目录或指定的路径中必须含有 FORMAT 指令才能使用。例如：

BACKUP C: A:/F

/L　可用来产生一个日志文件。如果未指定文件名称，DOS 会自动设定为 BACKUP.LOG，这个日志文件会被存储在源磁盘的根目录下。如果日志文件已存在，相关信息会附加到该文件上。日志文件的第一个记录是存储备份的日期和时间，其后的记录则存储备份文件的磁盘数和完整的路径与文件名称。例如：

BACKUP C: A:/S/L

使用 BACKUP 指令时所用的备份磁盘最好事先以 DOS 格式化之后再使用，如果要将 10MB 的硬盘整盘进行备份大约需要准备 9 张已格式化的 1.2MB 磁盘，而在 BACKUP 指令进行备份时，如备份磁盘的空间已被存满，DOS 就会显示信息提醒你插入新的磁盘，以便继续拷贝。因此你应该把每一张磁盘贴上标签，并且记录备份的日期及磁盘的编号，以便重新存回去的时候可按进行备份的编号顺序插入磁盘。

如果备份磁盘是一部硬盘，备份文件就会放在一个叫做\BACKUP 的子目录中；如备份磁盘是软盘，备份文件就会存放在根目录中。但执行 BACKUP 指令时源磁盘是不能贴有写保护的，因为 BACKUP 需要记下正在进行备份的文件(/M)。

当 BACKUP 指令完成退出的时候，它会依执行的结果送出 0～4 的代码，你可以用批处理文件的 IF ERRORLEVEL 指令来测试这些代码。详细说明请看第三章的范例介绍。

2. RESTORE 指令　－从备份磁盘存回备份文件

RESTORE 指令也是 DOS 的临时指令，可用来将备份磁盘重新存回到指定的磁盘上。其使用格式如下：

[d:][path]RESTORE d:[d:][filename][.ext][/S][/P]
[/B:mm－dd－yy][/A:mm－dd－yy][/M][/N][/L:time][/E:time]

使用 RESTORE 指令时需指明所要存回的备份磁盘及文件名称，文件名称可以通配符的

方式来指定所要存回的备份文件。但要注意的是，RESTORE 指令只能存回以 BACKUP 指令进行备份的文件。

例如，要从备份磁盘 A 将文件 YOUNG.XXX 存回到硬盘 C 时，可用如下指令完成：

RESTORE A：C：YOUNG.XXX

RESTORE 指令执行时会提示你并要求插入第一张包含所要存回文件的磁盘，如果你不确定的话，不妨把备份磁盘的第一张插入驱动器内，如所指定的文件不存在，RESTORE 指令就会要求你插入第二张磁盘。

如果要将全部的备份磁盘从驱动器 A 存回到硬盘驱动器 C，则输入下列指令行即可：

RESTORE A：C：*.*

那么 RESTORE 指令执行时也会给你提示，当它把一张备份磁盘全部存回后，就会要求你插入第二张备份磁盘。

使用 RESTORE 指令时可以配合如下八个参数来达到不同的效果：

/S　　表示除了要存回所指定目录中的文件外，也包括该目录所属的全部子目录文件。例如：

RESTORE A：C：*.* /S

即表示要将磁盘 A 中的所有文件(包括全部子目录)全部存回到硬盘 C 中。

/P　　参数/P 会使得在存回那些从上一次进行备份以后已经被更正的文件之前给你一个提示(Prompt)，于是你就可以选择究竟是否要把这些文件存回来。这个参数可以避免误将已更改过的文件被取代掉。例如：

RESTORE A：C：*.* /P

其他的参数功能说明如下：

/B　　重新存回指定日期之前更改过的所有文件。

/A　　重新存回指定日期之后更改过的所有文件。

/M　　重新存回自从备份后有所更改或被删除的文件。

/N　　重新存回并不存在的磁盘的文件。

【注意】参数/B、/A、/N 不可同时使用。

/L　　重新存回指定时间之后有所更改的文件。

/E　　重新存加指定时间之前有所更改的文件。

3. MSBACKUP 指令--　制作与存回备份文件

MSBACKUP 指令是 DOS 6.0 新增的临时指令，它具有制作与存回备份文件的双重功能，而且是以交互式的窗口效果来进行的，因此使用起来比 BACKUP 和 RESTORE 指令更方便可靠。MSBACKUP 指令的使用格式为：

MSBACKUP [setup－file][/BW | /LCD | /MDA]

各选择项的功能说明如下：

setup－file　　指定备份的设定文件，如不指定则以预设的 DEFAULT.SET 为设定文件。

BW　　以黑白的色彩来显示操作窗口。

LCD　　以膝上型计算机液晶屏幕兼容的方式显示操作窗口。

MDA　　以单色屏幕来显示操作窗口。

MSBACKUP 指令只能从硬盘来执行，因此它必须存储在硬盘上。其操作方式请根据窗口的提示信息，就可进行备份文件的制作与存回的工作。

5.7　设计自己的文件系统

在你熟悉了 DOS 的基本指令和树状结构文件系统后，我们将继续介绍更高级的 DOS 指令。在深入研究时，如果你的主机安装了硬盘驱动器，请先调整一下磁盘上的文件系统，以便你操作计算机时可以更简便地使用 DOS 的指令。

1. 当你每次列出根目录时都会显示全部的 DOS 指令，为了避免这个问题，你可以将 DOS 的指令存放在一个子目录中。其操作过程如下：

(1) 先建立一子目录

C)MKDIR \DOS

(2) 再将系统主磁盘插入驱动器 A 中，输入：

C)COPY A:*.* C:\DOS

(3) 当开始操作时，只要输入：

C)PATH \DOS

那么无论当前目录是什么，都可以使用任何的 DOS 指令，但在显示根目录时就不会再显示 DOS 的指令了。

2. 如果是多人共用一部计算机时，为了避免文件混淆不清，最好赋予每人一个子目录以便存储个人的文件。例如有张、陈、林三位用户共用一部计算机，那么在根目录下就可建立\ZHANG、\CHEN 及\LIN 三个子目录提供给三人分别使用。如果用户张要使用计算机时，可以输入：

CHDIR \ZHANG

来存取他自己的文件。另外输入：

PATH \DOS

以便使用 DOS 子目录内的各个指令。

3. 如果应用程序需要用到很多的文件时，为了方便起见最好将有关的程序和数据文件全部存储到一个子目录中。就像有一个演示用的范例程序需用到五个文件，而其中一个为启动该程序的批处理文件，如将这些文件全部存储于\SAMPLE 的子目录中，那么你一开始就可输入：

PATH \DOS;\SAMPLE

以便使用 DOS 的指令(在\DOS 子目录中)及范例程序(在\SAMPLE 子目录中)。无论当前目录是什么，如果要演示范例程序只需输入该批处理文件的名称即可。

4. 如果工作文件太多，应尽量把这些文件存储在一个或两个子目录中，这样对于磁盘的拷贝工作会比较容易进行，因此你可直接以子目录来拷贝，而不必一个文件一个文件地拷贝。

第六章 数据的传输管道与过滤功能

6.1 前 言

本章首先要介绍的是一些高级的 DOS 指令，利用这些指令你就可以重新引导输入输出的流向(I/O Redirection)，以便改变标准输入设备或标准输出设备。而利用标准输入以及标准输出的数据管道(Pipe)概念，就可以让一个程序的输出作为另一个程序的输入。另外，借助过滤程序(Filter)也可以将一个程序的输入或输出排序(Sort)，或是在一个文本文件中找出所需数据，或是在屏幕上显示数据然后暂停，而提示类似—More—的信息。其次要介绍的是 DOS 5.0 和 6.0 所提供的改良型 DIR 指令，最后则是介绍如何利用 PRINT 指令来达到更高效果的打印功能。

6.2 数据输入输出的改向

当你装入 DOS 时，标准输入设备是键盘，而标准输出设备为屏幕。但是你可以利用 DOS 标准输入以及标准输出重新引导的功能，让程序从另外一个不是标准输入设备的地方取得输入数据，或是把输出数据送到一个不是标准输出设备的地方。

要重新引导标准输入输出的流向，可以在所下达的指令后加一个特殊的符号。如要重新引导标准输入，可以在所要输入的设备或文件前加上"〈"符号。例如要执行一个文件名为 YOUNG 的程序时平常只要输入：

```
YOUNG
```

程序中如需要输入数据时，就可让用户从标准输入设备的键盘来输入数据。但是，如果想要让这个程序从输入文件 INFILE 读取数据而不是从键盘输入时，就可输入：

```
YOUNG <INFILE
```

如要重新引导标准输出，可以在所要输出的设备或文件前加上")"符号，例如要直接打印磁盘目录时，可以利用如下指令将 DIR 指令的输出从屏幕改变到打印机上：

```
DIR >PRN:
```

如要将文件目录送到文件名为 DIRFILE 的文件上，可以输入：

```
DIR >DIRFILE
```

之后，便可以利用 EDIT 指令来编辑此文件或利用 TYPE 指令来显示其内容。

如果使用")"来重新引导标准输出到一个文件上，而这个文件并不存在，DOS 将会重新建立这个文件。如果文件已存在则旧有的数据将被新数据覆盖。如果你想将新数据附加到源文件的旧有数据后，可以使用")>"符号。例如下列的指令将会产生一个文件名为 DIRFILE 的文件，内含 A:，B:和 C:的磁盘目录。

```
DIR A: >DIRFILE
```

```
DIR B: )>DIRFILE
DIR C: )>DIRFILE
```

下面是一个程序范例，用来累积任何磁盘的磁盘目录到文件名为 DIRFILE 的文件上：

```
:PROGRAM NAME:COLLECT.BAT
:* ---------------------------------------------
:* Batch file to accumulate directory listings
:*
:* Get first disk ready
    ECHO OFF
    ECHO Insert the first disk into B:
    PAUSE
:*
:* Send the directory listing to the file DIRFILE
    DIR B: )DIRFILE
:*
:* Get the next disk ready
:LOOP
    ECHO Insert the next disk into B:
    ECHO Press(Ctrl+Break) to stop,or
    PAUSE
:*
:* Add the directory listing to the end of the file
    DIR B: )>DIRFILE
:*
:* Process the next disk
    GOTO LOOP
:*
:* ---------------------------------------------
```

这个批处理文件的文件名为 COLLECT.BAT，执行时要将所要累积的磁盘插入驱动器 B 中，直到按 Ctrl+Break 键结束。其执行结果如下：

```
A:\)COLLECT
A:\)      ECHO OFF
Insert the first disk into B:
Press any key to continue...    (插入第一张后按任一键)

Insert the next disk into B:
Press (Ctrl+Break) to stop,or
Press any key to continue...    (插入第二张后按任一键)

Insert the next disk into B:
Press (Ctrl+Break) to stop,or
```

```
Press any key to continue...  (按 Ctrl+Break 键停止)
^C

Terminate batch job(Y/N)? Y

A:\>
```

· 累积完毕后，可用如下的 TYPE 指令来显示各磁盘的文件目录，例如：

```
A:\>TYPE C:DIRFILE

 Volume in dirve A is ENG-DB3PLUS
 Directory of A:\

DBASE    EXE    133632   10-25-86   3:00a
DBASE    MSG     12420   01-05-87   2:48p
DBASE    OVL    266240   07-24-86   12:00p
DBASE    SER     18432   07-29-86   12:00p
DBASEINL OVL     27648   07-29-86   12:00p
       5 file(s)     458372 bytes
                     636416 bytes free

 Volume in drive A has no label
 Volume Serial Number is 281C-12D8
 Directory of A:\

COMMAND  COM     47845   06-01-91   1:00p
PE2      BAT        36   10-30-91   9:01a
PE2      EXE     73472   03-05-91   10:58a
PE2      HLP     21121   12-23-88   1:45p
PE2      PRO      3992   06-20-89   10:43a
       5 file(s)     146466 bytes
                     994304 bytes free

A:\>
```

（存储 dBASE III 的磁盘目录）

（存储 PE 的磁盘目录）

注意，程序中的第一个 DIR 指令只使用了">"符号，这是为了预防 DIRFILE 文件已经存在时会取代原有的数据。

6.3 数据的过滤功能

过滤(Filters)就是一个程序或指令，它从标准输入设备读取数据后，以某种方式加以转换，然后再把结果送到标准输出设备上。这种处理方式，我们就说数据已经被这个程序过滤了。

由于过滤程序可以不同的方式组合，因此只要使用少许过滤程序便能替代许多DOS的指令而完成复杂的工作。

DOS过滤程序包括SORT、FIND和MORE，其功能简介如下。

1. SORT过滤程序

SORT指令从标准输入设备读取数据，把这些数据按顺序排列后，再把排列结果送到标准输出设备上。其使用格式为：

[d:][path]SORT[/R][/+n]

SORT指令属于DOS的临时指令，你可排序长度不超过64K(65536字节)的任何文件。格式中如果指定/R表示以反方向排序，如果指定/+而紧跟着行数(n，即每行数据的第n字)，则表示从指定的行数开始排列数据，否则就从第一行开始排序。

SORT指令如果配合输入输出重新引导的功能就可以让DOS接受来自键盘以外地方的输入，而且把它的输入送到屏幕以外的设备上，然而你并不能同时使用同一文件作为输入和输出用。例如下列指令：

SORT〈MYFILE〉RESULT

SORT就会从MYFILE文件读取数据，排序后再把结果输出到RESULT文件上。

SORT在排序时无法区别大小写的字母，而排序是根据附录D的ASCII交换码的标准来处理。请看下面的操作范例。

在这个范例中，我们将建立两个文件，文件名为SORTTEST.1和SORTTEST.2。首先，利用简单的SORT指令来测试一下其排序的功能，请先输入：

A:\>SORT

此时，DOS将会等待你输入数据以便排序，请输入如下数据：

```
aaa
xxx
Aba
laa
^Z
```

按Ctrl+Z完成输入后，DOS便显示如下的排序结果：

```
laa
aaa
Aba
xxx

A:\>
```

请注意，数字数据放在文字数据前面，而大小写的字母被视为相同。现在，请以COPY指令建立SORTTEST.1，其文件内容为：

```
A:\>COPY CON: SORTTEST.1
ccc
abb
```

```
baa
^Z
        1 file(s) copied
A:\>
```

然后可以将此文件当成标准输入来排序，输入：

```
A:\>SORT < SORTTEST.1
```

则可显示：

```
abb
baa
ccc

A:\>
```

如果要将排序结果输出以文件名为 SORTTEST.2 的文件上，可以输入：

```
A:\>SORT < SORTTEST.1 > SORTTEST.2
```

然后可以利用如下的 TYPE 指令来显示排序的结果：

```
A:\>TYPE SORTTEST.2
abb
baa
ccc

A:\>
```

如果要以反方向来排序，则可输入：

```
A:\>SORT /R <SORTTEST.1
```

即可显示：

```
ccc
baa
abb

A:\>
```

如果要从第二行开始排序，则可输入：

```
A:\>SORT /+2 < SORTTEST.1
```

即可显示：

```
baa
abb
ccc

A:\>
```

完成上述操作后，就可利用 ERASE 或 DEL 指令将 SORTTEST.1 和 SORTTEST.2 文

件删除掉。

2. FIND 过滤程序

FIND 指令可以从你所指定的文件中寻找包含指定字符串的所有输入行，并将它送到标准输出设备上。其使用格式为：

[d:][path]FIND[/V][/C][/N]"string"[[d:][path]filename[.ext [...]

FIND 指令是 DOS 的临时指令。下例就是用来显示文件 FILE1、FILE2 和 FILE3 中包含字符串"YOUNG"的输入行：

FIND "YOUNG" FILE1 FILE2 FILE3

要注意的是，所指定的文件名不能使用通配符"*"和"?"。如果要以标准输入的方式来传送文件给 FIND 指令，则可输入：

FIND <FILE1

格式中如选/V 是用来显示所有不包含指定字符串的输入行，如选择/C 则只会显示包含指定字符串的行数，如选择/N 则会在包含指定字符串的输入行前注明这一行在文件中的行号。如果你已经选择了/N 或/V，但又选了/C，那么 FIND 指令就不理会/N 及/V 了。

所指定的字符串必须用双引号括住，如果字符串本身含有双引号，那么将两个双引号写在一起以便当作一个双引号看待。例如，要在文件 FILE1 中寻找含有"YOU" "NG"字符串的输入行，就需以如下的 FIND 指令完成：

FIND "YOU" "NG" FILE1

与 SORT 指令不同的是：FIND 指令可以区别大小写的字母，因此大写的字符串是不会吻合小写字符串的。请看下面的操作范例。

请先利用 COPY 指令建立一个文件名为 FINDTEST 的文件，其内容如下：

```
A:\>COPY CON: FINDTEST
line 1 A
line 2 a
LINE 3 A
END OF FILE
^Z
        1 file(s) copied

A:\>
```

如果想寻找所有包含"line"的输入行，可以输入：

A:\>FIND "line" FINDTEST

将会显示：

```
----------  FINDTEST
line 1 A
line 2 a

A:\>
```

注意:DOS 一般都会先显示一行注明文件名称的信息在上面,如果是以重新引导标准输入的方式传送文件给 FIND 指令,那就不会显示该信息。例如:

```
A:\>FIND "line" < FINDTEST
line 1 A
line 2 a

A:\>
```

如果要寻找所有包含"line"的输入行,并显示其行号,则可输入:

A:\>FIND /N "line" FINDTEST

将会显示:

```
---------- FINDTEST
[1]line 1 A
[2]line 2 a

A:\>
```

如果只想显示包含"line"的行数,那么就输入:

A:\>FIND /C "line" FINDTEST

将会显示:

```
---------- FINDTEST:2

A:\>
```

如果要显示不包含"line"的输入行,则可输入:

A:\>FIND /V "line" FINDTEST

则会显示:

```
---------- FINDTEST
LINE 3 A
END OF FILE

A:\>
```

可见,FIND 指令对于大写的"LINE"和小写的"line"是有所区别的,因此当你要寻找这两个字符串时就必须分别以如下的 FIND 指令来完成:

```
A:\>FIND "line" FINDTEST

---------- FINDTEST
line 1 A
line 2 a

A:\>FIND "LINE" FINDTEST
```

FINDTEST

LINE 3 A

A:\>

完成上述操作后就可利用 ERASE 或 DEL 指令将 FINDTEST 文件删除掉。

3. MORE 过滤程序

MORE 指令可以从标准输入设备读取数据，并且把整个屏幕的数据逐一的显示在标准输出设备上。其使用格式为：

[d:][path]MORE

MORE 指令是 DOS 的临时指令，当它显示一屏数据后，会暂停下来并在屏幕最后一行显示如下信息：

...More...

以提醒用户按下任一键，就可继续显示下一屏的数据，直到数据都显示完为止。

利用 MORE 指令可让你在显示一个大的文件内容时不必反复地按下 Ctrl+S 键来暂停显示，因此操作起来比较方便。请看下面的操作范例。

请先利用 COPY 指令建立一个文件名为 MORETEST 的文件，其内容为含有 100 行的相同字符串"XXXXX"。

```
A:\>EDLIN MORETEST
New file
*I
       1:*XXXXX
       2:*^C

*1,1,1,99C                    表示拷贝第一行的内容 99 次
*E

A:\>
```

文件建立完成后，可用 TYPE 指令来显示其内容，然而显示的速度相当快。如果以 MORE 指令来显示，当显示...More...信息时就可按下任一键来达到逐屏显示的效果。例如：

```
A:\>MORE < MORETEST

XXXXX
XXXXX
XXXXX
XXXXX
XXXXX
XXXXX
XXXXX
XXXXX
```

XXXXX
XXXXX
XXXXX
XXXXX
XXXXX
XXXXX
XXXXX
XXXXX
XXXXX
XXXXX

More （按任一键继续显示）

XXXXX
XXXXX
XXXXX
XXXXX
XXXXX
:
:
XXXXX
XXXXX
XXXXX
XXXXX
XXXXX
XXXXX
XXXXX

More —— （按任一键继续显示）

XXXXX
XXXXX
XXXXX
XXXXX
XXXXX

XXXXX
XXXXX
XXXXX
XXXXX
XXXXX
XXXXX
XXXXX

```
XXXXX
XXXXX
XXXXX

A:\>
```

完成上述操作后，就可利用 ERASE 或 DEL 指令将 MORETEST 文件删除掉。

4. 程序范例

为了提高程序的可读性，在设计批处理文件时都会在程序中适当地加入某些注解文字，当 DOS 在执行批处理文件时，这些注解文字虽然不被执行，但却影响 DOS 从磁盘读取批处理文件的时间，无形中延缓了执行的速度。为了避免这个问题，提供如下的程序范例以便让你在执行批处理文件之前，可先把源程序中的注解文字都"过滤"掉，然后产生一个没有注解文字的新批处理文件。

由于批处理文件中的注解行都是以":"为首，因此可以利用如下的 FIND 指令：

```
FIND /V ":" 〈TEST.BAT〉 $TEST.BAT
```

来过滤批处理文件 TEST.BAT 内的所有注解行，并产生一个文件名为 $TEST.BAT 的批处理文件。问题是上述的 FIND 指令也会把任何含有":"的输入行都过滤掉，那么程序以":"为首的标记(Label)不是也被删除掉了吗？如果改为如下 FIND 指令就可只过滤那些以":*"为首的注解行了：

```
FIND /V ":*" 〈TEST.BAT〉 $TEST.BAT
```

完整的程序如下所示：

```
:PROGRAM NAME:FILTER.BAT
:* ------------ FILTER.BAT ------------
:* Batch file to delete comments from a batch file
:*
:* Delete all the lines that contain ":*"
   ECHO Processing...%1.BAT
   FIND /V ":*" 〈%1.BAT〉 $%1.BAT
:*
:* Display message---processing is complete
   ECHO Processing...new file is: $%1.BAT
:*
:* Finished
:* ------------------------------------
```

如果要利用这个程序(其文件名为 FILTER.BAT)来过滤某一个批处理文件内的所有注解行时，就可在 FILTER 之后输入该文件名，而不必指出 BAT 的扩展名。例如，要过滤上一个范例程序 COLLECT.BAT，只要输入：

```
A:\>FILTER COLLET      (输入此指令)
```

```
A:\>      ECHO Processing... COLLECT.BAT
Processing... COLLECT.BAT


A:\>      FIND /V ":*" <COLLECT.BAT> $COLLECT.BAT

A:\>      ECHO Processing complete... new file is: $COLLECT.BAT
Processing complete... new file is: $COLLECT.BAT

A:\>
```

过滤后，可以利用 TYPE 指令来显示所产生的新文件 $COLLECT.BAT，以证实原批处理文件内的注解行已全部被删除掉：

```
A:\>TYPE $COLLECT.BAT
:PROGRAM NAME:COLLECT.BAT
    ECHO OFF
    ECHO Insert the first disk into B:
    PAUSE
    DIR B: >DIRFILE
:LOOP
    ECHO Insert the next disk into B:
    ECHO Press(Ctrl+Break) to stop,or
    PAUSE
    DIR B: >>DIRFILE
    GOTO LOOP

A:\>
```

FILTER.BAT 这个程序范例对于 BASIC 程序设计员是相当有用的，因为只要修改一下程序中的 FIND 指令，就可过滤 BASIC 程序中所有以"*"或"REM"为首的注解行，然后再执行此 BASIC 程序就可提高程序执行的速度。

6.4 数据管道

DOS 数据管道的功能是把一个程序的标准输出，当作另外一个程序的标准输入使用。因此，通过这种数据管道的功能可以自动地把标准输入与标准输出连接起来，而能够连续地执行若干程序。若要完成管道的功能，可以将一起执行的各种程序名称间用一条垂直线(|)接在一起。例如：

```
PROG1 | PROG2 | PROG3 | PROG4
```

就可逐一的执行各程序，而上一个程序的标准输出是下一个程序的标准输入(也就是每一个程序就是一个过滤程序)。

DOS 在执行管道功能时，会在根目录中建立一些临时性的文件来保存要存放到数据管道

中的输入与输出数据，由于 DOS 会自动地删除这些文件，因此用户不必在意它们是否存在。

使用 DOS 指令时如配合管道功能，可以达到更理想的效果。下面这些例子都是相当典型的管道应用。例如，当使用 DIR 指令时如文件目录太多时，可以利用如下指令：

```
DIR | MORE
```

把 DIR 指令的标准输出内容转换为过滤程序 MORE 的标准输入，而让 MORE 逐屏地把文件显示在屏幕上，那这种方法不就与 DIR/P 的功能相同了吗？如果希望把一个排序过的文件目录逐一地显示出来，可以利用下面的指令：

```
DIR | SORT | MORE
```

如果只要显示文件含"COM"的文件，指令如下：

```
DIR | FIND "COM" | MORE
```

如果要按照文件的大小来排序它们，指令如下：

```
DIR | FIND "COM" | SORT/+14 | MORE
```

由于文件目录中显示文件大小的字节数是从第 14 列开始，因此可以在 SORT 指令中选择参数/+14 来按照的大小排序。同理，当要按照文件建立的日期(按年(第 30 列)、月(第 27 列)、日(第 24 列))来排序时，就可利用如下指令：

```
DIR | SORT/+30 | SORT/+27 | SORT/+24 | MORE
```

当然，你也可以重新引导第一个程序的标准输入或最后一个程序的标准输出，例如下面的指令：

```
DIR | SORT>PRN:
```

就可将排序过的文件目录打印出来。现在，提供下列综合范例供你参考。

1. 显示文件目录时，按文件名排序：

```
:PROGRAM NAME:DIRNAME.BAT
@ECHO OF
REM Sort directory by file name
DIR SORT > ZIGNORE.ME
MORE < ZIGNORE.ME
DEL ZIGNORE.ME
```

2. 显示文件目录时，按文件大小排序：

```
:PROGRAM NAME:DIRSIZE.BAT
@ECHO OFF
REM Sort directory by file size
DIR SORT/+14 > ZIGNORE.ME
MORE < ZIGNORE.ME
DEL ZIGNORE.ME
```

3. 显示文件目录时，按建立文件的年份排序：

```
:PROGRAM NAME:DIRYEAR.BAT
@ECHO OFF
REM Sort directory by year
DIR SORT+30 > ZIGNORE.ME
```

```
MORE < ZIGNORE.ME
DEL ZIGNORE.ME
```

4. 显示文件目录时，按文件的扩展名排序：

```
:PROGRAM NAME:DIREXT.BAT
@ECHO OFF
REM Sort directory by file extension
DIR SORT +10 > ZIGNORE.ME
MORE < ZIGNORE.ME
DEL ZIGNORE.ME
```

5. 显示文件目录时，按建立文件的时间排序：

```
:PROGRAM NAME:DIRTIME.BAT
@ECHO OFF
REM Sort directory by time
DIR SORT/+35 > ZIGNORE.ME
MORE < ZIGNORE.ME
DEL ZIGNORE.ME
```

6. 显示文件目录时，按文件名排序，并且可以将显示结果存储到你所指定的文件中：

```
:PROGRAM NAME:DIROUT.BAT
@ECHO OFF
REM Sort directory by name ---disk output version
IF %1n == N GOTO NORMAL
DIR SORT > %1
ECHO Data saved in %1:
:NORMAL
DIR | SORT > ZIGNORE.ME
MORE < ZIGNORE.ME
DEL ZIGNORE.ME
```

当你执行这个范例时，如果指定一个文件名则会把显示结果存储到你所指定的文件中，否则只会在屏幕上显示结果。

7. 当你利用 TIME 指令或 DATE 指令来显示时间和日期时，DOS 将会显示类似如下信息，然后暂停以等待你输入新的时间和日期：

```
A:\>TIME
Current time is 11:38:40.18a
Enter new time:
```

由于 PC/AT 主机能自动更新时间和日期，因此对于 PC/AT 的用户而言，这个暂停是没有必要的，为此我们提供了如下 TOD.BAT(Time of Date)的范例程序，可让你在不需要暂停的情况下同时显示时间和日期。

为了设计此程序，首先要建立一个文件名为 ENTER 的文件，而其内只含有一个按Enter键的字符，例如下列的操作：

```
A:\>COPY CON:ENTER
          (在此请按一下 Enter 键)
^Z        1 file(s) copied

A:\>
```

建立完成后，就可分别利用如下的指令：

```
TIME<ENTER
DATE<ENTER
```

以 ENTER 文件为 TIME 和 DATA 指令的标准输入，因此你就可以避免自行按 Enter 键了。最后再利用管道功能将 TIME 和 DATE 指令的标准输出作为过滤程序 FIND 的标准输入，以便寻找能显示时间和日期的"Current"字符，指令如下：

```
TIME<ENTER | FIND "Current"
DATE<ENTER | FIND "Current"
```

完整的程序范例显示如下：

```
:PROGRAM NAME:TOD.BAT
:* ---------- TOD.BAT ----------
:* Batch file to display the time and date (Time of Date)
:* (The file ENTER must contain only one <ENTER> character)
   ECHO OFF
:*
:* Display the time and date
   TIME < ENTER FIND "Current"
   DATE < ENTER FIND "Current"
```

如果要显示当前的时间和日期，只要输入这个批处理文件的文件名 TOD 即可，例如：

```
A:\>TOD      (输入 TOD)
A:\>    ECHO OFF
Current time is 11:45:33.54a
Current date is Tue 05-05-1992
A:\>
```

6.5　DOS 5.0 和 DOS 6.0 的 DIR 指令

如果你使用的 DOS 版本为 5.0 或 6.0，那么在上节提供的有关 DIR 指令显示功能的范例，可以直接利用 DOS 5.0 或 6.0 的改良型 DIR 指令来完成，而不必再用设计批处理文件的方式才能解决。

新的 DIR 指令可以配合如下的选项来达到不同的功能：

DIR /B　显示当前磁盘内所有的文件名称，其他信息则不显示出来。

DIR /L　以小写字母的方式显示文件目录。

DIR /S　显示当前磁盘内所有的文件目录，包括各子目录内的全部文件。

DIR /O　以文件名的字母顺序显示文件目录。

DIR /A　以指定的属性来显示文件目录。

上列配合 DIR 指令的各选项中，O 和 A 选项还可另外指定不同的显示格式，例如 O 选项后可以配合如下参数来指定不同的排序方式，使文件目录查看起来更加方便：

N —— 按文件名的字母顺序排列

E —— 按扩展名的字母顺序排列

S —— 按文件大小由小到大排列

D —— 按建立文件日期先后排列

G —— 将子目录列于文件前

如果你想将文件目录按相反顺序列出，可以在参数前加上一个负号，例如 DIR/O:—S 表示要以文件大小由大到小的顺序排列。

另外，配合选项 A 的参数有下列五种，可以根据你所指定的属性来显示文件目录：

H —— 只显示隐藏文件

S —— 只显示系统文件

D —— 只显示子目录的名称

A —— 只显示具有保存属性的文件

R —— 只显示只读文件

如果你不想显示上列各属性的文件，只要在参数前加一个负号即可，例如 DIR/:—S 表示要显示非系统的文件，而 DIR/A:-D 则只显示文件。

配合 DIR 指令的各个选项，不但可以单独使用，也可以混合使用，以满足你的各种需求。但是，要提醒你只有 DOS 5.0 和 DOS 6.0 的版本才提供上述选项的功能。

6.6 DOS 的打印功能

DOS 提供了各种不同方式的打印功能，目前就我们所知的有如下四种方式：

1. 按 Shift+PrtSc 键来打印显示在屏幕上的数据。
2. 按 Ctrl+Shift+PrtSc 键启动打印机来打印数据。
3. 以 COPY 指令将文件内容输出到打印机上，例如：

```
COPY YOUNG PRN:
```

4. 重新引导程序或 DOS 指令的标准输出到打印机上，例如：

```
DIR A:>PRN
```

在此，我们再介绍另一种打印的功能，就是利用 DOS 的 PRINT 指令。利用 PRINT 指令来打印的好处是可以在打印的同时也可以处理其他的工作，以充分利用 PC 机的计算机资源。使用 PRINT 指令时，DOS 会先把要打印的文件建立一个打印等候队列(Print Queue)，然后在 DOS 有空时就可逐一打印等候队列中的文件内容。要注意的是，在使用 PRINT 指令时，必须有一部打印机连接在你的计算机上。

要利用这种方式来打印，先要以 PRINT 指令来建立打印等候队列，以便要打印的文件可以传送到等候队列中等待打印。建立打印等候队列的 PRINT 指令，其使用格式如下：

```
[d:][path]PRINT[/D:device][/B:buffsiz][/U:busytick]
```

[/M:maxtick][/S:timeslice][/Q:quesiz]

PRINT 指令是 DOS 的临时指令。现将格式中的各个参数说明如下：

/D　device 是用来指明打印用的设备，如果没有指定，就以设定值 PRN(打印机)为打印设备。

/B　buffsiz 是用来指出内部缓冲区(Internal buffer)的容量，单位是字节，设定的值是 512 个字节。如果把参数 B 的值加大，就可以使得 PRINT 指令的效率更高。

/Q　quesiz 是用来指明等候队列中，究竟可以容纳多少个需要打印的文件。这个值可以从 1 到 32，设定的值是 10。

/S　timeslice 是用来说明时间片(Time Slice)的值，设定的值是 8 个时间片，这里的值可以从 1 到 225。

/U　busytick 这是一个时钟单位(Clock Tick)的量，它指出在打印设备能够使用之前，PRINT 所应等待的时间。通常/U 叫做忙碌单位(Busy Tick)，它的设定值是 1。如果 PRINT 等待的时间超过 U 单位，于是就会放弃分配给它的时间。

/M　maxtick 是用来指出 PRINT 能够用来在打印设备上输出一个符号的时钟单位，/M 叫做最大单位(Maxtick)。设定的值是两个最大单位，这个值的范围可以从 1 到 225 个最大单位，在你每次使用 PRINT 指令的时候，并不一定都需要指出这个参数，你只要写一次就可以了。

当你使用 PRINT 指令的时候，DOS 就会为你在内存中建立一个打印等候队列，并显示如下信息：

Resident part of PRINT installed

PRINT queue is empty

/D、/B、/Q、/S、/U 以及/M 这几个参数只能在第一次使用 PRINT 指令时指定，如果你再次使用它们，DOS 将会忽略它们。因此，在建立好打印等候队列后，如果想要改变原设定的参数值，就必须重新装入 DOS，然后执行新设定的 PRINT 指令。当第一次执行 PRINT 指令时如未通过/D 来指定设备名称，DOS 就会显示如下信息：

Name of list device[PRN]:

来要求你指定输出用的打印设备，它包含 LPT1、LPT2、LPT3、PRN、COM1、COM2 及 AUX 等，指定设备名称时并不需要在名称后加上冒号(:)。DOS 的设定值为 PRN，因此当你直接按下 Enter 键时就设定为 PRN。

打印等候队列建立完成后，就可再利用 PRINT 指令来打印文件内容和管理打印等候队列的工作。这个 PRINT 指令的使用格式如下：

[d:][path]PRINT[/C][/T][/P][d:][path][filename][.ext]...]

要打印文件内容时，可在 PRINT 指令中指定这些文件的名称，DOS 就会把这些文件放到打印等候队列中准备打印。例如：

PRINT FILE1 FILE2 FILE3

任何在等候队列中的文件，在打印前它的内容不可以改变，也不能删除。而且存储这些文件的磁盘必须一直留在驱动器中，直到打印完为止。如果想要显示当前打印等候队列的状态，

可直接输入不包含任何参数的 PRINT 指令，也就是：

PRINT

你将会看到如下信息：

A:\FILE1 is currently being printed

A:\FILE2 is in queue

A:\FILE3 is in queue

信息的第一行表示 FILE1 文件是第一个将被打印的文件，而 FILE2 和 FILE3 文件则在队列中等候打印。如果此时打印机还没有接通电源，则会显示如下信息，提醒你注意：

Errors on list device indicate that it

may be off—line. Please check it.

如果要把打印等候队列中的某些文件清除，可以在参数/C 后指出这些要删除的文件名。例如：

PRINT /C FILE1 FILE2

就可将 FILE1 和 FILE2 文件从队列中清除。被取消打印的文件，DOS 将会显示如下信息：

FILE A:FILE1 cancelled by operator

清除文件时，队列中所有同名的文件将会被全部取消打印。例如，要打印两次文件名为 YOUNG 的文件内容时，可以输入：

PRINT YOUNG YOUNG

如果你接着输入：

PRINT /C YOUNG

DOS 会将队列中的 YOUNG 文件全部清除掉。如果想清除队列中的全部文件，只要输入：

PRINT /T

即可，DOS 也会显示如下信息：

All files cancelled by operator

参数/P 只是用来指定所要打印的某一个文件，因此，下面两个指令行的功能是相同的：

PRINT /P YOUNG

PRINT YOUNG

DOS 都会把文件 YOUNG 加到打印等候队列中。PRINT 指令一般都会打印你所指定的全部文件，除非你使用了/C 或/T 来取消打印。例如：

PRINT FILE1/C FILE2/P FILE3

DOS 会取消打印 FILE1 文件，而把 FILE2 和 FILE3 文件加到队列中等候打印。例如：

PRINT FILE1 FILE2 FILE3/C

就会把 FILE1 和 FILE2 文件加到队列中，而把 FILE3 文件清除掉。例如：

PRINT FILE1/C FILE2 FILE3

则会把队列中的 FILE1、FILE2 和 FILE3 文件清除掉。

第七章　批处理文件的设计技巧

7.1　前　言

批处理文件虽然不是一般的程序设计语言，但是在设计批处理文件时却可以像 BASIC、PASCAL 或 C 语言一样来编写程序。因此，遵守程序语法的结构化设计技巧，不但可以缩短批处理文件的设计时间，而且可以加速程序的执行速度和提高程序的可读性。

在设计批处理文件时，应遵守下列的设计技巧：

- 程序中要做充分的注解说明，以便了解其功能。
- 程序执行时要在屏幕上显示提示信息，以便掌握目前正在进行的工作。
- 以锯齿状的方式来编写程序，将有助于程序的阅读和调试。
- 程序本身要有自动查错的功能，以免终止程序的执行。

本章除了要介绍上列的技巧外，也要为你提供一些设计的要领，在你熟悉了这些技巧和要领后，一定可以设计出效率更高的好程序。

7.2　批处理文件的程序结构

如前所述，批处理文件虽然不是正规的程序设计语言，但它具有如下程序结构：

```
@ECHO OFF
REM this is a batch file structure
ECHO tell the user what is going on
...
...
:END
```

首先，以 ECHO OFF 指令作为程序的开头，然后利用 REM 和 ECHO 指令在程序中做详细的注解说明，最后以 :END 注解行作为程序的结束。你可以利用这个基本的结构逐步扩大程序，以便设计你所要的批处理文件。例如下列的程序范例就是利用这个结构设计而成的：

```
:PROGRAM NAME:SAMPLE1.BAT
@ECHO OFF
REM COPY TO PRINTER OR TO TEMP FILE
ECHO CHECK PRINT OR STORE FILE,PLEASE WAIT A MOMENT...
IF TO%1== =TOPRINT GOTO PRINTOUT
REM STORE THE FILES TO DISK
ECHO SAVING...
IF EXIST C:\BATCH\*.BAT COPY C:\BATCH\*.BAT C:\TEMP>NUL
```

```
GOTO END
:PRINTOUT
IF NOT EXIST C:\BATCH\*.BAT GOTO END
ECHO PRINTING...
FOR %%F IN (C:\BATCH\*.BAT) DO COPY %%F PRN >NUL
:END
```

这个范例程序具有拷贝文件和打印文件内容的功能，如果你想拷贝所有以 BAT 为扩展名的文件，只要输入这个批处理文件的文件名即可。如果要将每个文件的内容输出到打印机上，则必须在批处理文件的文件名后指定 PRINT 这个参数来执行即可。

上列的程序完全是按照基本的结构设计的，程序中也做了充分的注解说明，而且在屏幕上也显示适当的提示信息，最重要的是程序本身还具备了查错的功能，可以避免因输入错误而影响程序的正常执行。唯一的缺点是没有以锯齿状的方式来编写程序，使程序参考起来比较复杂，因此造成程序的调试工作不易进行。如果将程序编写成如下格式：

```
:PROGRAM NAME:SAMPLE2.BAT

@ECHO OFF
REM COPY TO PRINTER OR TO TEMP FILE
ECHO CHECK PRINT OR STORE FILE,PLEASE WAIT A MOMENT...

IF TO%1==TOPRINT GOTO PRINTOUT

    REM STORE THE FILE TO DISK
    ECHO SAVING...
    IF EXIST C:\BATCH\*.BAT COPY C:\BATCH\*.BAT C:\TEMP>NUL
    GOTO END

:PRINTOUT
    IF NOT EXIST C:\BATCH\*.BAT GOTO END
    ECHO PRINTING...
    FOR %%F IN (C:\BATCH\*.BAT) DO COPY %%F PRN >NUL
:END
```

整个程序的阅读效果就完全不一样了。这种编排方式与程序执行功能虽然毫不相关，但对于程序的参考价值却有举足轻重的影响，尤其在进行大程序的调试工作时更能发挥极大的辅助作用。

7.3 程序的查错功能

为了防止因用户的错误操作而影响程序的执行，在批处理文件中加入查错功能是一项必要的工作。例如下列用来删除存储在\TEMP 子目录中的 *.BAK 文件：

```
:PROGRAM NAME:SAMPLE3.BAT
@ECHO OFF
REM DELETE ALL .BAK FILE IN \TEMP
ECHO REMOVING ALL UGLY .BAK FILES
CD \TEMP
DEL *.BAK
ECHO ALL DONE!
:END
```

当你执行这个范例时，如果文件目录中没有\TEMP 这个子目录，则会显示 Invalid directory的错误信息，表示\TEMP 子目录不存在。这时候程序中的 CD\TEMP 指令因无法改变子目录到\TEMP 上，所以 DEL *.BAK 指令所删除的文件将是存储在当前目录中的 *.BAK 文件。显然，这已违背了原来的设想。为了避免这个问题，只要将程序改成如下即可：

```
:PROGRAM NAME:SAMPLE4.BAT
@ECHO OFF
REM DELETE ALL .BAK FILE IN \TEMP
ECHO REMOVING ALL UGLY .BAK FILES
IF NOT EXIST \TEMP\*.BAK GOTO END
    CD \TEMP
    DEL *.BAK
    ECHO ALL DONE!
:END
```

在删除文件之前，先利用 IF 指令来判断你所要改变的子目录是否存在。如果存在，才执行 CD 和 DEL 指令，否则直接结束而不会删除任何文件。

在批处理文件中，利用 IF 指令来完成查错的功能是非常方便的。但是，当你使用 IF 指令来检查用户所输入的参数时就要特别小心。例如下列的范例：

```
:PROGRAM NAME:MOVE1.BAT
@ECHO OFF
REM MOVE FILES FROM %1 TO %2
ECHO Moving Files...
COPY %1 %2
DEL %1
ECHO Files moved
:END
```

这个批处理文件的功能是用来移动你所指定的文件到指定的文件目录中。当你用如下指令来执行这个批处理文件时，屏幕会显示文件移动的情况：

```
A:\>MOVE1 C:\BATCH\T?.BAT A:
Moving Files...
C:\BATCH\T.BAT
C:\BATCH\T1.BAT
C:\BATCH\T2.BAT
```

```
C:\BATCH\T3.BAT
C:\BATCH\T4.BAT
C:\BATCH\T5.BAT
C:\BATCH\T6.BAT
        8 file(s) copied
Files moved
A:\>
```

执行这个批处理文件时，你要指定两个参数以便移动文件，如果你只指定一个参数，那么文件将被移到当前目录中。例如执行这个批处理文件时，没有指定任何参数，就会出现下列问题：

```
A:\>MOVE1
Moving Files...
Required parameter missing
Required parameter missing
Files moved
A:\>
```

为了避免这个问题，只要在程序中加入一个 IF 指令来检查是否输入参数即可。但是，一定要记得在"=="左右各加入一个字符串，以免参数%1 是空字符串时造成错误。改良后的程序显示如下：

```
:PROGRAM NAME:MOVE2.BAT
@ECHO OFF
REM MOVE FILES FROM %1 TO %2
IF %1NOTHING == NOTHING GOTO WARNING
    ECHO Moving Files...
    COPY %1 %2
    DEL %1
    ECHO Files moved
    GOTO END

:WARNING
    ECHO Please specify a source filename


:END
```

再执行这个批处理文件时，如未指定任何参数，不但会造成上述问题，而且还会自行显示下列错误信息来提醒你注意：

```
A:\>MOVE2
Please specify a source filename
A:\>
```

你是否发现，MOVE1 和 MOVE2 这两个程序范例的功能与 DOS 6.0 新增的 MOVE 指令

是一样的。

7.4　设备文件 NUL:的用途

NUL:在 DOS 中只是一个测试用的设备文件，并没有特殊的用途，虽然在操作 DOS 时很少使用到这个设备文件，但是在批处理文件中却是一个很特殊的工具。

NUL:最主要的功能是可以让你清除执行 DOS 指令时所显示的信息，例如，当你执行下列的 COPY 指令时，屏幕上就会显示执行的过程和结果：

```
A:\>COPY C:\BATCH\T?.BAT A:
C:\BATCH\T.BAT
C:\BATCH\T1.BAT
C:\BATCH\T2.BAT
C:\BATCH\T3.BAT
C:\BATCH\T4.BAT
C:\BATCH\T5.BAT
C:\BATCH\T6.BAT
        8 file(s) copied

A:\>
```

如果执行 COPY 指令时 DOS 找不到所要拷贝的文件，也会显示如下信息来提醒你注意：

```
A:\>COPY C:\BATCH\T?.BAS A:
File not found     C:\BATCH\T?.BAS
        0 file(s) copied

A:\>
```

上列这些提示信息，在 DOS 提示符下可让你掌握执行 DOS 指令的过程和结果。但是，在执行批处理文件时，如果再出现这些提示信息，可能会影响你的显示菜单，尤其是在控制良好的批处理文件中，所有的提示信息都完全由你的程序来显示，当然不希望这些 DOS 提示信息来破坏你的菜单。现在，请你将上例的两行指令修改如下，当你再执行它们时就不会出现上列的提示信息了：

```
A:\>COPY C:\BATCH\T?.BAS A:>NUL:

A:\>COPY C:\BATCH\T?.BAS A:>NUL:

A:\>
```

这就是 NUL:的功能，很理想吧！但是在批处理文件中使用 NUL:时，因为不会再显示任何提示信息，所以你一定要自行显示相关的提示信息来提醒执行的过程和结果。下列改良的 MOVE3.BAT 批处理文件就是配合 NUL:设计而成的：

```
:PROGRAM NAME:MOVE3.BAT
```

```
@ECHO OFF
REM MOVE FILES FROM %1 TO %2
IF %1NOTHING == NOTHING GOTO WARNING
    ECHO Moving Files...
    COPY %1 %2 >NUL
    DEL %1 >NUL
    ECHO Files moved
    GOTO END

:WARNING
    ECHO Please specify a source filename

:END
```

当你再执行这个批处理文件时，就只会显示如下你自行设计的提示信息了：

```
A:\>MOVE3 C:\BATCH\T?.BAT A:
Moving Files...
Files moved
A:\>
```

7.5 自动按键

在前一章的 6.4 节的范例中，曾经使用了一个自动按 Enter 键的批处理文件，以便执行时不必受到按键的影响。另外，在你执行如下的 DEL 指令时：

```
DEL *.*
```

为了安全起见，DOS 会显示如下的信息来提示你，是否确定要删除全部的文件：

```
Are you sure(Y/N)?
```

如果确定要删除按 Y，否则按 N 即可。假如你在批处理文件中使用 DEL *.* 指令，一样会显示如上的提示信息来询问你，像上例的 MOVE3.BAT 批处理文件，当你输入如下的指令来执行时：

```
MOVE3 C:\BATCH\*.* C:\BACKUP\
```

就会显示提示信息。这个信息在 DOS 提示符下可用来提醒你注意，以免无意删除全部的文件，但是在执行 MOVE3.BAT 这个批处理文件时，就有点多此一举了。因为 MOVE3.BAT 的功能是用来移动文件，当它拷贝文件到你所指定的文件目录中后，就可毫无顾虑地将源文件删除掉，所以再也不必显示提示信息了。

你也可以在批处理文件中加入一个自动按键的功能，就可避免这个困扰。现在，请你先建立如下的各文件，然后将它们加入批处理文件中就可达到自动按键的效果。

1. 自动按 Enter 键：

```
A:\>COPY CON: ENTER
        (按 Enter 键一次)
^Z
```

```
        1 file(s) copied

A:\>
```

2. 自动按 Y 键：

```
A:\>COPY CON:YES
Y          (按 Y 键一次)
^Z
        1 file(s) copied

A:\>
```

3. 自动按 N 键：

```
A:\>COPY CON:NO
N          (按 N 键一次)
^Z
        1 file(s) copied

A:\>
```

4. 自动按 Enter 键和 N 键：

```
A:\>COPY CON:ENTERN
        (按 Enter 键和 N 键各一次)
N
^Z
        1 file(s) copied

A:\>
```

当你执行下列改良后的 MOVE4.BAT 批处理文件，不但不会显示任何 DOS 的提示信息，而且还会自动按下 Y 键以便删除文件：

```
:PROGRAM NAME:MOVE4.BAT
@ECHO OFF
REM MOVE FILES FROM %1 TO %2
IF %1NOTHING == NOTHING GOTO WARNING
    ECHO Moving Files...
    COPY %1 %2 >NUL
    DEL %1 <YES >NUL
    ECHO Files moved
    GOTO END

:WARNING
    ECHO Please specify a source filename
```

```
:END
```

最后，再将 MOVE4.BAT 批处理文件加上如下检查文件是否存在的 IF 指令，就可成为一个完整无缺的程序范例：

```
:PROGRAM NAME:MOVE5.BAT
@ECHO OFF
REM MOVE FILES FROM %1 TO %2
IF %1NOTHING == NOTHING GOTO WARNING
IF NOT EXIST %1 GOTO WARNING

    ECHO Moving Files...
    COPY %1 %2 >NUL
    DEL %1 <YES >NUL
    ECHO Files moved
    GOTO END

:WARNING
    ECHO Please specify a correct source filename

:END
```

7.6 DOS 操作环境变量

当你在 DOS 提示符下输入 SET 指令时，就可显示 DOS 操作环境变量当前所设定的值。例如：

```
A>SET
COMSPEC=C:\COMMAND.COM
PATH=E:\DOS 3.3

A>
```

其中 COMSPEC 就是 DOS 用来记住 COMMAND.COM 文件存放的位置，以便装入指令处理程序时可以找到它，而 PATH 则是当前你所指定的路径名称。COMSPEC 变量的内容可以在 DOS 的提示符下利用 SET 指令来改变它。例如：

```
A>SET COMSPEC=Z:\DOS 4.0\COMMAND.COM

A>SET
PATH=E:\DOS 3.3
COMSPEC=Z:\DOS 4.0\COMMAND.COM

A>
```

可看到 COMSPEC 变量的内容已变成新的值。如果是在配置文件(请参考第九章的说明)

中，则必须利用如下的 SHELL 指令来改变它：

SHELL=C:\DOS 3.3\COMMAND.COM

而 PATH 变量的值可以利用 DOS 的 PATH 来指定它，例如：

```
A>PATH=\;C:\DOS 3.3;E:\BATCH;D:\DOS 4.0


A>SET
COMSPEC=Z:\DOS 4.0\COMMND.COM
PATH=\;C:\DOS 3.3;E:\BATCH;D:\DOS 4.0

A>
```

可看到 PATH 变量的内容已变成新的值。如果你指定了其他有关 DOS 操作环境变量的值，那么 SET 指令也会将它们显示出来，例如下列的 PROMPT 指令：

```
A>PROMPT $P$G

A:\>SET
COMSPEC=Z:\DOS 4.0\COMMAND.COM
PATH=\;C:\DOS 3.3;E:\BATCH;D:\DOS 4.0
PROMPT=$P$G

A:\>
```

除了上述有关 DOS 操作环境变量外，你也可以自行设定你所要的变量值，例如下列的 MY 变量：

```
A:\>SET my=Hello,I like batch file!

A:\>SET
COMSPEC=Z:\DOS 4.0\COMMAND.COM
PATH=\;C:\DOS 3.3;E:\BATCH;D:\DOS 4.0
PROMPT=$P$G
MY=Hello,I like batch file!

A:\>
```

自定义变量名称也是由 SET 指令来设定它的值，变量名称可以由等号左边任何的字母甚至是空格组合而成，而变量值则包括等号右边到你按 Enter 键之间的字符串。因此，为了避免混淆，等号前后最好不要输入任何空格，变量名称不管大小写，DOS 都会自动将它变成大写，但设定值不受影响。

自定义变量名称的值都是以字符串的方式来设定，因此下列的设定方式：

SET N=100

仍然是以 100 的字符串方式设定到变量 N，而不是数字 100。如果你使用下列的设定方式：

```
SET N="100"
```

那么变量 N 的设定值是"100"，包括前后的双引号。例如下列的实际操作：

```
A:\>SET N=100

A:\>SET Q="123"

A:\>SET
COMSPEC=Z:\DOS 4.0\COMMAND.COM
PATH=\;C:\DOS 3.3;E:\BATCH;D:\DOS 4.0
PROMPT=$P$G
MY=Hello,I like batch file!
N=100
Q="123"

A:\>
```

如果你想删除你所设定的变量名称，可用如下的 SET 指令来删除：

```
A:\>SET N=

A:\>SET Q=

A:\>SET MY=

A:\>SET
COMSPEC=Z:\DOS 4.0\COMMAND.COM
PATH=\;C:\DOS 3.3;E:\BATCH;D:\DOS 4.0
PROMPT=$P$G

A:\>
```

DOS 只提供 127 个字符的空间来存储你所设定的变量名称和它的值(包括等号)。因此，当你不再使用某一变量名称时，最好立刻将它删除掉，以免造成 Out of environment space(空间不足)的错误。如果你需要较大的存储空间，可以在配置文件中利用如下的 SHELL 指令来扩大：

```
SHELL=COMMAND.COM /E:n
```

其中的 n 可以是介于 160 到 32767 的数值，如果你所指定的数值小于 160 或大于 32767，DOS 会自动设定为 160 或 32767。

自定义变量主要用于批处理文件中，可用它来存储有关 DOS 操作环境的变量值，以便改变当前值后，可以再恢复它的原设定值。但是，在批处理文件中要使用变量名称的当前值时，必须在变量名称的前后加一个%符号，以便 DOS 去引用这个变量名称所存储的值。例如下列的批处理文件：

```
:PROGRAM NAME:GREET.BAT
@ECHO OFF
REM TEST ENVIRONMENT VARIABLES
SET NAME=DR.ZHANG
ECHO GLAD TO MEET YOU,%NAME%
```

当你执行这个范例时，将会显示如下结果：

```
A:\>GREET
GLAD TO MEET YOU,DR.ZHANG
A:\>
```

下列的范例程序就是利用自定义变量名称来保存 PATH 变量中的路径名称，然后将 PATH 设定成新的路径名称，最后再把 PATH 恢复为原来的路径名称：

```
:PROGRAM NAME:SAVEPATH.BAT
@ECHO OFF
SET TEMP=%PATH%
ECHO ORIGTINAL PATH:%TEMP%
PATH=C:\SYSTEM\DOS;C:\LOTUS;U:
ECHO NEW PATH:%PATH%
REM CD \LOTUS
REM 123
SET PATH=%TEMP%
ECHO NOW PATH:%PATH%
:END
```

执行结果如下：

```
A:\>SAVEPATH
ORIGINAL PATH: C:\WINDOWC;C:\WINDOWS;C:\DOS
NEW PATH: C:\SYSTEM\DOS;C:\LOTUS;U:
NOW PATH: C:\WINDOWC;C:\WINDOWS;C:\DOS
A:\>
```

可看到当前的 PATH 值又恢复为原来的路径名称。你可以将上列批处理文件的 PATH 值改成你所需要的路径名称，以便执行 LOTUS 1—2—3。当完成 LOTUS 1—2—3 的操作后，就可将 PATH 值恢复为原来所指定的路径。当你要执行 LOTUS 1—2—3 程序时，记住要把第七行和第八行的 REM 删除掉。

7.7　综合范例

提供下列的程序范例，供你参考并作为复习，以达到实用的目的。

1. 拷贝文件到你所指定的文件目录中：

```
:PROGRAM NAME:COPY2.BAT
@ECHO OFF
```

```
REM A 'Gang Copy' batch file example
IF !%1==! GOTO HELP
    SET DISK=%1
    ECHO Copy a group of files to drive %DISK%
:LOOP
SHIFT
IF !%1=! GOTO END
    COPY %1 %DISK% > NUL
    ECHO %1 copied to %DISK%
    GOTO LOOP
:HELP
ECHO COPY2 Command format:
ECHO copy2 [drive:[\path]][file1][file2]...[fileN]
:END
```

执行这个范例时，必须在批处理文件的文件名 COPY2 后，先指定文件目录然后指定你所要拷贝的文件，以便拷贝这些文件到你所指定的文件目录中。

2. 删除你所指定的文件：

```
:PROGRAM NAME:DEL2.BAT
@ECHO OFF
REM A 'Gang Delete' batch file
IF %1NOTHING == NOTHING GOTO HELP
    ECHO Delete a group of files
:LOOP
IF NOTHIGN%1 == NOTHING GOTO END
    DEL %1 (C:\SYSTEM\UTIL\YES >NUL
    ECHO %1 is/are gone...
    SHIFT
GOTO LOOP
:HELP
ECHO DEL2 Command format:
ECHO del2[file1][file2]... [fileN]
:END
```

执行这个范例时，只要在批处理文件的文件名 DEL2 之后指定所要删除的文件，就可将这些文件全部删除掉。

3. 移动文件到你所指定的文件目录中：

```
:PROGRAM NAME:MOVES.BAT
@ECHO OFF
REM A 'Gang Move' batch file
IF %NOTHING == NOTHING GOTO HELP
    SET DISK=%1
    ECHO Moving a group of files to path %DISK%
```

```
:LOOP
SHIFT
IF %1NOTHING == NOTHING GOTO END
    COPY %1 %DISK% >NUL
    DEL %1 <C:\SYSTEM\UTIL\YES >NUL
    ECHO %1 moved to %DISK%
    GOTO LOOP
:HELP
ECHO MOVE2 Command format:
ECHO move2 [drive:[\path]]「file1」[file2] ... [fileN]
:END
SET DISK=
```

执行这个范例时，必须在批处理文件的文件名 MOVES 后先指定文件目录，然后指定你所要移动的文件，以便拷贝这些文件到你所指定的文件目录中，并将源文件删除掉。你可以利用 DOS 6.0 所提供的 MOVE 指令来改良这个程序，以避免拷贝文件后的删除操作。

4. 拷贝文件到一张新的磁盘上，以作为备份：

```
:PROGRAM NAME:BACKUP2.BAT
ECHO OFF
REM BACKUP FILES
IF %1NOTHING == NOTHING GOTO HELP
IF NOTHING %1 GOTO HELP
ECHO INSERT A NEW DISK IN DRIVE A AND
PAUSE
    ECHO FORMATTING...PLEASE WAIT
    FORMAT A: <ENTER >NUL
    ECHO DISK FORMATTED AND COPYING FILES...
    COPY %1 A: >NUL
    ECHO FILES BACKUPED!
    GOTO END
:HELP
ECHO PLEASE SPECIFY A CORRECT SOURCE FILES
:END
```

执行这个范例时，只要在批处理文件的文件名 BACKUP2 后指定你所要拷贝的文件，就可将这些文件拷贝到驱动器 A 中的磁盘上。因为这个批处理文件在拷贝文件前，会自动将驱动器中的磁盘格式化，所以要小心使用。而且当前目录中也要含有 FORMAT 这个临时指令，才能正常地执行 FORMAT 指令。

5. 这个范例的功能和上例相同，只是利用 DEL 指令先将驱动器 A 中的文件全部删除掉，然后再利用 XCOPY 指令来拷贝你所指定的文件(例如 C:\BATCH*.*)：

```
:PROGRAM NAME:BACKUP3.BAT
ECHO OFF
```

```
REM BACKUP FILES
IF %1NOTHING == NOTHING GOTO HELP
IF NOT EXIST %1 GOTO HELP
    ECHO INSERT A FORMATTED DISK IN DRIVE A AND
    PAUSE
    ECHO DELETEING... PLEASE WAIT
    DEL A:\*.* (YES )NUL
    ECHO COPYING FILES...
    XCOPY %1 A: )NUL
    ECHO FILES BACKUPED!
    GOTO END
:HELP
ECHO PLEASE SPECIFY A CORRECT SOURCE FILES
:END
```

第八章　DOS 6.0 的新指令及内存管理

8.1　消除疲劳的 DOSKEY 指令

传统的 DOS 提供了一组非常特殊的编辑键，你可以利用它们在 DOS 提示符下进行命令行的编辑工作，以减少重复输入相同指令的时间。原因在于 DOS 提供了一个命令语句存储区，把你所输入的命令行自动地存储起来，然后你可以按各编辑键的用法，调出存储区内的命令行，进行修改或直接使用此命令行。

特殊编辑键的功能列表如下：

按键	编辑功能
F1 或→	从语句存储区复制一个字符到命令行上。
F2	从语句存储区中连续拷贝字符，直到用户所指定的字为止，然后把这些字符放在命令行上。
F3	把保留在语句存储区的所有字符拷贝到命令行上。
DEL	从语句存储区中跳过一个字符(此字不复制)。
F4	从语句存储区中跳跃字符，直到用户所指定字为止(该指定的字不跳过去)。
ESC	使刚刚输入的变成无效，而仍保持语句存储区不变。
INS	进入/跳出插入模式。
F5	把语句存储区清除干净，然后把剩下的命令语句存储到语句存储区中，同时屏幕跳到下一行开头处。
BACKSPACE 或←	删除命令行中的一个字符，显示光标并向左移一格，但语句存储区的内容保持不变。

命令语句存储区只能保存一个命令行，你虽可利用 DOS 所提供的特殊编辑键调出并加以修改或使用，但也只能调出你刚刚输入的最后一个命令行。显然这不是很聪明的做法。不过，DOS 5.0 和 DOS 6.0 提供了一个灵活性更强的命令行保存指令——DOSKEY。它可以存储、修改或查看你所输入的一连串命令行，甚至具有宏定义(Macro)指令的功能，使命令行的输入工作变得更容易、更轻松。DOSKEY 指令的完整使用格式为：

```
DOSKEY [/REINSTALL][/BUFSIZE=size][/MACROS]
       [/HISTORY][/INSERT | /OVERSTRIKE][/MACRONAME=[text]]
```

各选项的功能说明如下：

REINSTALL	重新设定 DOSKEY 指令。
BUFSIZE=size	设定存储命令行的缓冲区大小。
MACROS	显示当前所定义的全部宏。

HISTORY	显示当前所存储的命令行。
INSERT	重新调出命令行进行编辑时，处于插入模式。
OVERSTRIKE	重新调出命令行进行编辑时，处于修改模式。
MACRONAME=text	定义宏指令并设定一个名称。

当你下达 DOSKEY 指令后，DOS 会在内存中建立一个缓冲区，来存储你所输入的命令行，然后你可以利用↑或↓键逐一调出各行命令进行修改，以减少重新输入指令的时间。这比上面的所介绍的特殊编辑键要来得方便多了。因为通过 DOSKEY 指令你可以轻易调出先前所输入的全部命令行，找到所要的命令行后就可进行修改。当然你也可以利用 DOS 所提供的编辑键和特殊编辑键来编辑你所调出的命令行。

下达 DOSKEY 指令时，DOS 会自动保留一个 512 字节的缓存区来存储你所输入的命令行，你可以根据你的需要来改变缓冲区的大小。当 DOS 保留的缓冲区存满了时，它会采取先进先出的方式，将最先存入的命令行移出，以便挪出空间。如果你想显示当前存储在缓冲区的全部命令行，可以按 F7 键来显示。而按 Alt+F7 键则是用来删除存储在缓冲区的全部命令行。另外，你可用命令行的第一个字符来寻找你所要的命令行。当你按下所要的字母后，按下 F8 键就可列出以这个字母为首的命令行。按 F9 键则可以行首的号码来指定你所要的命令行。例如下列的操作范例：

```
A:\>(在此按 F7 键)
1: DIR *.COM
2: COPY *.? *.BAK
3: DIR *.BAK
4: DEL *.BAK
5: CLS
A:\>
```

上列指令就是在下达 DOSKEY 指令后，缓冲区所存储的全部命令行，这时候，你就可以利用↑或↓键逐一列出各个命令行，或输入行首的字母，然后按 F8 键来选用以这个字母为首的命令行，例如下列的命令行就是按字母 C 后再按 F8 键所显示出来的一行：

```
A:\>COPY *.? *.BAK
```

而下例是利用按 F9 键，以行首的号码来选用第三行的命令行：

```
1: DIR *.COM
2: COPY *.? *.BAK
3: DIR *.BAK
4: DEL *.BAK
5: CLS
A:\>(在此按 F9 键)
A:\>Line number:3
```

当找到你所要的命令行后，就可进行修改或直接按 Enter 键来执行它。但是要记住先下达 DOSKEY 指令，DOS 才会建立一个缓冲区来存储你所输入的命令行，为了避免忘记，你可以将 DOSKEY 指令放在批处理文件 AUTOEXEC.BAT 中，那么在每次装入 DOS 系统时，都会自动地执行 DOSKEY 指令。AUTOEXEC.BAT 文件将在下一章中说明。

8.2　宏指令的定义

DOSKEY 指令除了能保存及修改输入的命令行外，还能供你定义宏指令。所谓宏指令就是一组指令的集合，就像批处理文件的功能一样，用来执行特定的工作。但是宏指令存储在内存中，不像批处理文件是以文件的类型存储在磁盘上，因此宏指令执行起来速度较快，而且也没有路径的限制，不管在任何子目录下，都可以直接执行，唯一的缺点就是宏指令在关机或重新装入 DOS 后会消失，而无法像批处理文件一样可以存储到文件中持续使用。例如下列的 DOSKEY 指令就是用来定义一个名为 D 的宏指令，它可以用来显示文件名符合通配符"*.?"的文件：

```
A:\>DOSKYE D=DIR *.?
A:\>D ------ 只要下达宏名称 D 即可
A:\>DIR *.?

 Volume in drive A has no label
 Volume Serial Number is 1700-182B
 Directory of A:\

YOUNG      <DIR>          02-25-92   1:03a
TEST                 15   01-01-92   1:03p
TEST1                15   01-01-92   1:03p
TEST2                15   01-01-92   1:03p
ENTER                 2   02-26-92   11:13a
FILE1                15   01-01-92   1:03p
FILE2                15   01-01-92   1:03p
FILE3                15   01-01-92   1:03p
PLAID      <DIR>          02-27-92   11:13a
        9 file(s)          92 bytes
                       534016 bytes free

A:\>
```

定义宏时也可以利用分隔符 $T 来定义一个以上的指令，使宏指令也可以像批处理文件一样，可以连续执行多个指令。例如下列的 DOSKEY 指令就是用来定义一个含有三行指令的宏：

```
A:\>DOSKEY=DIR *.? $T COPY *.? *.BAK $T DIR *.BAK
A:\>D ------> 下达宏名称 D 即可
A:\>DIR *.?

 Volume in drive A has no label
 Volume Serial Number is 1700-182B
```

```
 Directory of A:\

YOUNG        <DIR>          02-25-92    1:03a
TEST                   15   01-01-92    1:03p
TEST1                  15   01-01-92    1:03p
TEST2                  15   01-01-92    1:03p
ENTER                   2   02-26-92   11:13a
FILE1                  15   01-01-92    1:03p
FILE2                  15   01-01-92    1:03p
FILE3                  15   01-01-92    1:03p
PLAID        <DIR>          02-27-92   11:03a
            9 file(s)          92 bytes
                          534016 bytes free

A:\>COPY *.? *.BAK
  TEST
  TEST1
  TEST2
  ENTER
  FILE1
  FILE2
  FILE3
            7 file(s) copied

A:\>DIR *.BAK

 Volume in drive A has no label
 Volume Serial Number is 1700-182B
 Directory of A:\

MIRROR     BAK         12288  02-26-92    2:48a
TEST       BAK            15  01-01-92    1:03p
TEST1      BAK            15  01-01-92    1:03p
TEST2      BAK            15  01-01-92    1:03p
ENTER      BAK             2  02-26-92   11:13a
FILE1      BAK            15  01-01-92    1:03a
FILE2      BAK            15  01-01-92    1:03a
FILE3      BAK            15  01-01-92    1:03a
            8 file(s)      12380 bytes
                          530432 bytes free

A:\>
```

DOSKEY 指令在定义宏时也可配合%1 到%9 的参数，使宏指令的功能更具有灵活性。例如下例的宏指令会将你所指定的文件拷贝成你所指定的文件名，然后再删除原先的文件：

```
A:\>DOSKEY K=COPY $1 $2 $T DEL $1

A:\>K *.BAK *.DAT <———输入宏名称，以及你所要拷贝的文件名
A:\>COPY *.BAK *.DAT
   MIRROR.BAK
   TEST.BAK
   TEST1.BAK
   TEST2.BAK
   ENTER.BAK
   FILE1.BAK
   FILE2.BAK
   FILE3.BAK
         8 file(s) copied

A:\>DEL *.BAK
A:\>
```

根据上列的执行结果，显示完全按照宏中所定义的指令来执行。除了分隔符 $T 和参数%1～%9 外，你也可以利用下表的符号，在定义宏指令时来表示这些具有特殊功能的符号：

符号	功能说明
$G	代表重新引导的>符号
GG	代表重新引导的>>符号
$L	代表重新引导的<符号
$B	代表数据管道|符号
$T	分隔指令的符号
$$	代表一个$符号
$1～$9	代表$1～$9
$*	表示在执行宏时，所要加入的其他参数

请参考下列的宏指令实例，就可明白上表各种符号的用法。

例一：将 DIR 指令所显示的结果引导到 BAKFILE 文件中。

```
A:\>DOSKEY DIRBAK=DIR *.BAK $G BAKFILE

A:\>DIRBAK
A:\>DIR *.BAK > BAKFILE

A:\>TYPE BAKFILE
```

```
 Volume in drive A has no label
 Volume Serial Number 17000-182B
 Directory of A:\

MIRROR     BAK        12288   02  26-92   2:48a
       1 file(s)      12288 bytes
                     149504 bytes free

A:\>
```

例 1:执行宏指令时可以加入所需要的其他参数。

```
A:\>DOSKEY DIRS=DIR $ *
A:\>DIRS *.COM /O:N

 Volume in drive A has no label
 Volume Serial Numberi is 1700  182B
 Directory of A:\

COMMAND      COM     47845   06-01-91    1:00p
DISKCOMP     COM     10652   06-01- 91   1:00p
DISKCOPY     COM     11793   06- 01-91   1:00p
DOSKEY       COM      5883   06-01-91    1:00p
DOSSHELL     COM      4623   04- 09-91   5:00a
EDIT         COM       413   06-01-91    1:00p
FORMAT       COM     32911   04-09 -91   5:00a
KEYB         COM     14986   04-09-91    5:00a
MIRROR       COM     18169   06-01-91    1:00p
MODE         COM     23537   04-09-91    5:00a
MORE         COM      2618   06-01 -91   1:00p
MOUSE        COM     14249   09- 28-87   8:44p
SYS          COM     13440   06-01-91    1:00p
TREE         COM      6901   06-01--91   1:00p
UNFORMAT     COM     18576   06-01-91    1:00p
       15 file(s)     226596 bytes
                     149504 bytes free

A:\>
```

如果你所定义的宏名称和 DOS 指令相同,则在输入宏名称时先输入一个空格,就表示要执行 DOS 指令,而不是要执行宏指令。例如下列的宏名称 DIR 和 DOS 的 DIR 指令:

```
A:\>DOSKEY DIR=DIR *.BAK
A:\>DIR
```

```
A:\>DIR *.BAK

Volume in drive A has no label
Volume Serial Number is 1700-182B
Directory of A:\

MIRROR     BAK      12288   02-26-92    2:48a
      1 file(s)      12288 bytes
                    518144 bytes free

A:\>    DIR TEST.DAT   (输入指令时先输入一个空格)

Volume in drive A has no label
Volume Serial Number is 1700-182B
Directory of A:\

TEST       DTA        15   01-01-92    1:03p
  1 file(s)          15 bytes
                    518144 bytes free

A:\>
```

而下列的 DOSKEY 指令则是用来显示当前你所定义的全部宏名称：

```
A:\>DOSKEY /MACROS
D=DIR *.? $T COPY *.? *.BAK $T DIR *.BAK
K=COPY $1 $2 $T DEL $1
DIR=DIR *.BAK
DIRBAK=DIR *.BAK $G BAKFILE
DIRS=DIR $*

A:\>
```

如果要删除你原先所定义的宏，只要在下达 DOSKEY 指令时不指定任何内容给指定的宏名称即可。例如：

```
A:\>DOSKEY DIR=
A:\>DOSKEY /MACROS
D=DIR *.? $T COPY *.? *.BAK $T DIR *.BAK
DIRBAK=DIR *.BAK $G BAKFILE
DIRS=DIR $*

A:\>
```

可看到宏名称 DIR 确实已被删除了。

8.3 安全可靠的保护措施

以往在操作 DOS 系统时，最让用户头疼的是必须处处小心，万一要是操作错误就无法挽回。像在执行下列的 DELTE 或 FORMAT 指令时：

```
A:\>DEL *.*
All files in directory will be deleted!
Are you sure(Y/N)? y

A:\>FORMAT A:
Insert new diskette for drive A:
and press ENTER when ready...
```

如果删除了不想删除的文件或格式化了不想格式化的磁盘时，除了生气外，谁也没有办法。这种惨痛的经历，只要操作过 DOS 的用户都有过。

不过 DOS 5.0 和 DOS 6.0 在这方面已做了改进，它提供了一套磁盘文件的保护措施，可以让你恢复已被删除的文件或恢复已被格式化的磁盘。DOS 5.0 和 DOS 6.0 所提供的保护措施就是通过 MIRROR、UNDELETE 和 UNFORMAT 三个指令来完成的。

首先为你介绍 UNDELETE 指令，这个指令可以用来恢复已被删除的文件。

1. UNDELETE ——恢复文件

文件被删除后，如果想要恢复它，可以利用 UNDELETE 指令来恢复。请参考下列的实际操作：

```
A:\>DIR TEST *

Volume in drive A has no label
Volume Serial Number is 1700-182B
Directory of A:\

TEST        15   01   01   92    1:03p
TEST1       30   01   01   92    1:05p
TEST2       30   01   01   92    1:06p
        3 file(s)         75 bytes
                     748544 bytes free

A:\>DEL TEST *
A:\>DIR TEST *

Volume in drive A has no label
Volume Serial Number is 1700-182B
Directory of A:\
```

```
File not found

A:\>
```

经过上面的操作，可看到以 TEST 为首的三个文件已全被删除掉。现在，就利用如下的 UNDELETE 指令来恢复它们：

```
A:\>C:UNDELETE TEST *

Directory of A:\
File Specifications:TEST *.*

    Deletion-tracking file not found.
    MS-DOS directory contains 3 deleted files.
    Of those, 3 files may be recovered.
Using the MS-DOS directory.

    ? EST1      30    01-01-92    1:05p ...A Undelete(Y/N)? Y
    Please type the first character for ? EST1      .  : T

File successfully undeleted.

    ? EST2      30    01-01-92    1:06p ...  A Undelete(Y/N)? Y
    Please type the first character for ? EST2      .  : T

File successfully undeleted.

    ? EST       15    01-01-92    1:03p ...A Undelete(Y/N)? Y
    Please type the first character for ? EST       .  : T

File successfully undeleted.
```

```
A:\>DIR TEST *

Volume in drive A has no label
Volume Serial Number is 1700-182B
Directory of A:\

TEST1        30    01-01-92    1:05p
TEST2        30    01-01-92    1:06p
TEST         15    01-01-92    1:03p
        3 file(s)        75 bytes
```

```
629248 bytes free

A:\>
```

经过 UNDELETE 指令处理后，这三个文件就被恢复了。

UNDELETE 指令在处理过程中，会询问你是否要恢复该文件。按 Y 表示要恢复，继续输入该文件名的第一个字母 T 就可完成恢复的过程，最后再用 DIR 指令来显示这些文件，确定都已被恢复。

当你下达 UNDELETE 指令后，DOS 首先会显示如下的一行信息：

```
Deletion—tracking file not found.
```

表示 DOS 无法找到一个记录删除信息的跟踪文件，这是因为在删除文件前未下达 MIRROR指令所致。虽然找不到跟踪文件，DOS 仍会尽力来恢复已被删除的文件，因此为了安全起见，还是建议你开机后或在执行 DELETE 或 FORMAT 指令前先执行 MIRROR 指令，来记录文件删除的情况，以便 UNDELETE 或 UNFORMAT 指令进行恢复时容易跟踪。

UNDELETE 指令的完整使用格式为：

```
UNDELETE [[d:][path]filename][/LIST | /ALL][/DOS | /DT]
```

各选项的功能说明如下：

LIST　只显示能被恢复的文件，但不会进行恢复。

ALL　恢复所有被删除的文件，但不会显示提示信息来要求确认。

DOS　利用 DOS 标准的保护模式来恢复文件，并显示提示信息来要求确认。

DT　利用 MIRROR 指令所记录的跟踪文件来恢复文件，并显示提示信息来要求确认。

提供下列的操作实例供你参考：

```
A>C:UNDELETE /LIST

Directory of A:\
File Specifications: *.*

    Deletion—tracking file not found.

    MS—DOS directory contains 3 deleted files.
    Of those, 3 files may be recovered.

Using the MS—DOS directory.

    ? EST       15   01—01—92   1:03p ...A

    ? EST1      30   01—01—92   1:05p ...A
    ? EST2      30   01  01—92   1:06p ...A

A>C:UNDELETE /ALL/DT
```

```
Directory of A:\
File Specifications: *.*

    Deletion—tracking file not found.

    MS-DOS directory contains 3 deleted files.
    Of those, 3 files may be recovered.

/DT was specified. Because there is no delete-tracking file for
this drive,the UNDELETE command cannot continue.

A>
```

配合 DT 选项的 UNDELETE 指令，表示要使用 MIRROR 指令所记录的跟踪文件来恢复已被删除的文件。如果在使用 DELETE 指令前未下达 MIRROR 指令，就会显示上面的错误信息，而且不会进行恢复的工作。

DOS 6.0 所提供的 UNDELETE 指令其保护的功能比 DOS 5.0 更加完善。它具有三种保护形式，以强化恢复文件的功能。6.0 版本的 UNDELETE 指令，其使用格式如下：

```
UNDELETE [[d:][path]filename][/DT | /DS | /DOS]
UNDELETE [/LIST | /ALL | /PURGE[:d] | /STATUS | /LOAD |
                  /UNLOAD | /S[:d] | /Td:[-n]]
```

新增的选项功能说明如下：

DS	利用删除岗哨(Delete Sentry)所记录的删除信息来恢复文件，并显示提示信息来要求确认。
PURGE	删除岗哨目录中的内容，预设值为当前目录。
STATUS	显示每一磁盘实际所使用的删除保护形式。
LOAD	用 UNDELETE.INI 文件中所定义的信息，来装入 UNDELETE 程序到内存中，如果 UNDELETE.INI 文件不存在，则使用预设值。
UNLOAD	将内存中的 UNDELETE 程序解除并关闭恢复文件的功能。
S	启动删除岗哨的保护功能，并装入 UNDELETE 程序到内存中。它将会记录删除文件的信息在指定的磁盘内，如未指定则记录在当前目录中。
T	启动删除跟踪(Delete Tracker)的保护功能，并装入 UNDELETE 程序到内存中。它将会记录删除文件的信息在指定的磁盘内，如未指定则记录在当前目录中。n 为跟踪文件的个数，其用法请参考后面 MIRROR 指令的说明。

新版 UNDELETE 指令具有三种形式的保护功能，即上述最高层次的删除岗哨保护模式、中层的删除跟踪保护模式和最低层次的 DOS 标准保护模式。5.0 版的 UNDELETE 指令只具有两种保护模式。

新的 UNDELETE 指令虽具有较强的保护功能，但是恢复被删除的文件还是亡羊补牢，经常做好文件备份的工作，才是最安全的。

2. MIRROR 指令——记录删除文件的情况

MIRROR 指令是 DOS 5.0 进行数据保护的一个指令。当你删除文件时，DOS 并非真正将数据删除，而只是在这些被删除的文件上做记号，表示这些位置是可以存储数据的地方。因此，那些被删除的文件要等到 DOS 在上面写入新的数据后，才会真正从磁盘上消失。那么 MIRROR 指令就是用来记录磁盘文件的删除情况，以便事后跟踪用。它的使用格式为：

MIRROR [d:][/Td:[-n]][/U][/l]

各选项的功能说明如下：

Td:[-n]　　用来指定你所要跟踪的驱动器代号和跟踪文件的个数。

U　　用来解除 MIRROR 的跟踪功能。

l　　只保留最后一个版本的跟踪信息。

请参考下面的操作实例：

```
A:\>MIRROR /TA
Creates an image of the system area.
Drive a being processed.
The MIRROR process was successful.
Deletion-tracking software being installed.
the following drives are supported:
Drive A  -  Default files saved.
Installation complete.
A:\>
```

首先以 MIRROR 指令建立跟踪文件后，就可使用 UNDELETE 指令来恢复被删除的文件，如下例：

```
A:\>DIR TEST *

Volume in drive A has no label
Volume Serial Number is 1700-182B
Directory of A:\

TEST        15    01-01-92    1:03p
TEST1       15    01-01-92    1:03p
TEST2       15    01-01-92    1:03p
        3 file(s)        45 bytes
                    577536 bytes free

A:\>DEL TEST *
A:\>DIR TEST *

Volume in drive A has no label
Volume Serial Number is 1700  182B
```

```
Directory of A:\

File not found

A:\>
A:\>UNDELETE /ALL/DT

Directory of A:\
File Specifications: *.*
    Searching deletion-tracking file...
        Deletion-tracking file contains 3 delted files.
        Of those, 3 files have all clusters available,
                  0 files have some clusters available,
                  0 files have no clusters available.
        MS-DOS directory contains 3 deleted files.
        Of those, 3 files may be recovered.
Using the deletin-tracking file.

    TEST        15   1-01-92   1:03p ...A Deleted:2-26-92   3:08a
File successfully undeleted.

    TEST1       15   1-01-92   1:03p ...A Deleted:2-26-92   3:08a
File successfully undeleted.

    TEST2       15   1-01-92   1:03p ...A Deleted:2-26-92   3:08a
File successfully undeleted.

A:\>
```

可以看到所有的文件已被恢复，利用下列的 DIR 指令就可确定这三个文件又出现在磁盘目录中：

```
A:\>DIR TEST *

Volume in drive A has no label
Volume Serial Number is 1700-182B
Directory of A:\

TEST        15   01-01-92   1:03p
TEST1       15   01-01-92   1:03p
TEST2       15   01-01-92   1:03p
        3 file(s)        45 bytes
                     563712 bytes free
```

```
A:\>
```

当你第一次删除文件时，MIRROR 指令会在磁盘的根目录中建立一个名为 PCTRACKR.DEL 的文件，其中记录了执行 UNDELETE 指令时所需的相关信息，以后每次删除文件时，PCTRACKR.DEL 文件都会自动更新，由于 PCTRACKR.DEL 这个文件属于 DOS 的系统文件，因此在使用 DIR 指令显示文件目录时，必须配合如下的选项/A 才能显示它：

```
A:\>DIR /A:S

Volume in drive A has no label
Volume Serial Number is 1700  182B
Directory of A:\

IBMBIO      COM    33462  06  01-92  1:00p
IBMDOS      COM    37394  06-01-92  1:00p
MIRORSAV    FIL       41  12-18-91  8:45p
PCTRACKR    DEL    13654  12-18-91  8:45p
        4 file(s)      84551 bytes
                      138752 bytes free

A:\>
```

除了 PCTRACKR.DEL 文件外，还有一个 MIRORSAV.FIL 文件也是由 MIRROR 指令所保留，内含 UNDELETE 和 UNFORMAT 指令恢复磁盘文件所需的跟踪信息。当你下达 MIRROR 指令时，它都会保留如下两个版本的信息：

MIRROR.FIL　　存储最近一版的信息

MIRROR.BAK　　存储再上一版的信息

如果在下达 MIRROR 指令时，指定了选项 1，就只保留最后一版的信息。

为了避免 PCTRACKR.DEL 文件记录的数据过多，而导致文件过大，MIRROR 指令会根据磁盘的规格预设 PCTRACKR.DEL 所能记录的文件个数，超过的部分则采取先进先出的方式覆盖原来的文件。下表就是各类磁盘的 PCTRACKR.DEL 所能记录的文件个数和所占的容量：

磁盘规格	记录的文件个数	PCTRACKR.DEL 的大小
360K	25	约 5K
720K	50	约 9K
1.2 和 1.4M	75	约 14K
20M	101	约 18K
32M	202	约 36K
32M 以上	303	约 55K

你可以根据需要自行设定 PCTRACKR.DEL 所能记录的文件个数。例如你希望驱动器 C

最多可以记录 400 个被删除的文件数据时，则可以利用下列 MIRROR 指令来设定：

MIRROR /TC－400

执行 MIRROR 指令有利于恢复被删除的文件，不过仍要小心，当你误删除文件后，应该停止任何操作，立刻执行 UNDELETE 指令来恢复文件，否则有些文件可能被覆盖掉，那就无法挽回了。

3. UNFORMAT 指令——恢复磁盘

磁盘在格式化后，其中的内容全被销毁，如果要恢复一张已经过格式化的磁盘，就要使用 UNFORMAT 指令，它的使用格式为：

UNFORMAT d:[/J][/U][/P][/L][/TEST]

各选项的功能说明如下：

J　　检查 MIRROR 指令所记录的跟踪文件。

U　　恢复磁盘时不参考 MIRROR 指令所记录的跟踪文件。

P　　将提示信息送到打印机打印。

L　　恢复磁盘时显示主目录及文件名，但不参考 MIRROR 指令所记录的跟踪文件。

TEST　模拟恢复的操作，但不会真正进行恢复的工作。

下列的操作实例就是利用 UNFORMAT 指令来恢复一张已经被格式化的磁盘：

```
A:\>UNFORMAT B:

Insert disk to rebuild in drive B:      (将所要恢复的磁盘插入驱动器 B 中后，
and press ENTER when ready.             按下 Enter 键就可开始恢复的工作)

Restores the system area of your disk by using the image file created
by the MIRROR command.

    WARNING ↓ ↓ WARNING ↓ ↓

This command should be used only to recover from the inadvertent use of
the FORMAT command or the RECOVER command. Any other use of the UNFORMAT
command may cause you to lost data ↓ Files modified since the MIRROR image
file was created may be lost.

Searching disk for MIRROR image.

The last time the MIRROR or FORMAT command was used was at 13:37 on 01-01-92

The MIRROR image file has been validated.

Are you sure you want to update the system area of your drive B(Y/N)? Y

The system area of drive B has been rebuilt.
```

```
You may need to restart the system.

A:\>
```

恢复之后，可以利用 DIR 指令来显示文件目录，以便确定是否真的恢复了。假如在使用 FORMAT 指令前忘了使用 MIRROR 指令，同样可以利用 UNFORMAT 指令来恢复磁盘，只是没有百分之百的保证能将数据恢复。因此，最安全的办法是利用批处理文件 AUTOEXEC.BAT，在每次开机装入 DOS 时自动执行MIRROR指令，以防万一。例如下面就是一个典型的 AUTOEXEC.BAT文件内容，供你参考使用：

```
@ECHO OFF
PROMPT $P$G
MOUSE
MIRROR
DOSSHELL
```

最后，再提醒你一次，虽然 DOS 5.0 和 DOS 6.0 提供了这项磁盘文件的保护措施，你还是要小心使用 DELETE 和 FORMAT 指令，因为一时的疏忽，就是再好的保护措施也有不尽人意的地方。

8.4 内存的管理

计算机内存是存放程序或数据的地方，虽然计算机处理的速度与存储器的大小无直接关系，但是存储器的大小却决定了计算机所能处理的工作量。因此内存的空间越大，计算机的处理能力也就越大。DOS 5.0 以前的版本，最让用户头疼的是，不但 DOS 本身占用了不少内存空间，而且 DOS 最多也只能控制到 640K 的内存，使软件的执行空间一直无法突破 640K 的限制。

随着应用软件的功能越来越强，而所需的存储器空间也就越大，另外，由于 IC 的价格越来越便宜，因此内存的空间也越来越大，一般 80386 主机所安装的内存都已超过 1024K(1 Mega)，为了让这些程序能够顺序地执行，DOS 5.0 和 DOS 6.0 对内存的管理做了一项重大的改进，突破了原先 640K 的限制，能够使用这些扩展的内存。

因此，对一个 DOS 5.0 和 6.0 的用户而言，除了熟悉 DOS 指令的使用方法外，了解 DOS 的内存管理也是一件非常重要的工作。首先为你介绍一般的主机所拥有的内存。

内存可分为：

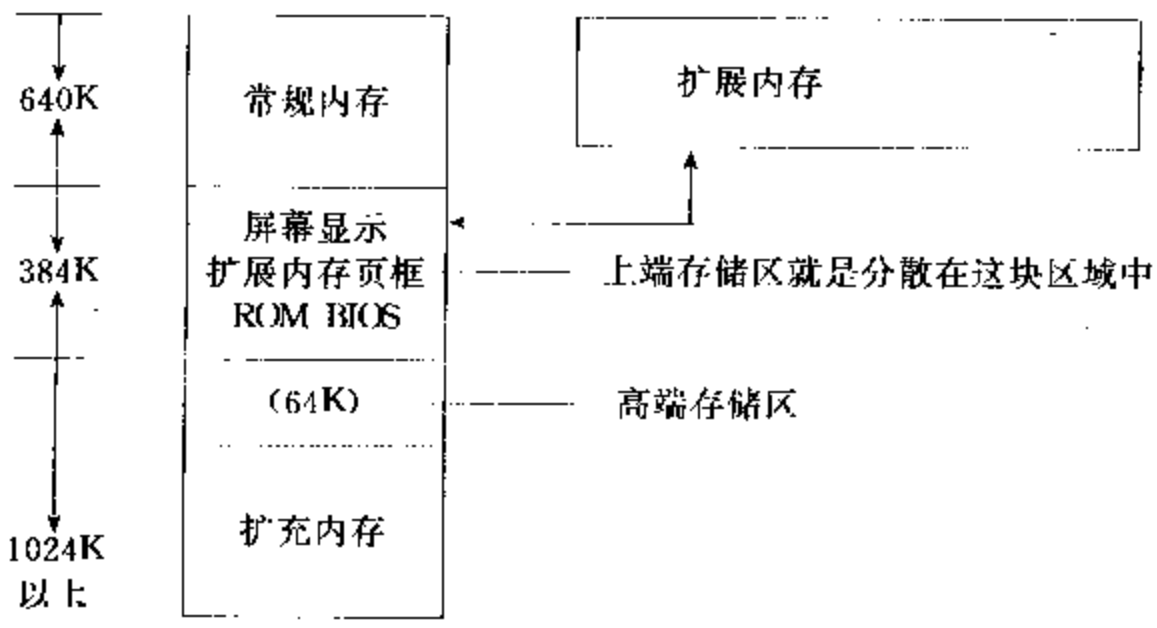
1. 常规内存(Conventional Memory)
2. 扩充内存(Extended Memory)
3. 扩展内存(Expanded Memory)

常规内存就是一般主机安装的 640K 内存，当你装入 DOS 时，DOS 会占用一部分的常规内存，而一般的程序不需特别的设定，就可直接使用常规内存。扩充内存就是主机板上超过 1024K 的部分，如果主机只有 1024K，那么常规内存以外的 384K 也就是扩充内存。DOS 5.0 以前的版本都是直接将 DOS 装入常规内存，使可用的空间变得更少，DOS 5.0 和 6.0 则可以

将本身的部分装入扩充内存的前 64K，称为高端存储区(High Memory Area，简称 HMA)。如果你的主机备有扩充内存，DOS 就可驻留在此，留给你更多的常规内存空间。当你要使用扩充内存前，必须先装入一个扩充内存的设备驱动程序，DOS 提供一个叫 HIMEM. SYS 的驱动程序可以让你在 CONFIG. SYS 配置文件中设定，以便 DOS 管理扩充内存。CONFIG. SYS 配置文件的定义方法将在下章中做详细的说明。

除了主机上的内存外，还可以在主机的扩展槽上加插内存条来扩展内存，这部分加插的内存就称为扩展内存。扩展内存除了加插的内存条，也必须配合一套管理程序才能使用，DOS 所提供的 EMM386. EXE 就是一个具有模拟及管理扩展内存功能的设备驱动程序。

DOS 5.0 和 6.0 除了支持扩充及扩展内存外，它还可以将某些程序(例如上述的设备驱动程序 HIMEM. SYS 或 EMM386. EXE)放到一个叫做上端存储区(Upper Memory Blocks，简称 UMB)的地方，以便保留更多的常规内存空间。上端存储区其实就分散在常规内存上的 384K 内存中，通常这个区域是保留给硬件(像屏幕显示或 ROM BIOS)使用，而上端存储区就是这些被占用的剩余内存。如果你拥有 80386 以上并配有扩充内存的主机，就可以利用这些区域，使用方法请参考下面的说明。为了便于了解，将各内存的相关位置列表如下供你参考：



DOS 5.0 和 DOS 6.0 既然可以保留更多的常规内存空间，那么如何才能知道目前所剩的内存空间呢？DOS 提供的 MEM 指令可以用来检查内存的使用情况。例如下列为 5.0 版的 MEM 指令所显示的信息：

```
C:\>MEM

  655360 bytes total conventional memory
  655360 bytes available to MS  DOS
  628912 largest executable program size
 7340032 bytes total contiguous extended memory
       0 bytes available contiguous extended memory
 7274496 bytes available XMS memory
         MS  DOS resident in High Memory Area
C:\>
```

从上面的信息可知，常规内存还有 628912 字节的剩余空间，而扩充内存则有 7340032 字

节(7168K)的空间,因部分的 DOS 系统被装入 64K 的高端存储区,所以扩充内存还有 7274496 字节(7104K)的剩余空间。在使用 MEM 指令时,你可以配合/P 选项来显示程序驻留在存储器的详细地址,另外你也可用/C 选项来显示当前程序使用内存的情形。因为使用/P 或/C 选项后,所显示的信息比较多,你可以利用重新引导的方式将 MEM 指令所显示的信息存储到文件中,以方便查看。例如下列的实例操作:

```
C:\>MEM/C >MEMC
C:\>TYPE MEMC
Conventional Memory:

  Name          Size in Decimal        Size in Hex
--------      ---------------------      ----------
MSDOS          12960   ( 12.7K)            32A0
HIMEM           2896   (  2.8K)            B50
COMMAND         2624   (  2.6K)            A40
DSMOUSE         7568   (  7.4K)            1D90
FREE              64   (  0.1K)            40
FREE           629024  (614.3K)            99920

Total FREE:    629024  (614.3K)

Total bytes available to programs : 629088 (614.3K)
Largest executable program size   : 628912 (614.2K)

    7340032 bytes total contiguous extended memory
          0 bytes available contiguous extended memory
    7274496 bytes available XMS memory
          MS-DOS resident in High Memroy Area
C:\>

C:\>MEM/P >MEMP
C:\>TYPE MEMP

 Address    Name          Size          Type
--------  --------  ----------  ----------------

000000                  000400  Interrupt Vector
000400                  000100  ROM Communication Area
000500                  000200  DOS Communication Area

000700    IO            000A60  System Data

001160    MSDOS         0013D0  System Data
```

```
002530      IO              0018E0      System Data
              HIMEM         000B50      DEVICE=
                            000130      FILES=
                            000100      FCBS=
                            000200      BUFFERS=
                            0001C0      LASTDRIVE=
                            000740      STACKS=
003E20      MSDOS           000040      System Program
003E70      COMMAND         000940      Program
0047C0      MSDOS           000040      Free
004810      COMMAND         000100      Environment
004920      DSMOUSE         000030      Environment
004960      DSMOUSE         001D60      Program
0066D0      MEM             000060      Environment
006740      MEM             0176F0      Program
01DE40      MSDOS           0821B0

    655360 bytes total conventional memroy
    655360 bytes available to MS-DOS
    628912 largest executable program size

   7340032 bytes total contiguous extended memory
         0 bytes available contiguous extended memory
   7274496 bytes available XMS memory
           MS-DOS resident in High Memory Area
```

上列 MEM 指令所显示的信息，就是以如下的配置文件来装入 DOS 后的内存使用情况：

```
C:\>TYPE CONFIG.SYS
DEVICE=C:\SETVER.EXE
DEVICE=C:\HIMEM.SYS
DOS=HIGH
FILES=10

C:\>
```

在装入 DOS 之前，必须确认开机磁盘上要含有 SETVER.EXE 和 HIMEM.SYS 两个文件，才不会造成错误。其中 SETVER.EXE 指令是用来设定 DOS 版本号的对照表，而 FILES 则是用来设定同时能被打开而从事文件处理的最大文件个数。

如果开机装入 DOS 时不含配置文件，那么 DOS 会被装入常规内存中，因此剩余空间只有下列的 585536 字节：

```
C:\>MEM
```

```
655360 bytes total conventional memory
655360 bytes available to MS—DOS
585536 largest executable program size

7340032 bytes total contiguous extended memory
7340032 bytes available contiguous extended memory

A:\>
```

这也就是为什么必须借助配置文件的设定，以便保留更多的常规内存空间来执行应用程序。配置文件及内存管理有关的配置命令，将在下一章做详细的说明。

DOS 6.0 在内存管理上又增强了不少功能，使常规内存的使用空间已接近 640K 的最高境界，而且可以通过内存管理程序 MEMMAKER 所提供的操作窗口，容易而有效地执行管理工作，使你在 PC 机上操作更得心应手。

因此，6.0 版的 MEM 指令功能也做了增强，以便提供更详细的信息，让你了解内存的使用情况，以及程序装入内存的相关状况。新版的 MEM 指令，其使用格式如下：

MEM [/C | /D | /F | /M:name | /P]

各选项的功能说明如下：

C(Classify)	显示程序驻留在常规内存和 UMB 的情况。
D(Debug)	显示程序和驱动程序驻留在常规内存的情况，并列出各程序的地址和类型。
F(Free)	显示常规内存和 UMB 可用的地址和剩余空间。
M(Module)	和选项/D 相同，但只显示指定的某一程序。
P(Page)	逐屏显示信息，并暂停等待按键以继续显示。

上列的选项，除了可以和/P 合用外，其余都不能合用。例如下列的操作范例与信息：

```
C:\>MEM /D /F
Too many parameters -- /F

C:\>MEM /F
Free Conventional Memory:

Setment          Total
-------  ----------  ------

0048D            80    (0K)
004A3           112    (0K)
004AA            80    (0K)
005B2         88608   (87K)
01B54        543423  (531K)

Total free: 632304 (617K)
```

No upper memory available

C:\)MEM

Memory Type	Total	Used	Free
Conventional	640K	23K	617K
Upper	0K	0K	0K
Adapter RAM/ROM	384K	384K	0K
Extended (XMS)	7168K	64K	7104K
Total memory	8192K	471K	7721K
Total under 1MB	640K	23K	617K

Largest executable program size 617K(632016 bytes)

Largest free upper memory block 0K (0 bytes)

MS DOS is resident in the high memory area.

C:\)MEM /D /P

Conventional Memory Detail:

Segment	Total	Name	Type
00000	1039 (1K)		Interrupt Vector
00040	271 (0K)		ROM Communication Area
00050	527 (1K)		DOS Communication Area
00070	2656 (3K)	IO	System Data
		CON	System Device Driver
		AUX	System Device Driver
		PRN	System Device Driver
		CLOCK $	System Device Driver
		A: - C:	System Device Driver
		COM1	System Device Driver
		LPT1	System Device Driver
		LPT2	System Device Driver
		LPT3	System Device Driver
		COM2	System Device Driver
		COM3	System Device Driver
		COM4	System Device Driver
00116	5072 (5K)	MSDOS	System Data
00253	6400 (6K)	IO	System Data
	576 (1K)	SETVERXX	Installed Device=SETVER

```
Press any key to continue... (按任一键继续)

                1136    (1K)    XMSXXXX0    Installed Device=HIMEM
                1488    (1K)                FILES=30
                 256    (0K)                FCBS=4
                 512    (1K)                BUFFERS=15
                 448    (0K)                LASTDRIVE=E
                1856    (2K)                STACKS=9,128
003E3             80    (0K)    MSDOS       System Program
003E8           2640    (3K)    COMMAND     Program
0048D             80    (0K)    MSDOS       Free --
00492            272    (0K)    COMMAND     Environment
004A3            112    (0K)    MEM         Environment
004AA             80    (0K)    MSDOS       -- Free --
004AF           4144    (4K)    DOSKEY      Program
005B2          88608   (87K)    MEM         Program
01B54         543424  (513K)    MSDOS       -- Free --
                        (部分显示未列出)
Memory Summary:

Type of Memory      Total       -      Used       +      Free
----------      -------      ---      ----      ---      ----

Conventional     655360  (640K)    23168  (23K)    632192  (617K)
Upper                 0  (0K)          0  (0K)         0  (0K)
Adapter RAM/ROM  393216  (384K)   393216  (283K)       0  (0K)
Extended (XMS)  7340032 (7168K)    65536  (64K)    7274496 (7104K)

Press any key to continue...(按任一键继续)

----------      -------      ---      ----      ---      ----

Total memory    8388608 (8192K)   481920  (417K)   7906688 (7721K)
Total under 1MB  655360  (640K)    23168  (23K)    632192  (617K)

Memory accessible using Int 15h       0       (0K)
Largest executable program size  632016       (617K)
Largest free upper memory block       0       (0K)
MS-DOS is resident in the high memory area.

XMS version 3.00; driver version 3.09
```

第九章　配置文件及启动文件

9.1　定义 DOS 的配置文件

在 DOS 的操作环境中，有许多配置是可以改变的，也就是说 DOS 对于各种环境有它预设的配置，但是这些配置可以利用一个特殊的文件来改变它。你可以在启动磁盘的根目录上建立一个文件名为 CONFIG.SYS 的配置文件，以便装入 DOS 时自动读取该文件中的命令，并以此定义系统的配置，以满足以后的操作需要。

如果装入 DOS 时并没有找到 CONFIG.SYS 配置文件，DOS 就会以预设的值(Default Value)来建立系统配置。因此，当你想要改变这些预设值时，就必须让 CONFIG.SYS 配置文件中的改变有所作用，也就是一定要重新装入 DOS 才会有效。

CONFIG.SYS 配置文件的建立，可像一般的批处理文件，利用编辑指令 EDLIN 或 EDIT 将配置命令按顺序输入文件中即可。但是，在完成文件建立的工作后，一定要重新装入 DOS 或等到下一次装入 DOS 时，你所加到配置文件的值才会产生效果。

现在，我们就来介绍这些能放到 CONFIG.SYS 文件中的 DOS 5.0 和 DOS 6.0 配置命令。

1. BREAK 命令

用来设定 Ctrl+Break 或 Ctrl+C 键检查，以便停止程序的执行，其使用格式为：

BREAK=[ON | OFF]

BREAK 的预设值为 OFF，一般 DOS 在从键盘读取数据或输入数据到屏幕或打印机时，如果你按下 Ctrl+Break 或 Ctrl+C 键，就命令停止执行。如果设定为 ON 时，则 DOS 在执行任何操作时，都会检查是否按 Ctrl+Break 或 Ctrl+C 键，如果有就立即停止执行。虽然你对 DOS 有较大的控制权，但这会把整个执行速度减慢，因为 DOS 需要更多的时间去检查你是否按 Ctrl+Break 或 Ctrl+C 键。

2. BUFFERS 命令

用来设定磁盘缓冲区的大小，以加快读写的速度。其使用格式为：

BUFFERS=X[,Y]

X 为设定磁盘缓冲区的数值，其值必须为介于 1 到 99 间的整数，而 DOS 的预设值会根据内存的大小而定。例如 640K 的内存，其预设值为 15。所谓磁盘缓冲区就是在内存中的一块保留区域，当 DOS 要从磁盘读写数据时，会先检查数据是否已经在缓冲区中，如果已经存在，DOS 就直接从缓冲区来读写，而不必再从磁盘上读写数据，这样可以加快处理的速度，因为从内存读取数据的速度要比从驱动器读取快得多。

每个磁盘缓冲区大约占用 532 个字节的内存空间，所以缓冲区如果设定太大，相对的内存空间也会被占用太多，而影响其他程序的执行。一般要执行数据库程序(例如 dBASE III+或

dBASE Ⅳ)时,缓冲区应定在 15 到 30 之间,效果较好。另外,如果主机含有 CACHE 存储器时,也可以设定 1 到 8 个缓冲区在 CACHE 上,Y 的预设值为 1。

3. COUNTRY 命令

用来设定各国的时间和日期格式,还有货币符号(例如 $,£ 等)和小数点的分隔方式(例如 3.14 或 3,14)。其使用格式为:

COUNTRY=XXX[,YYY]

XXX 代表三位数字的国家代号,设定值为 001(即美国)。YYY 代表三位数字的代码页。DOS 系统磁盘上有一 COUNTRY.SYS 文件,其中提供如下各国的代号和代码页供你选用:

国名	代号	代码页	日期格式	时间格式
美国	001	437,850	01/03/1991	5:35:00.00p
加拿大(法属)	002	863,850	1991 -01—03	17:35:00,00
拉丁美洲	003	850,437	03/01/1991	5:35:00.00p
荷兰	031	850,437	03—01—1991	17:35:00,00
比利时	032	850,437	03/01/1991	17:35:00,00
法国	033	850,437	03.01.1991	17:35:00,00
西班牙	034	850,437	03/01/1991	17:35:00,00
匈牙利	036	852,850	1991—01- 03	17:35:00,00
南斯拉夫	038	852,850	1991 01—03	17:35:00,00
意大利	039	850,437	03/01/1991	17:35:00,00
瑞士	041	850,437	03.01.1991	17:35:00,00
捷克	042	852,850	1991—01--03	17:35:00,00
英国	044	437,850	03/01/1991	17:35:00,00
丹麦	045	850,865	03—01- 1991	17:35:00,00
瑞典	046	850,437	1991—01 03	17:35:00,00
挪威	047	850,865	03.01.1991	17:35:00,00
波兰	048	852,850	1991—01--03	17:35:00,00
德国	049	850,437	03.01.1991	17:35:00,00
巴西	055	850,437	03/01/1991	17:35:00,00
英属地区	061	437,850	03/01/1991	17:35:00,00
葡萄牙	351	850,860	03 01--1991	17:35:00,00
芬兰	358	850,437	03.01.1991	17:35:00,00
中国台湾 *	088	938,850,437	1991/01/03	17:35:00,00
纽西兰 *	031	850,437	03--01—1991	17:35:00,00

国名	代号	代码页	日期格式	时间格式
日本 *	081	932,850,437	1991—01 -03	17:35:00,00
中国大陆 *	086	938,850,437	1991 01—03	17:35:00,00

注:代码页中第一个代码是 DOS 的预设码,国名右方注明 * 者为 DOS 6.0 新增的。

4. DEVICE 指令

用来设定设备驱动程序(Device Driver)的文件名称。其使用格式为:

DEVICE=[d:][path]filename[.ext]

所谓设备驱动程序,就是利用程序使 DOS 可以做一些额外的设备控制,例如外接驱动器、控制扩充内存和鼠标等。

由 DOS 提供而已被 DOS 装入的标准设备驱动程序,包括标准输入、标准输出、标准列表、软磁盘及硬盘设备等的支持服务。另外一个计时设备的驱动程序也会一起被装入内存中。基本上对于 DOS 所提供服务的各项设备而言,你并不需使用 DEVICE 命令来指定。但是为了满足一些特殊的需求,DOS 5.0 提供如下的设备驱动程序供你选用:

(1) ANSI.SYS —扩充屏幕及键盘的功能。

(2) DISPLAY.SYS —提示非英语系国家的字符显示。

(3) DRIVER.SYS —建立一个逻辑驱动器代号,以便读写实际的驱动器。

(4) EGA.SYS— 在 EGA 的屏幕上执行 Shell 介面中的 Task Swapper 功能。

(5) EMM386.EXE— 模拟扩展内存,或在 80386 以上的主机中设定上端存储区。

(6) HIMEM.SYS— 在 80286 以上的主机中设定扩展内存及高端存储区。

(7) PRINTER.SYS —提供非英语系国家的字符打印。

(8) RAMDRIVE.SYS— 利用部分的内存来模拟一部虚拟的驱动器。

(9) SETVER.EXE— 设定 DOS 版本号的对照表。

(10) SMARTDRV.SYS —将扩充内存或扩展内存设定成一个 CACHE 的快速存储区。

DOS 6.0 另外提供如下的设备驱动程序供你选用,以发挥更大的支持服务:

(1) CHKSTATE.SYS— 可通过 MEMMAKER 程序去跟踪内存最佳利用的过程。

(2) DBLSPACE.SYS—可决定 DBLSPACE.INI 文件在内存中的最后位置。

(3) INTERLNK.EXE —连接计算机主机或外围设备以便传送数据。

(4) POWER.EXE —可在计算机开机后不用时,降低电力的需求。

(5) SIZER.EXE— 可通过 MEMMAKER 程序来决定设备驱动程序和驻留程序在内存中所占的空间。

屏幕及键盘的扩充功能将在下一节做详细的说明,在此,先为你说明 DRIVER.SYS 和 RAMDRIVE.SYS 这两个驱动程序的用法,其余的驱动程序将会在介绍其他的配置命令时做进一步的说明。DRIVER.SYS 的使用格式为:

DEVICE=DRIVER.SYS /D:number [/F:factor]

其中的 number 就是你所要的驱动器名称,其范围是 0 到 127,此代码必须是你目前未使用的驱动器名称。例如你已有驱动器 A 和 B,那么就必须使用 2 号,因为 0=A、1=B、2=C,依此类推。而 factor 则可根据下表来选用:

值	软盘驱动器类型
0	160/180K
0	180/360K
1	1.2M
2	720K(3.5 英寸)
7	1.4M(3.5 英寸)
9	2.88M(3.5 英寸)

例如要外加一部 720KB 的 3.5 英寸的软盘驱动器时，应在配置文件中定义 DEVICE 命令如下：

DEVICE=C:\DRIVER.SYS/D:2/F:2

RAMDRIVE.SYS 这个设备驱动程序可以利用一部分的内存来模拟一部驱动器，而成为所谓的虚拟磁盘(Virtual Disk)。使用虚拟磁盘的方法与一般真正的磁盘完全相同，但使用虚拟磁盘的最大好处是存取速度非常快，因为它的操作速度是内存的速度，因此比真正的磁盘的存取速度快得多。然而在使用虚拟磁盘时也要注意，由于虚拟磁盘会占用一部分的内存，当你重新装入 DOS 或把电源关掉时，虚拟磁盘中的内容就会消失，因此在使用虚拟磁盘之后务必把所存储的文件拷贝到真正的磁盘上存储起来。

你如果要建立虚拟磁盘，只要 CONFIG.SYS 文件中以 DEVICE 命令来指定 RAMDRIVE.SYS 这个设备驱动程序即可，其使用格式如下：

DEVICE=[d:][path]RAMDRIVE.SYS[bbb][sss][ddd][/E][/A]

其中 bbb 是指虚拟磁盘的容量，单位是 K 字节，如未指定或指定的容量超过你的计算机所能使用的内存，DOS 就以 64K 为设定值。如果你所指定的容量建立起来后，会使内存少于 64K，那么 DOS 就会调整其容量(也就是会把虚拟磁盘的容量减少)。

sss 是指扇区(Sector)的容量，单位是 K 个字节，能够使用的值是 128、256 或 512，如未指定，DOS 就以 512 为设定值。而 ddd 是指虚拟磁盘用来存储文件的数目，这个值的范围可以从 2 到 512，如未指定，DOS 就以 64 为设定值。

/E 是用来要求 RAMDRIVE.SYS 使用扩充内存，而/A 则使用扩展内存来充当虚拟驱动器，如果未指定/E 或/A，RAMDRIVE.SYS 会以常规内存来充当虚拟驱动器。但要注意的是，/E 或/A 只对加有扩充内存或扩展内存的主机才有效。如果你在未装有这些内存的主机上使用了这个选项，DOS 就会显示错误的信息并且不会安装虚拟驱动器。

在 RAMDRIVE.SYS 安装过程中，如果 DOS 认为你所指定的 sss 值(即磁盘容量)或 ddd 值(即目录栏位)需要调整的话，就会自动地调整它，并显示如下信息：

Sector size adjusted 或

Directory entries adjusted

5. DEVICEHIGH 命令

用来装入设备驱动程序到上端存储区。其使用格式为：

DEVICEHIGH=[d:][path]filename[.ext]

在 80386 以上的主机中，可以将一些设备驱动程序放到高端存储区，以节省常规内存。但是在设定 DEVICEHIGH 之前，要先设 DEVICE＝HIMEM. SYS、DOS＝UMB 和 DEVICE＝EMM386. EXE，否则设备驱动程序仍会放在常规内存中。例如下列的命令就是将 MOUSE. SYS 这个驱动程序放到高端存储区中。

```
DEVICE=C:\DOS\HIMEM. SYS
DOS=UMB
DEVICE=C:\DOS\EMM386. EXE NOEMS
DEVICEHIGH=C:\DRIVER\MOUSE. SYS
```

如果高端存储区没有足够的空间，那么 DOS 就自动将程序装入常规内存中。上例 DEVICE 命令行中的 NOEMS 表示只使用高端存储区，而不模拟扩展内存，如果二者都要，则可以改用 RAM 选项。

6. DOS 命令

用来将部分的 DOS 放在高端存储区，以及作为常规内存与上端存储区的联系。其使用格式为：

```
DOS=HIGH | LOW[,UMB | ,NOUMB]
```

或

```
DOS=[HIGH, | LOW,]UMB | NOUMB
```

DOS 的预设值为 LOW 和 NOUMB，HIGH 表示要将部分的 DOS 放到高端存储区，而 LOW 则是将 DOS 全部放到常规内存中。UMB 表示要使用上端存储区，而 NOUMB 则不使用。如果你选用 HIGH 或 UMB，则在选用前必须先以 DEVICE 设定 HIMEM. SYS，而在选用 UMB 后，也要以 DEVICE 设定 EMM386. EXE。例如下列的命令就是将部分的 DOS 装入高端存储区，并且把 MOUSE. SYS 驱动程序放到上端存储区中：

```
DEVICE=C:\DOS\HIMEM. SYS
DOS=HIGH,UMB
DEVICE=C:\DOS\EMM386. EXE NOEMS
DEVICEHIGH=C:\DRIER\MOUSE. SYS
```

7. DRIVEPARM 命令

用来设定驱动器的参数以便取代当前系统中的定义。其使用格式为：

```
DRIVEPARM=/d:number
```

这个命令一般的用户很少使用，你如果需要可以请教为你装机的计算机厂商。

8. FCBS 命令

用来指定能够同时被 FCBS(File Control Blocks，文件控制表)打开的文件数目。其使用格式为：

```
FCBS=m,n
```

m 是指一次能被 FCBS 打开的文件总数，它的设定值为 4，允许的范围是 1 到 255。如果一个程序经常同一时刻用 FCBS 打开的文件数目超过 m 值，那么 DOS 会自动地关闭某些被

FCBS 打开的文件，而 n 就是指出 DOS 不可以自动关闭的文件数目。因此，当指定了 n 后，被 FCBS 打开的文件的前 n 个就会被保护起来，而不会被 DOS 关闭。n 的设定值为 0，允许的范围是 0 到 255。

较早期的应用程序是以 FCBS 的方式来建立、打开、删除或读写文件，现在新的程序已不再用这种方法打开文件，因此只有程序需要你设定 FCBS 时才设，否则用下面的 FILES 命令来设定所能打开的文件即可。

9. FILES 命令

用来指定同时能被打开而进行文件处理的最大文件个数。其使用格式为：

FILES=X

X 值的允许范围是 8 到 255，预设值为 8。由于 DOS 会保留 5 个文件区来管理 DOS 所预设的标准输入输出设备。例如键盘、屏幕、打印机等，因此当使用预设值时，就只能打开 3 个文件。当你执行程序时，如果显示"Too Many Files Are Open"的错误信息，就必须增加 FILES 的数目，一般都设在 15 到 30 之间。

10. INSTALL 命令

用来装入下列的临时指令到内存中，以便往后变成内部指令来使用。

FASTOPEN.EXE

KEYB.COM

NLSFUNC.EXE

SHARE.EXE

其中 SHARE 指令使用于网络上，NLSFUNC 指令用来提供非英语系国家的 DOS 支持，KEYB 指令用来选择键盘的类型，而 FASTOPEN 指令则用来快速存取所要的文件。INSTALL的使用格式如下：

INSTALL filename

filename 就是上面四个指令中的一个，例如：

INSTALL KEYB.COM US

INSTALL FASTOPEN.EXE C:

其实，这些指令也可以在 DOS 提示符下直接执行，或是通过批处理文件来执行。但是利用 INSTALL 命令来装入它们而变成常驻程序时，并不会建立该程序的环境变量区块，所以程序占的内存空间会比直接执行或由批处理文件执行要少。

11. LASTDRIVE 命令

用来指定所能使用到的驱动器的最大值。其使用格式为：

LASTDRIVE=X

X 值可以是 A～Z 间的任一字母，它用来表示 DOS 能够接受的最后一个正确的驱动器代号，预设值为当前主机使用代号的下一个。至于 LASTDRIVE 所能定出来的最小值，就是你在主机上已安装好的驱动器数目，如果 X 的值小于主机中实际的驱动器数目，DOS 将不理会配置文件中的这个命令。

12. NUMLOCK 命令

它是DOS 6.0 新增的命令，可用来指定 Num Lock 键在开机时是开或关的状态。其使用格式为：

NUMLOCK=[ON | OFF]

预设值为 ON。要特别注意的是这个命令只能在配置文件中使用。

13. REM 命令

用来加入注解文字到配置文件中，其使用格式为：

REM [comment]

其中 comment 就是你所要注解的说明文字。该注解文字不会被当作命令执行。

14. SHELL 命令

用来指定 COMMAND.COM 文件的路径名称，以便装入 DOS 时可以找到这个文件。另外，你也可以利用这个指令来指出其他处理程序的文件名称及路径，以便装入 DOS 时可以把它存入内存中，用来取代 COMMAND.COM 这个文件。因为内部指令及批处理文件指令是存放在 COMMAND.COM 文件中，所以当你使用了 COMMAND.COM 以外的指令处理程序时，就无法再使用这些指令了，除非在你的指令处理程序中也包括这些指令。这也就是为什么 COMMAND.COM 文件一般都存放在开机磁盘的根目录下，以便装入 DOS 时可以找到它，如果你使用 SHELL 命令改变了存放 COMMAND.COM 文件的路径，一定要确定开机磁盘上含有这个路径及 COMMAND.COM 文件，才不会造成错误，其使用格式为：

SHELL=[d:][path]filename[parameters]

15. STACKS 命令

用来改变系统堆栈(Stack)。其使用格式为：

STACK=n,s

其中 n 为堆栈的个数，而 s 则为堆栈的大小。当系统发生硬件中断时，DOS 会设置一个堆栈给这个中断信号，以便存储那些被中断执行的程序数据，等到中断处理完毕后再将数据取出。n 的数值可以是从 8 到 64，而 s 是以字节为单位，可以是 32 到 512。在没有指定的情况下，IBM PC、IBM PC/XT 和 IBM 便携机的预设值为 n=0,s=0，而其他的机型则为 n=9,s=128。

16. SWITCHES 命令

用来强迫增强型的键盘如同一般的键盘功能。其使用格式为：

SWITCHES=/K

有些应用程序在增强型的键盘上不能正确使用时，可以利用这个命令使一个增强型的键盘当作传统型的键盘来使用。除了选项/K 外，DOS 6.0 还新增了如下功能的选项：

W	如要执行 Windows 3.0 增强模式时，可将 WINA20.386 文件移到根目录以外的子目录中。
N	可防止用户按功能键 F5 或 F8 来避开启动指令的执行。

F　　在开机显示"Starting MS—DOS..."信息后，不要延迟 2 秒钟的时间。

综合上述的配置命令，提供你一个典型的 CONFIG.SYS 配置文件如下：

```
BUFFERS=20
FILES=15
BREAK=ON
SHELL=A:\YOUNG\COMMAND.COM
INSTALL=FASTOPEN.EXE C:=50
COUNTRY=088
LASTDRIVE=G
DEVICE=RAMDRIVE.SYS 160
DEVIMAND.COM
INSTALL=FASTOPEN.EXE C:=50
COUNTRY=086
LASTDRIVE=G
DEVICE=RAMDRIVE.SYS 160
DEVICE=SETVER.EXE
DEVICE=HIMEM.SYS
DOS=HIGH
```

建好上面的配置文件后，特别要提醒你在重新装入 DOS 前必须确认开机磁盘上含有如下的文件：

```
FASTOPEN.EXE
COUNTRY.SYS
RAMDRIVE.SYS
SETVER.EXE
HIMEM.SYS
KEYB.COM
```

而且一定要有一含有 COMMAND.COM 文件的子目录名 YOUNG，否则无法安装虚拟磁盘，甚至无法正常装入 DOS。一切准备好后，就可以冷启动或热启动的方式重新装入 DOS，当装入 DOS 后屏幕会显示如下信息：

```
Microsoft RAM Drive version 3.06 virtual disk D:
    disk size:160K
    Sector size:512 bytes
    allocation unit:1 sectors
    Directory entries:64

HIMEM:DOS XMS Driver,Version 2.77—02/27/91
XMS Specification Version 2.0
Copyright 1988—1991 Microsoft Corp.
```

```
Installed A20 handler number 1.
64K High Memory Area is available.

FASTOPEN installed

Microsoft(R) MS—DOS(R) version 5.00
                (C)Copyright Microsoft Corp 1981—1991.

A:\>D:        (此时就可开始使用虚拟驱动器 D 了)

D:\>DIR

    Volume in drive D is MS--RAMDRIVE
    Directory of D:\

File not found

D:\>A:

A:\>MEM

    655360 bytes total conventional memory


    655360 bytes available to MS  DOS
    466752 largest executable program size

    393216 bytes total contiguous extended memory
         0 bytes available contiguous extended memory
    327680 bytes available XMS memory
           MS—DOS resident in High Memory Area

A:\>
```

由上面所显示的信息和操作可看出，你不但拥有一部 160K 的虚拟驱动器 D 外，也将部分的 DOS 放在高端存储区，并且装入 FASTOPEN 指令以便快速寻找并存取硬盘上的文件。另外，你也指定了所需的 FILES 和 BUFFERS 数目，并指定国家代码为大陆的 086(你可以利用 TIME 和 DATE 指令来显示我国的时间和日期格式)，而磁盘名称的最大值则设定为代号 G。

9.2　DOS 6.0 新增的配置文件功能

1. 配置文件的暂停执行功能

在前面介绍 SWITCH 命令时，曾提到 DOS 6.0 新增的选项/N，它的功能是用来防止用户

按功能键 F5 或 F8 来避开启动指令的执行。这也就是 DOS 6.0 提供的一个新功能，可以让你在打开主机装入 DOS 显示"Staring MS－DOS..."信息时，利用功能键 F5 或 F8 来跳过配置文件和启动文件的执行。

这个新功能在测试配置文件的内容时非常方便，当你打开主机装入 DOS 显示"Starting MS－DOS..."信息时，可以按 F8 键来要求 DOS 在执行配置文件的各行指令前，显示提示信息来询问你是否执行它。例如下列的实际操作：

```
Starting MS  DOS...   (在此按下 F8 键)
MS－DOS will prompt you to confirm each CONFIG.SYS command.
DOS=HIGH[Y,N]? Y
DEVICE=C:\DOS6\SETVER.EXE[Y,N]? Y
DEVICE=C:\DOS6\HIMEM.SYS[Y,N]? Y
SHELL=C:\DOS6\COMMAND.COM C:\DOS6\ /P[Y,N]? Y
FILES=15[Y,N]? N
BUFFERS=10[Y,N]? N
Process AUTOEXEC.BAT[Y,N]? N

Micorsoft(R) MS－DOS(R) Version 6
          (C)Copyright Microsoft Corp 1981－1993.


C:\>
```

下列的实际操作，则是按 F5 键直接跳过配置文件和启动文件的执行，那么 DOS 将以预设的配置来定义系统。

```
Starting MS  DOS...   (在此按下 F5 键)
MS  DOS is bypassing your CONFIG.SYS and AUTOEXEC.BAT FILES.

Microsoft(R) MS  DOS(R) Version 6
          (C)Copyright Microsoft Corp 1981  1993.

C:\>
```

DOS 6.0 还提供了一种不需按功能键 F5 或 F8，也可以在执行配置文件时显示提示信息的功能。你只要在某一命令行后加一个问号，例如：

```
DEVICE? =C:\DOS6\HIMEM.SYS
```

就可在执行到该命令行时，显示提示信息来询问你是否执行它。

2. 多配置文件的选择功能

PC 机的功能越来越强，程序包的种类也越来越多，而且各程序所需的操作环境也不完全相同，为了应付这种多变的状况，DOS 6.0 提供了一种多配置文件选择目录的功能，以满足用户的需要。这个新增的功能在多人共用一部 PC 机时，更显得有用。它是利用如下五个新的配置命令来完成的：

INCLUDE——用来包含配置设定区段的内容。
MENUITEM——用来定义配置功能表上的选择目录。
MENUDEFAULT——用来指定配置功能表的预选项目及暂停的时间。
MENUCOLOR——用来指定配置功能表的屏幕背景及文字颜色。
SUBMENU——用来定义配置功能表的选择子目录。
首先利用如下的范例来说明多配置文件选择目录的功能：

```
REM PROGRAM NAME: TCONFIG.SYS
REM ----------------------------

[MENU]
MENUITEM GENERAL  , A CONFIGURATION FOR GENERAL
MENUITEM WINDOWS  , A CONFIGURATION FOR WINDOWS
MENUITEM dABSE    , A CONFIGURATION FOR dBASE
MENUITEM QBASIC   , A CONFIGURATION FOR QBASIC

SUBMENU = USERS   , A CONFIGURATION FOR USERS
MENUCOLOR  4, 7
MENUDEFAULT = USERS, 15

[COMMON]
DEVICE = C:\DOS6\SETVER.EXE
DEVICE  C:\DOS6\HIMEM.SYS
SHELL  C:\DOS6\COMMAND.COM C:\DOS6\ /P
DOS  HIGH

[GENERAL]
FILES = 15
BUFFERS = 10

[WINDOWS]
FILES = 25
BUFFERS = 20

[dBASE]
INCLUDE = WINDOWS

[QBASIC]
INCLUDE = GENERAL

[USERS]
MENUITEM = ZNANG,A CONFIGURATION FOR ZHANG
MENUITEM = CHEN, A CONFIGURATION FOR CHEN
```

```
MENUITEM = LIN, A CONFIGURATION FOR LIN

[ZHANG]
REM BLOCK FOR USER:ZHANG

[CHEN]
REM BLOCK FOR USER:CHEN

[LIN]
REM BLOCK FOR USER:LIN

[COMMON]
REM BLOCK FOR OTHER COMMON USE
```

注意:此配置文件范例存储在 TCONFIG.SYS 文件中,请你调整[COMMON]区段中各命令所指定的路径名称后,再将 TCONFIG.SYS 的文件更改为 CONFIG.SYS 就可执行。

当你重新装入 DOS 时,屏幕上就会显示如下的功能目录,供你选择不同的配置设定:

```
MS  DOS 6 Startup Menu

   1. A CONFIGURATION FOR GENERAL
   2. A CONFIGURATION FOR WINDOWS
   3. A CONFIGURATION FOR dBASE
   4. A CONFIGURATION FOR QBASIC
   5. A CONFIGURATION FOR USERS
Enter a choice: 5      Time remaining: 11

F5=Bypass startup files F8=Confirm each CONFIG.SYS line [N]
```

当前反相的效果停留在第五项上,如果你在 15 秒内没有做任何的选择,那么 DOS 就自动选择这一项来执行。然后会显示如下的子目录供你选择:

```
MS-DOS 6 Startup Menu

   1. A CONFIGURATION FOR ZHANG
   2. A CONFIGURATION FOR CHEN
   3. A CONFIGURATION FOR LIN
Enter a choice: 1

F5=Bypass startup files F8=Confirm each CONFIG.SYS line [N]
```

当前反相的效果停留在第一项上,也就预选的项目为第一项,因为没有设定暂停的时间,所以必须选定项目按下 Enter 键后才会执行。

熟悉了多配置文件选择目录的功能后,现在就来进一步介绍上述五个新配置命令的详细

用法。

(1) INCLUDE 命令

其使用格式为：

INCLUDE = blockname

其中 blockname 就是配置设定区段的名称，以便在某一区段中包含指定区段的内容。例如上面的范例中，在[dBASE]的区段中包含[WINDOWS]区段中所设定的配置命令。

(2) MENUITEM 命令

其使用格式为：

MENUITEM = blockname [,menu_text]

其中的 blockname 就是你要显示在选择目录中的区段名称，而 menu_text 则是用来说明区段功能的提示信息。要注意区段名称的长度不能超过 70 个字符，而且不能使用\、/、,、;、=、[、]等字符。

(3) MENUDEFAULT 命令

其使用格式为：

MENUDEFAULT = blockname [,timeout]

其中的 blockname 就是用来指定预选的区段名称，而 timeout 则是所要等待的时间，长短介于 0～99 秒之间，也就是在所指定的暂停时间内，如未做任何选择，DOS 将会自动去执行预选的设定区段。如指定 0 秒，则不会显示选择目录，而且立即执行预选的区段。如未指定秒数，则会一直等待直到你按下 Enter 键才会执行预选的区段。

(4) MENUCOLOR 命令

其使用格式为：

MENUCOLOR = x[,y]

其中 x 和 y 分别代表彩色屏幕上所要显示的文字和背景颜色代码，各代码的颜色列表如下：

代码	颜色	代码	颜色
0	黑色	8	灰色
1	蓝色	9	亮蓝色
2	绿色	10	亮绿色
3	青蓝色	11	亮青蓝色
4	红色	12	亮红色
5	紫红色	13	亮紫红色
6	棕色	14	黄色
7	白色	15	亮白色

(5) SUBMENU 命令

其使用格式为：

SUBMENU — blockname [,menu text]

其中的 blockname 就是要显示选择子目录的区段名称，而 menu text 则是用来说明区段功能的提示信息。例如上面范例中的[USERS]区段，就是用来定义子目录的三个项目，以便选择。

另外，在上面的范例中使用了两个区段名称：一个是[MENU]，这是用来组织 MENU-ITEM 命令所指定的选择项目，以便显示一个功能目录供你选择。而[COMMON]则是用来组织共享的配置命令，也就是出现在这个区段中的设定，都是各个区段所需的。为了避免重复出现在各个区段中，利用这种方式可以简化配置文件的内容。当然你也可以利用 INCLUDE 命令来共享其他的区段内容。

9.3 屏幕及键盘的扩充功能

某些应用程序需要你装入 ANSI.SYS 这个驱动程序来扩充屏幕的功能。如果你希望使用 DOS 所提供的一项扩充屏幕以及键盘控制功能，就可以在 CONFIG.SYS 文件内，以 DEVICE 命令来指定 ANSI.SYS 这个设备驱动程序如下：

DEVICE=ANSI.SYS

以便装入 DOS 时，使 DOS 把它的标准输入以及标准输出变成扩充功能了。

现在，就以实例来说明 ANSI.SYS 所提供的扩充功能。

利用 ANSI.SYS 这个程序驱动程序所提供的指令，可以让你在 DOS 提示符下来扩充屏幕的显示功能及键盘的输入功能。例如下列的指令：

ESC[2J

ESC[12;1H

ESC[65;71p

可以分别用来清除屏幕、设定光标位置及改变键盘的字符输入。ANSI.SYS 批处理文件中无法直接以 ESC 键的方式输入，而必须同时按 Ctrl+[键来输入这个符号。但是一般的文本编辑程序，在你同时按 Ctrl+[键时可能有不同的符号出现，最简单的方法是利用 EDLIN 指令来输入 ESC 这个符号，你可以先按 Ctrl+V 键，然后接着输入[符号，屏幕上就会出现 ^[的符号来表示这是同时按 Ctrl+[的结果。现在，请你实际建立如下的批处理文件来练习一下：

```
A:\>EDLIN CLEAR.BAT
New file
*I
       1:*@ECHO OFF
       2:*EHCO ^V[|2J
       3:*^C
*L
       1:*@ECHO OFF
       2:*ECHO ^[|2J
```

```
*E

A:\>
```

文件建立完成后，你只要在DOS提示符下输入CLEAR，就可和CLS指令的功能一样将屏幕清除掉。如果屏幕无法清除，首先核对一下你所输入的内容是否如上所列，如有错误请修改后，再执行一次看看。如果还是无法清除屏幕，就必须列出CONFIG.SYS配置文件的内容，检查是否含有如下的ANSI.SYS设备驱动程序：

```
DEVICE—ANSI.SYS
```

假如没有，请加入这一行后，重新装入DOS，然后再执行CLEAR即可。

下表列出ANSI.SYS所提供的全部指令和各个指令的功能，在你使用这个指令时，一定要特别注意各个指令中的字母，绝不可以误用大小写，以免造成错误。例如ESC[34m绝不能输入ESC[34M。

ESC[n1;n2H	设定光标的位置，其中n1代表行数，而n2代表列数，如果省略n1和n2，则表示设定光标到第一行第一列，即屏幕左上角的位置。
ESC[n1;n2f	这个指令的功能和ESC[n1;n2H相同。
ESC[nA	往上移动光标n行，如未指定n，则往上移动一行。
ESC[nB	往下移动光标n行，如未指定n，则往下移动一行。
ESC[nC	往右移动光标n个字符，如未指定n，则往右移动一个字符。
ESC[nD	往左移动光标n个字符，如未指定n，则往左移动一个字符。
ESC[s	存储当前光标的位置。
ESC[u	取得ESC[s指令所存储的光标的位置。
ESC[2J	清除屏幕，并将光标移到左上角。
ESC[K	清除当前光标所在的一行。
ESC[=7h	当输入的字符数超过80个时，自动到下一行继续显示。
ESC[=7l	当输入的字符数超过80个时，只在第80列上显示你所输入的字符。
ESC[n1;n2p	键盘按键的改变，n1和n2代表字符的ASCII代码，也就是当你按n1所代表的键时，将会显示n2所代表的字符。例如ESC[71;84P表示按G键时将会显示T字符。
ESC["c1c2"p	这个指令的功能和ESC[n1;n2p相同，只是直接以字符表示你所要改变的按键。例如ESC["AG"p。
ESC[o;n;"string"p	这个指令的功能和上面两个指令类似，只是利用键盘上的特殊键，例如F1～F10的功能键，或是配合Shift、Ctrl、Alt和功能键所组合的按键来产生你所指定的字符串(string)。例如ESC[0;68;"CHKDSK"p就表示当你按F10功能键时，就可显示CHKDSK的指令。如果你想在显示CHKDSK指令后按下Enter键来执行这个指令，只要变成ESC[0;68;"CHKDSK";13p即可。各按键的代码(o;n)列表如下供你参考：

	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10
	59	60	61	62	63	64	65	66	67	68
Shift+	84	85	86	87	88	89	90	91	92	93
Ctrl+	94	95	96	97	98	99	100	101	102	103
Alt+	104	105	106	107	108	109	110	111	112	113

例如你想利用按 Alt+F10 来执行 CHKDSK 指令，则只要 ESC [0;113;"CHKDSK";13p 就可达到这个效果。如果要取消你所指定的按键显示功能，则必须重新装入 DOS 才行。

ESC[n;nm

这个指令可用来改变屏幕的显示属性。各 n 值所能改变的属性依你所使用的显示器种类而有所不同，列表如下供你参考：

n 值	彩色显示器	单色显示器
0	正常显示	正常显示
1	高亮度	高亮度
2	正常亮度	正常亮度
4	蓝色	加底线
5	闪烁	闪烁
7	反相影像	反相影像
8	反相字符	反相字符
30	黑色前景	
31	红色前景	
32	绿色前景	
33	黄色前景	
34	蓝色前景	加底线
35	紫红色前景	
36	深蓝色前景	
37	白色前景	
40	黑色背景	
41	红色背景	
42	绿色背景	
43	黄色背景	
44	蓝色背景	
45	紫红色背景	
46	深蓝色背景	
47	白色背景	

ESC[=nh　这个指令可用来设定屏幕的显示模式。各 n 值所能设定的模式列表如下仅供参考：

n 值	显示模式
0	单色文字(40 列 * 25 行)
1	彩色文字(40 * 25)
2	单色文字(80 * 25)
3	彩色文字(80 * 25)
4	四色中分辨率图形(320 * 200 像素)
5	与 4 相同
6	双色分辨率图形(640 * 200 像素)
14	彩色图形(640 * 200)
15	单色图形(640 * 350)
16	彩色图形(640 * 350)
17	彩色图形(640 * 480)
18	彩色图形(640 * 480)
19	彩色图形(320 * 200)

9.4　在 PROMPT 指令中使用 ANSI.SYS

PROMPT 指令是用来改变 DOS 提示符，你也可以在 PROMTP 指令中配合 ANSI.SYS 所提供的功能来显示一些特别的效果。例如下列的指令：

PROMPT te[D$e[D$e[D$g

就是用来显示时间(不包括百分秒部分)和提示符)。这个指令其实和下列的指令的功能完全相同：

PROMPT thhh$g

因为 $e[D 和 $h 是一样的。

在 PROMPT 指令中使用 ANSI.SYS，必须以 $e 来代替 ESC，然后紧接着 ANSI.SYS 所提供的指令。例如 ESC[D 在 PROMPT 指令中就必须以 $e[D 的方式来使用。下列的 PROMPT 指令是综合了各种不同的 ANSI.SYS 功能，在彩色屏幕上显示一个非常有趣的提示符：

PROMPT $e[s$e[1;1H$e[7m$e[Kpe[1;45H$d @ te[0m$e[u$g

为了让你了解这个指令的功能，现在将它分解并说明如下：

$e[s　存储当前光标的位置(即 ESC[s)

$e[1;1H　移动光标到屏幕左上角(即第一行第一列的位置)

$e[7m　设定屏幕为反相效果(即 ESC[7m)

$e[K　清除光标所在的一行(即 ESC[K)

$p	显示当前目录的路径
$e[1;45H	移动光标到屏幕第 1 行第 45 列的位置(即 ESC[1;45H)
$d	显示日期
@	先显示一个空格,然后显示@符号
$t	先显示一个空格,然后显示时间
$e[0m	设定屏幕为正常显示(即 ESC[0m)
$e[u	恢复到原来的光标位置
$g	显示符号〉

而下列的 PROMPT 指令则是用来显示一个功能目录,并且将提示符改变成[ ],以便用户在方框中输入所选择的号码:

```
PROMPT MAIN MENU $_ 1. DOS $_ 2. 123 $_ 3. WP $_ 4. dBASE
    $_ CHOICE (1-4) [ ] $E[D$E[D
```

当你执行了这个指令后,屏幕将会显示如下的功能目录供你选用:

```
MAIN MENU
1. DOS
2. 123
3. WP
4. dBASE
CHOICE (1-4) [ ]
```

当前的光标将会停留在方框中,以便提示你输入 1~4 的号码。你可以把这个显示功能目录的 PROMPT 指令存放在一个批处理文件中,然后另外再建立文件名为 1.BAT、2.BAT、3.BAT和 4.BAT 的批处理文件来执行中目录的功能,并且把提示符恢复为正常的显示。例如下列各批处理文件的内容:

```
:PROGRAM NAME:MENU2.BAT
@ECHO OFF
CLS
PROMPT MAIN MENU $_ 1.DOS $_ 2.123 $_ 3.WP $_ 4.dBASE $_ CHOICE(1-4)[ ]$E[2D
:END

:PROGRAM NAME:1.BAT
@ECHO OFF
PROMPT $P$G
:END

:PROGRAM NAME:2.BAT
@ECHO OFF
PROMPT $P$G
CD\123
123
:END
```

```
:PROGRAM NAME:3.BAT
@ECHO OFF
PROMPT $P$G
CD\WP
WP
:END

:PROGRAM NAME:4.BAT
@ECHO OFF
PROMPT $P$G
CD\dBASE
dBASE
:END
```

9.5　ANSI.SYS 批处理文件范例

在你执行下列各范例前，切记要在 CONFIG.SYS 配置文件中以 DEVICE 命令来指定 ANSI.SYS，以便装入 DOS，可以使你的屏幕和键盘具有扩充的功能，才能正确地执行这些范例。

1. 改变彩色屏幕的前景为蓝色，并在第十二行上显示提示信息。

```
:PROGRAM NAME:EXAMPLE.BAT
REM ANSI DISPLAY EXAMPLE
@ECHO OFF
REM FIRST,CLEAR THE SCREEN
@ECHO ^[[2J
REM TURN ON BLUE FOREGROUND COLOR
REM UNDERLINE MONOGROUND
ECHO ^[[34m

REM DISPLAY MESSAGE
ECHO FEELING BLUE?
REM RESET COLOR BACK TO NORMAL
ECHO ^[[0m
REM MOVE CURSOR TO ROW 12,COLUMN 1
ECHO ^[[12;1H
ECHO HELLO,I AM HERE!
```

如果你所使用的是单色屏幕，那么"FEELING BLUE?"这段文字下将会加一条底线。

2. 在彩色屏幕上画出一面美国国旗。

```
:PROGRAM NAME:FLAG1.BAT
@ECHO OFF
```

```
REM American Flag ANSI display program
CLS
ECHO ^[[37;44m* * * * * ^[[41m                    =
ECHO ^[[37;44m   * * * *   ^[[47m                 =
ECHO ^[[37;44m   * * * *   ^[[41m  (25 个空格)    =
ECHO ^[[37;44m* * * * * ^[[47m                    =
ECHO ^[[37;44m* * * * * ^[[41m                    =
ECHO ^[[47m                                       =
ECHO ^[[41m                                       =
ECHO ^[[47m                                       =
EHCO ^[[41m  (38 个空格)                          =
ECHO ^[[47m                                       =
ECHO ^[[41m                                       =
ECHO ^[[0m
```

为了便于区别空格起见，各行指令后都以等号结尾，当你在执行之前可将等号删除。你如果使用单色屏幕，则可利用下列的范例来显示美国国旗：

```
:PROGRAM NAME:FLAG2.BAT
@ECHO OFF
REM American Flag ANSI display program
CLS
ECHO ^[[37;44m* * * * * ^[[7m                     =
ECHO ^[[37;44m   * * * *   ^[[0m                  =
ECHO ^[[37;44m   * * * *   ^[[7m                  =
ECHO ^[[37;44m* * * * * ^[[0m                     =
ECHO ^[[37;44m* * * * * ^[[7m                     =
ECHO ^[[0m                                        =
ECHO ^[[7m                                        =
ECHO ^[[0m                                        =
ECHO ^[[7m                                        =
ECHO ^[[0m                                        =
ECHO ^[[7m                                        =
ECHO ^[[0m                                        =
```

3. 在 40 个字符宽的彩色屏幕上显示一个彩色的功能目录表：

```
*:PROGRAM NAME:MENUEX.BAT
@ECHO OFF
REM ANSI batch file to create a 40—column menu
CLS
REM Set 40 column screen mode (color):
REM ECHO ^[[=1h
ECHO ^[[5;15h ^[[1mMAIN MENU
```

```
ECHO ^ [[8;10H^ [[31;1m1. ^ [[2;31m Break into company computer
ECHO ^ [[10;10H^ [[32;1m2. ^ [[2m Look at payroll files
EHCO ^ [[12;10H^ [[33;1m3. ^ [[2m Give me raise
ECHO ^ [[14;10H^ [[34;1m4. ^ [[2m Give Jamision demotion
ECHO ^ [[16;10H^ [[35;1m5. ^ [[2m Stealthily log off
ECHO ^ [[20;15H^ [[1;5mEnter choice: ^ [[0m
```

为了节省建立文件的时间，你可以在 ECHO 指令后同时输入几个 ANSI.SYS 指令。例如下面两个指令：

```
ECHO ^ [[5;15H
ECHO ^ [[1mMAIN MENU
```

可以合并成下列的指令：

```
ECHO ^ [[5;15H^ [[1mMAIN MENU
```

这种合并方式虽可节省一些按键的时间，但是看起来有点复杂，不如分开清楚。

4. 在指定的屏幕位置上显示信息：

```
:PROGRAM NAME:LOCATE.BAT
@ECHO OFF
REM LOCATE batch file,positions cursor on the screen
REM input is %1=row,%2=column,%3=through %9=message to display
REM is insufficient input,cursor is unchanged
REM First,save the cursor position:
ECHO ^ [[s
IF !%1=! GOTO LOOPS
IF !%2=! GOTO LOOPS

ECHO ^ [[%1;%2H%3 %4 %5 %6 %7 %8 %9
GOTO END
:LOOPS
REM Error condition,restore cursor position
ECHO ^ [[u
:END
```

当你要执行这个范例时，只要在文件名后指定你所要的屏幕位置及提示信息即可。例如：

```
LOCATE 12 40 HELLO!
```

就可将"HELLO!"这个提示信息显示在屏幕中央。

5. 改变键盘按键所显示的字符：

```
:PRPGRAM NAME:CHANGE.BAT
@ECHO OFF
REM change Alt—N to n̄
ECHO ^ [[0;49;164p
REM change Control—N to Ñ
ECHO ^ [[14;165p
```

```
@ECHO OFF
REM change ' to é
ECHO ' [[96;130p
REM change ~ to ê
ECHO ' [[126;144p
```

执行这个范例后，就可利用下列的各按键来输入你所指定的字符：

Alt—N　　→ñ

CTRL N　　→Ñ

'　　→é

~　　→ê

6. 设定键盘按键以便输入你所指定的指令：

```
;PROGRAM NAME:NIFTY.BAT
@ECHO OFF
ECHO ' [[0;104;"C:";13p;"CD\";13p
ECHO ' [[0;105;"CD \WP\WP\";13p
ECHO ' [[0;106;"DIR A:";13p
ECHO ' [[0;107;"DIR *.BAT";13p
ECHO ' [[0;108;"DEL *.BAK";13p
ECHO ' [[0;109;"DEL TEMP*.*";13p
ECHO ' [[0;110;"FORMAT A:";13p
ECHO ' [[0;111;"CD \WORK\BOOK";13p
ECHO ' [[0;112;"CD \SYSTEM\TOOLS";13p
ECHO ' [[0;113;"SHOUTDOWN";13p
```

当你执行这个范例后，就可利用 Alt+F1 到 Alt+F10 键来输入各个指令。当然，你可以修改这个范例中所指定的指令，就可输入你在操作 DOS 时最常用的指令，以节省按键的时间。

7. 打开屏幕的滚动功能：

```
;PROGRAM NAME:WRAP  ON.BAT
@ECHO OFF
ECHO ' [[71
ECHO Character wrap is on
```

执行这个范例后，当你输入指令时，如果字符数超过 80 个时，只在第 80 列上(即屏幕最右边)显示你所输入的字符。如果你想关闭屏幕的滚动功能，则执行下一个范例即可。

8. 关闭屏幕的滚动功能

```
;PROGRAM NAME:WRAP—OFF.BAT
@ECHO OFF
ECHO ' [[7h
ECHO Character wrap is off
```

9. 改变键盘的排列格式：

```
;PROGRAM NAME:DVORAK.BAT
```

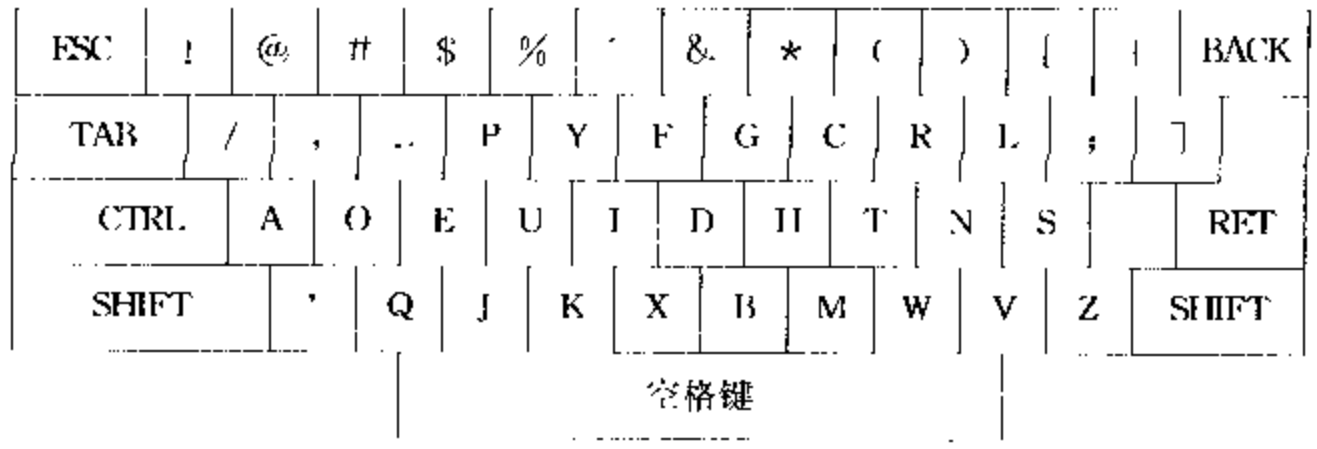
```
@ECHO OFF
REM Drorak keyboard layout;
ECHO Installing Duorak Keyboard Layout
ECHO Press Control + Break to stop,or
PAUSE
REM     to |,  to {
EHCO ^ [[31;27p
ECHO ^ [[45;91p
ECHO ^ [[95;123p
REM q to /,Q to ?
ECHO ^ [[113;47p
ECHO ^ [[81;63p
REM w to ,, W to less  than
ECHO ^ [[119;44p
ECHO ^ [[87;60p
REM e to . ,E to greater  than
ECHO ^ [[101;46p
ECHO ^ [[69;62p
REM R to P
ECHO ^ [[18;16p
ECHO ^ [[114;112p
ECHO ^ [[82;80p
REM T to Y
ECHO ^ [[20;25p
ECHO ^ [[116;121p
ECHO ^ [[84;89p
REM Y to F
ECHO ^ [[25;6p
ECHO ^ [[121;102p
ECHO ^ [[89;70p
REM U to G
ECHO ^ [[21;7p
ECHO ^ [[117;103p
ECHO ^ [[85;71p
REM I to C
ECHO ^ [[9;3p
ECHO ^ [[105;99p
ECHO ^ [[73;67p
REM O to R
ECHO ^ [[15;18p
ECHO ^ [[111;114p
ECHO ^ [[79;82p
REM P to L
```

```
ECHO ^ [[16;12p
ECHO ^ [[112;108p
ECHO ^ [[80;76p
REM [ to ;, { to :
ECHO ^ [[91;59p
ECHO ^ [[123;58p
REM S to O
ECHO ^ [[19;15p
ECHO ^ [[115;111p
ECHO ^ [[83;79p
REM D to E
ECHO ^ [[4;5p
ECHO ^ [[100;101p
ECHO ^ [[68;69p
REM F to U
ECHO ^ [[6;21p
ECHO ^ [[102;117p
ECHO ^ [[70;85p
REM G to I
ECHO ^ [[7;9p
ECHO ^ [[103;105p
ECHO ^ [[71;73p
REM H to D
ECHO ^ [[8;4p
ECHO ^ [[104;100p
ECHO ^ [[72;68p
REM J to H
ECHO ^ [[10;8p
ECHO ^ [[106;104p
ECHO ^ [[74;72p
REM K to T
ECHO ^ [[11;20p
ECHO ^ [[107;116p
ECHO ^ [[75;84p
REM L to N
ECHO ^ [[12;14p
ECHO ^ [[108;110p
ECHO ^ [[76;78p
REM ; to S
ECHO ^ [[59;115p
ECHO ^ [[58;83p
REM ' to —
ECHO ^ [[39;45p
```

```
ECHO ^ [[34;95p
REM Z to '
ECHO ^ [[122;39p
ECHO ^ [[90;34p
REM X to Q
ECHO ^ [[24;17p
ECHO ^ [[120;113p
ECHO ^ [[88;81p
REM C to J
ECHO ^ [[3;10p
ECHO ^ [[99;106p
ECHO ^ [[67;74p
REM V to K
ECHO ^ [[22;11p
ECHO ^ [[118;107p
ECHO ^ [[86;75p
REM B to X
ECHO ^ [[2;24p
ECHO ^ [[98;120p
ECHO ^ [[66;88p
REM N to B
ECHO ^ [[14;2p
ECHO ^ [[110;98p
ECHO ^ [[78;66p
REM , to W
ECHO ^ [[44;119p
ECHO ^ [[60;87p
REM . to V
ECHO ^ [[46;118p
ECHO ^ [[62;86p
REM / to Z
ECHO ^ [[47;122p
ECHO ^ [[63;90p
```

当你执行这个范例后，就可将如下标准的键盘排列格式：

改变成下列的 DVRAK 排列格式：



如果你想恢复为标准的键盘排列格式，则必须重新装入 DOS 才能恢复正常。

9.6 启动文件的使用

每一次你装入 DOS 时，系统都会在用来装入 DOS 的磁盘根目录中寻找一个叫做 AUTOEXEC.BAT的文件。AUTOEXEC.BAT 是一个特殊的批处理文件，它会在你打开主机或重新装入 DOS 时被自动执行，因此也称之为启动文件。记住，你每次以 DOS 系统开机时，DOS 不会自动去执行一些指令，例如要求你输入当前的日期和时间或显示一些有关的信息等，这就是存储于 AUTOEXEC.BAT 文件内的指令被执行的结果。换句话说，当你装入 DOS 时，也可以借助启动文件来执行你想要的指令，比如你可以通过 AUTOEXEC.BAT 使每次装入 DOS 时都能自动执行 PATH 指令，因而减少了很多操作上的不便。请看下面的范例，例如启动文件内含有如下的指令：

```
A:\>TYPE AUTOEXEC.BAT
DATE
TIME
B:
CD \BATCH
DIR *.*
A:\>
```

当你装入 DOS 时，AUOTEXEC.BAT 内的各个指令便会自动的被执行，结果如下所示：

```
A:\>AUTOEXEC
A:\>DATE
Current date is Wed 05-13-1992
Enter new date(mm-dd-yy):6-30-1992

A:\>TIME
Current time is 11:29:54.70a
Enter new time:10:10

A:\>B:
B:\>CD \BATCH
```

```
B:\BATCH)DIR *.*

Volume in drive B has no label
Volume Serial Number is 2859-- 14D6
Directory of B:\BATCH

.          〈DIR〉          05-13-92    10:54a
..         〈DIR〉          05-13-92    10:54a
T1         SET      125    05-08-92    12:20p
T2         SET      119    05-08-92    12:22p
T3         SET      174    05-08-92    12:31p
T4         SET      186    05-08-92    12:34p
T5         SET      145    05-08-92    12:37p
        7 file(s)        749 bytes
                      600064 bytes free

B:\BATCH)
```

现在，你的启动文件不仅会提示你输入日期和时间，它还会改变驱动器并显示其目录。当你精通了 DOS 的操作后，你一定会发现 AUTOEXEC.BAT 是一个相当好用的启动文件。

使用 AUTOEXEC.BAT 文件要特别注意的是：AUTOEXEC.BAT 一定要建立在你用来装入 DOS 的磁盘根目录中。如果你在配置文件中使用了 SHELL 命令改变了存取 COMMAND.COM文件的路径时，最好在 SHELL 命令后加一个/P 参数，就可在装入 DOS 时仍会自动执行 AUTOEXEC.BAT 文件。SHELL 命令的完整使用格式为：

SHELL COMSPEC=[d:][path]COMMAND.COM /P/E:n/C string

DOS 6.0 所提供的 COMMAND.COM 文件新增了几个选项，其使用格式变成：

SHELL COMSPEC=COMMAND [/C string | /E:n | /P | /MSG | /K filename]

各选项的功能分述如下：

/C string	用来指定所要执行的命令。
/E:n	用来指定存储环境变量的字符空间，n 值必须为介于 160 到 32768 之间的整数，而且只能是 16 的倍数。
/P	用来指定新的指令处理程序，以取代旧的指令处理程序，那么在这种情况下，EXIT 指令就没有结束的作用了。另外，DOS 也会在显示提示符前，先执行启动文件，如没有启动文件，DOS 会执行 DATE 和 TIME 两个指令。
/MSG	用来指定全部的错误信息都存储在内存中，这在使用磁盘来执行 DOS 时非常有用，因为不指定此选项，会有部分信息将存储在磁盘上。
/K filename	用来指定 DOS 在显示提示符前，所要执行的程序或批处理文件名称。这在使用 Windows 的软件时很有用，因为它可以用来指定非 AUTOEXEC.BAT 的启动文件。但在一般情况下，如使用此选项可能会造成问题。

另外，当你使用 AUTOEXEC.BAT 文件时，除非你在启动文件中加入 DATE 和 TIME 指令，否则在装入 DOS 时，DOS 是不会再提示你输入日期和时间的。

由于启动文件会在装入 DOS 时被自动的执行，因此常在启动文件的最后一行加上批处理文件的文件名，以便你开机后自动去执行这个批处理文件。例如下列的启动文件内容：

```
A:\>TYPE AUTOEXEC.BAT
DATE
TIME
MENU

A:\>
```

当你打开主机后，在显示日期和时间后，就会显示如下的功能目录表供你选用：

```
* * * Program Menu * * *
1 --- Word Processing
2 --- Spreadsheet
3 --- Database Management
4 --- Telecommunication
Please enter your choice:
A:\>
```

这是因为启动文件中含有 MENU.BAT 这个批处理文件，而在启动文件自动被执行时，这个批处理文件也紧随在 DATE、TIME 指令后被执行。

9.7 多配置文件与启动文件的配合

DOS 6.0 新增的多配置文件的选择功能，也可以配合 AUTOEXEC.BAT 启动文件自动执行的特性，来发挥设定各种配置及执行不同程序的效果。要发挥这种效果，可以利用第七章中所介绍的环境变量，来取得在多配置文件内所选择的区段名称，然后利用 GOTO 命令分别执行相对应的标记。

DOS 提供一个特殊的环境变量名为 CONFIG，它会存储在多配置文件中所选择的区段名称，因此可以在启动文件中利用这个环境变量来设计程序，以配合设定的配置来执行相对应的指令。提供下列的程序范例供你参考：

```
:* PROGRAM NAME:AUTO.BAT
@ECHO OFF
PROMPT $P$G
PATH C:\DOS6
SET TEMP=C:\DOS6
DOSKEY
UNDELETE /S

GOTO %CONFIG%
```

```
:GENERAL
  GOTO END

:WINDOWS
  CD \WINDOWC
  WIN
  GOTO END

:dBASE
   CD \dBASE3
   dBASE
   GOTO END

:QBASIC
   QBASIC
   GOTO END

:ZHANG
   CD \ZHANG
   DIR
   GOTO END

:CHEN
   CD \CHEN
   DIR
   GOTO END

:LIN
   CD \LIN
   DIR

:END
```

这个程序存储在 AUTO. BAT 文件中，你若要实际执行它，就必须将它更名为AUTOEXEC. BAT的启动文件，然后再将前面 9. 2 节中所提供的多配置文件范例(原存储在TCONFIG. SYS文件中)更名为 CONFIG. SYS，并将两个文件的内容调整成你所需要的配置及指令，最后重新装入 DOS 就可显示选择目录。

第十章 工具程序

10.1 前言

DOS 可以说是目前个人计算机上最为普及的操作系统，因此有很多使用 DOS 的高手，特别设计了一些非常方便的工具程序来满足自己的需要。由于这些工具程序在批处理文件的设计上很多帮助，所以特别提供给你参考使用，以提高批处理文件的执行功能。

10.2 工具程序的功能

我们将一些工具程序的主要功能分述如下，当你熟悉了这些工具的用法后，就可应用到你自己所设计的批处理文件中，以发挥特定的效果。

1. ASK

ASK 的功能类似 ECHO 指令，可用来显示你所指定的提示信息，并等待你按 Y 或 N 键。当回答"Y"时，ERRORLEVEL 的值为 0，而回答"N"时，ERRORLEVEL 的值为 1。因此，你可以在批处理文件中利用 ERRORLEVEL 的值来判断，以便执行不同的工作。例如下列的程序范例：

```
:PROGRAM NAME:ASK1.BAT
@ECHO OFF
REM TEST ASK COMMAND
ASK CONTINUE? (Y/N)=-)
IF ERRORLEVEL 1 GOTO NO
IF ERRORLEVEL 0 GOTO YES
GOTO END
:YES
   ECHO YOU PRESS "Y"
   GOTO END
:NO
   ECHO YOU PRESS "N"
:END
```

DOS 6.0 新增的 CHOICE 指令就具备了 ASK 工具程序的功能。

2. BLANKS

BLANKS 的功能是可以让你显示空行，其使用格式如下：

BLANKS n

其中 n 就是用来指定你所要的空行行数。例如下列的程序范例：

```
;PROGRAM NAME:BLANK1.BAT
@ECHO OFF
REM TEST BLANKS UTILITY
ECHO ----------
BLANKS 3
ECHO      I LIKE BATCH FILE
BLANKS 3
ECHO      ----------
;END
```

执行这个范例后，屏幕将显示如下信息：

```
I LIKE BATCH FILE
```

3. BOX

BOX 顾名思义就是用来画方框的一个工具程序，它的使用格式为：

BOX S=n C=x1,y1;x2,y2 "string"/C

格式中的各个参数都是可有可无的，其中 n 是用来指定你所画方框的格式，你可以指定如下的 n 值来显示方框：

n	方框格式
s	单边方框
d	双边方框
b	粗边方框

如未指定则设定为双边方框。x 和 y 值是用来指定你所要方框的显示位置，x1 和 y1 表示方框左上角的位置，而 x2 和 y2 则是方框右下角的位置，其中 x 值是列数，其值必须介于 0～70，而 y 值是行数，其值必须介于 0～24，例如下列的指令：

BOX S=d C=10,10;50,14

将会在屏幕的上半部显示一个长方型的双边方框。

格式中的"string"是用来指定你所要显示在方框内的提示信息，你可指定参数 C 将提示文字显示在方框中间。例如下列的程序范例：

```
;PROGRAM NAME:BOX1.BAT
@ECHO OFF
REM TEST BOX UTILITY
CLS
BOX S=d C=15,15;50,20 "HELLO,HOW ARE YOU?"/C
BLANKS 10
;END
```

BOX 是一个非常理想的工具，你可以经常利用它来美化你屏幕上所显示的提示信息。

4. CLZ

CLZ 的功能和 DOS 的 CLS 指令完全相同，只是在清除屏幕时看起来比较有趣而已。你可以先执行上列的 BOX 范例后，再比较 CLZ 和 CLS 的不同。

5. GREET

GREET 可以根据当前的时间来显示下列的问候词：

Good morming. (早安)

Good afternoon. (午安)

Good evening. (晚安)

你也可以在 GREET 之后指定你所要打招呼的对象，例如下列的程序范例：

```
:PROGRAM NAME:GREET1.BAT
@ECHO OFF
REM TEST GREET UTILITY
CLS
GREET My Love!
BLANKS
:END
```

你可以将上列的程序范例加入 AUTOEXEC.BAT 文件中，当装入 DOS 后就可显示这些问候词。

6. HOLD

HOLD 的功能和 PAUSE 类似，只是不管 ECHO 是 ON 或 OFF，HOLD 都会正常地显示提示信息。例如下列的程序范例：

```
:PROGRAM NAME:HOLD1.BAT
@ECHO OFF
REM TEST HOLD UTILITY
CLS
REM PAUSE
PAUSE GIVE ME A HUG!
REM HOLD
HOLD GIVE ME A HUG!
:END
```

当你执行上列的程序后，就可区别 HOLD 和 PAUSE 的不同。

7. LOCATE

LOCATE 类似 BASIC 语言中的 LOCATE 指令，可用来指定屏幕的位置，以便显示ECHO 指令后提示信息。其使用格式为：

LOCATE x,y

其中 x 表示列数，而 y 表示行数。如未指定 x 和 y，将自动定位在屏幕的左上角。例如下列的程序范例：

```
:PROGRAM NAME:LOCATE1.BAT
@ECHO OFF
REM TEST LOCATE UTILITY
CLS
LOCATE
GREET My Love!
LOCATE 20,10
ECHO I like batch file programing!
BLANKS 12
:END
```

8. READKEY

READKEY 可用来接受按键，并利用 ERRORLEVEL 的值来检查你所按的键。READKEY 的使用格式为：

READKEY [switches][comment]

其中 switches 参数可以是如下表的任一选择：

参数	按键	ERRORLEVEL 设定值
/F	功能键 F1～F10	1～10
/N	数字 0～9	0～9
/A	大小写的字母 A～Z	1～26
/E+	显示你所按的键	
/E	不显示你所按的键	

如果你未指定任何参数，那么 ERRORLEVEL 的值则被设定成你所按键的 ASCII 代码。格式中的 comment 则是用来显示你所指定的提示信息。例如下列的程序范例：

```
:PROGRAM NAME:READKEY1.BAT
@ECHO OFF
REM An example of READKEY
:START
READKEY Type a 1 or 2
IF ERRORLEVEL 51 GOTO RANGE
IF ERRORLEVEL 50 GOTO TWO
IF ERRORLVEVL 49 GOTO ONE
:RANGE
ECHO —— That number is out of range!
GOTO START
:ONE
```

```
ECHO -- You typed the number 1!
GOTO END
:TWO
ECHO -- You typed the number 2!
:END
```

下列的程序范例，因在 READKEY 之后指定了参数 N，所以在执行这个范例时只能输入数字 0～9，其他的按键将不被接受：

```
:PROGRAM NAME:READKEY2.BAT
@ECHO OFF
REM An example of READKEY
:START
READKEY/N Type a 1 or a 2 --}
IF ERRORLEVEL 3 GOTO RANGE
IF ERRORLEVEL 2 GOTO TWO
IF ERRORLEVEL 1 GOTO ONE
:RANGE
ECHO -- That number is out of range!
GOTO START
:ONE
ECHO -- You typed the number 1!
GOTO END
:TWO
ECHO -- You typed the number 2!
:END
```

DOS 6.0 新增的 CHOICE 指令虽然具备 ASK 工具程序的功能，却无 READKEY 的灵活性，也没有 READKEY 方便。

9. REPT

REPT 的功能和 BASIC 语言的 STRING $ 函数类似，可用来连续显示你所指定的字符。它的使用格式为：

```
REPT char n
```

其中 char 就是你所要重复的字符，而 n 则是你所要连续的次数。例如下列的程序范例：

```
:PROGRAM NAME:REPT1.BAT
@ECHO OFF
REM TEST REPT UTILITY
REPT * 23
BLANKS 3
ECHO I LIKE BATCH FILE
BLANKS 3
REPT * 23
```

```
:END
```

执行后，屏幕将会显示如下结果：

```
* * * * * * * * * * * * * * * * * * * * * * *
          I LIKE BATCH FILE
* * * * * * * * * * * * * * * * * * * * * * *
```

10. RESTPATH

RESTPATH 的功能和 7.5 节的程序范例类似，可用来存储当前的文件目录，以便你改变文件目录后，可以恢复为原来的文件目录。RESTPATH 的执行结果，可以下列的 ERRORLEVEL来判断：

ERRORLEVEL	执行结果
0	正常执行
1	无法找到文件目录
2	文件目录无法被恢复

例如下列的程序范例：

```
:PROGRAM NAME:REST1.BAT
@ECHO OFF
REM TEST RESTPATH UTILITY
CLZ
ECHO THE ORIGINAL PATH:
CD
CD > A:\CURRENT
C:
CD \DOS
ECHO CHANGE TO NEW PATH:
CD
A:\UTILITY\RESTPATH A:\CURRENT
ECHO BACK TO THE ORIGINAL PATH:
EC
:END
```

程序中首先利用 CD 指令将当前目录存入 CURRENT 文件中，然后改变当前目录，最后再利用 RESTPATH 恢复为原来的文件目录。例如下列的执行结果：

```
THE ORIGINAL PATH:
A:\UTILITY
CHANGE TO NEW PATH:
C:\DOS
BACK TO THE ORIGINAL PATH:
A:\UTILITY
```

请注意，上列程序范例中的 CURRENT 文件一定要指定路径名称，否则在你改变当前目录后，将无法找到它，而造成 RESTPATH 无法恢复为原来的文件目录。

11. SAY

SAY 的功能和 ECHO 指令类似，但比 ECHO 方便的是，你可以在 SAY 后指定任何的字符串（包括 ON 或 OFF），如果在 SAY 后只输入空格键，则会显示一行空行。你可以在字符串后加上一个分号，那么在显示字符串后将使光标停留在该字符串的后面，以便等待下一个输出。这个功能和在 BASIC 语言的 PRINT 指令后使用分号一样。例如下列的程序范例：

```
:PROGRAM NAME:SAY1.BAT
@ECHO OFF
REM TEST SAY UTILITY
CLZ
SAY HELLO.;
SAY DR. ZHANG
SAY
SAY DO YOU LIKE MS--DOS 5.0?
SAY
SAY YES.;
SAY I DO.
:END
```

当你执行这个范例后，屏幕上将显示如下信息：

```
HELLO.DR. ZHANG

DO YOU LIKE MS  DOS 5.0?

YES,I DO.
```

12. TSTAMP

TSTAMP 可以如下的格式同时显示时间和日期：

```
A:\UTILITY>TSTAMP
09:47:30 07/06/92

A:\UTILITY>
```

和一般的 DOS 指令一样，你也可以将 TSTAMP 的输出结果重新引导到其他的文件中，以便掌握文件的建立日期和时间。

13. VERNUM

VERNUM 与 DOS 的 VER 指令相同，可用来显示目前你使用的 DOS 版本号。但是 VERNUM比 VER 指令更具有灵活性，因为它是利用下表的 ERRORLEVEL 设定值来表示

DOS 版本号，所以你可以在批处理文件中依此辨别：

DOS 版本号	ERRORLEVEL 设定值
6.6	60
5.0	50
4.0	40
3.3	33
3.1	31
3.0	30
2.1	21
1.0	10

例如下列的程序范例：

```
:PROGRAM NAME:VERS.BAT
@ECHO OFF
ECHO This test the vernum batch file
VERNUM
IF ERRORLEVEL 60 GOTO VER60
IF ERRORLEVEL 50 GOTO VER50
IF ERRORLEVEL 40 GOTO VER40
IF ERRORLEVEL 33 GOTO VER33
IF ERRORLEVEL 31 GOTO VER31
ECHO You have a * very * early edition of DOS
GOTO END
:VER60
ECHO DOS Version 6.0 is in use.
GOTO END
:VER50
ECHO DOS Version 5.0 is in use.
GOTO END
:VER40
ECHO DOS Version 4.0 is in use.
GOTO END
:VER33
ECHO DOS Version 3.3 is in use.
GOTO END
:VER31
ECHO DOS Version 3.1 is in use.
:END
```

14. STIME 和 ETIME

STIME 和 ETIME 可以如下的格式来显示日期和时间：

```
A:\UTILITY>STIME
Start Time = Monday,July 6,1992 @ 9:53 am

A:\UTILITY>ETIME
End Time = Monday,July 6,1992 @ 9:55am
```

STIME 是开始计时的日期和时间，而 ETIME 则是结束计时的日期和时间，这两个工具程序的输出结果都可以重新引导到其他文件中，以便掌握建立文件的起始和结束时间。

15. ECOH

ECOH 的功能和 DOS 的 ECHO 指令类似，只是 ECOH 会以反相的效果来显示你所指定的提示信息。

16. HIDE

HIDE 可用来隐藏你所指定的文件，使你在执行 DIR 指令时无法显示在文件目录中。但是你仍可使用这个隐藏的文件。

17. UNHIDE

UNHIDE 的功能和 HIDE 相反，可用来显示被隐藏的文件，你只能在 HIDE 或 UNHIDE 之后指定单一的文件名称。

18. FINDHIDE

FINDHIDE 和 DOS 的 DIR 指令相同，只是会在隐藏的文件名称前显示一个星号以便辨别。

19. WAITN

WAITN 可用来暂停一段时间后，才继续执行下一个指令。它的使用格式为：

```
WAITN n
```

其中 n 就是你所要暂停的时间，以秒为单位。

10.3 综合程序范例

下面所提供的范例是综合上节介绍的工具程序设计而成的，你如果要实际上机操作，一定要含有这些工具程序才能正常执行。

1. 用功能目录表的方式来执行硬盘的备份工作：

```
:PROGRAM NAME:BACKS.BAT
@ECHO OFF
REM BACKS.BAT program:provides various types of backup
REM This program should be on the path so that it's available
REM at all times. Also,note that the DOS BACKUP.COM program
```

```
REM has been pathed.
REM This program encorporates programs on the supplemental
REM programs diskette.    Like:CLZ,LOCATE,READKEY,BOX.
:MAIN MENU
CLZ
BOX C=20,1;63;18
LOCATE 30,3
ECHO Hard Disk Backup Options
LOCATE 25,6
ECHO F1 — Backup the entire hard drive
LOCATE 25,8
ECHO F2 — Backup only modified files
LOCATE 25,10
ECHO F3 — Backup only C:\WP\BOOK
LOCATE 25,12
ECHO F10 — Cancel
:GETINPUT
LOCATE 30,15
READKEY /F Your choice:
IF ERRORLEVEL 10 GOTO END
IF ERRORLEVEL 3 GOTO ONLY
IF ERRORLEVEL 2 GOTO MODIFY
IF ERRORLEVEL 1 GOTO ALL
GOTO GETINPUT
:ONLY
CLZ
BOX "Backing up C:\WP\BOOK to drive A:"/C
BACKUP C:\WP\BOOK A:
GOTO END
:MODIFY
CLZ
BOX "Backing up modified files only"/C
BACKUP C:\*.* A:/S/M
GOTO END
:ALL
CLZ
BOX "Backing up entire hard drive"/C
BACKUP C:\*.* A:/S
:END
LOCATE 1,24
```

当你执行这个范例后，屏幕上将会显示如下的功能目录表供你选用：

```
        Hard Disk Backup Options
```

```
F1 — Backup the entire hard drive
F2 — Backup only modified files
F3 — Backup only C:\WP\BOOK
F10 — Cancel

Your choice:
```

第一项的功能是供你执行整个硬盘的备份，第二项的功能是供你只拷贝曾修改过的文件，而第三项的功能则是用来拷贝子目录\WP\BOOK 中的全部文件。你可以根据自己的需要，修改子目录的名称，以便执行备份的工作。另外，这个程序范例的功能与 DOS 6.0 新增的 MSBACKUP指令，是否有相同之处？

2. 建立一个工作日志文件，以便记录操作计算机的日期和时间(开始到结束)：

```
;PROGRAM NAME:LOG.BAT
@ECHO OFF
REM RUN TIME
CD > A:\CURRENT
C:
CD \QB45
ECHO WORK ON QUICKBASIC >> C:\LOGFILE.DAT
A:\UTILITY\STIME >> C:\LOGFILE.DAT
QB
A:\UTILITY\ETIME >> C:\LOGFILE.DAT
A:\UTILITY\RESTPATH A:\CURRENT
;END
```

当你要执行 QuickBASIC 语言时，只要执行这个范例就可自动进入 QuickBASIC 的控制中心，当你结束 QuickBASIC 的操作后，就可利用 TYPE 指令来显示 LOGFILE.DAT 文件的内容，其中记录了你开始操作 QuickBASIC 到结束的日期和时间。例如下列的 LOGFILE.DAT 内容：

```
WORK ON QUICKBASIC
Start time = Thursday, July 2, 1992 @ 10:58 am
End time = Thursday, July 2, 1992 @ 10:58 am
WORK ON QUICKBASIC
Start time = Wednesday, July 8, 1992 @ 8:02 am
End time = Wednesday, July 8, 1992 @ 8:04 am
```

附录 A　DOS 的指令索引及摘要

下列的指令均适用于 DOS 5.0，各指令均按字母顺序排列，并且附上各指令的使用格式以便参考，指令左上角注有"*"号者表示该指令并未在本书中提到，因此在此详细说明其功能。

* APPEND　用来连接一个目录到另一个目录中。首先装入 APPEND 的使用格式为：

APPEND path1[;path2;...pathN]

连接时，扩展名 EXE、COM 或 BAT 的文件将不会被连接。当 APPEND 装入后，就可使用如下的格式：

APPEND [/X][/E]

/X 将会连接所有的文件。

/E 将连接的路径存储到环境变量 APPEND 中。

APPEND 指令允许你存取数据文件，就像 PATH 指令能让你存取程序文件一样。单一的 APPEND 指令可用来检查当前所连接的目录，而如下的格式则是用来中断任何连接的目录：

APPEND;

* ASSIGN　用来重定驱动器的操作，其使用格式如下：

ASSIGN drive1 = drive2 [drive3 = drive4...]

下达此指令后，任何对驱动器 1 存取的操作，都会改成对驱动器 2 执行。若使用未含任何参数的 ASSIGN 指令，则所有驱动器都恢复为正常的存取状态。例如：

ASSIGN A = B　B = A

就可将驱动器 A 和 B 对调。而下列的指令则是用来显示当前的指令状态：

ASSIGN /S

* ATTRIB　用来设定文件的属性。另外，也可用来显示当前文件属性的设定方式。其使用格式为：

ATTRIB [+R] [−R] [+A] [−A] [+H] [--H] [+S]
[-S] [filename] [/S]

+R 设定文件的只读属性为正。

−R 设定文件的只读属性为负，也就是恢复文件为可读可写的属性。

+A 设定文件的保存属性为正。

−A 设定文件的保存属性为负。

+H 设定文件的隐藏属性为正。

-H 设定文件的隐藏属性为负。

+S 设定文件的系统属性为正。

—S 设定文件的系统属性为负。
/S 用来设定当前目录与其所属全部子目录的属性。
若使用未含任何参数的 ATTRIB 指令，则用来显示当前文件的属性设定方式。下例则是用来设定当前目录所有扩展名为 EXE 的文件为只能读取的属性：

```
ATTRIB +R *.EXE
```

BACKUP 主要是用来制作硬盘的备份。

```
BACKUP source drive [filename] target drive [/A]
       [/D: date] [/F] [/L] [/M] [/S] [/T:time]
```

BREAK 用来改变对控制中断(即按 Ctrl+Break 键)的检查。

```
BREAK [ON | OFF]
```

*CHCP 用来选择、改变或显示外国文字的代码页(Code Page)。

```
CHCP [nnn]
```

CHDIR 用来改变当前目录(与 CD 指令相同)。

```
CHDIR path
```

CHKDSK 用来显示磁盘及内存的使用情况。

```
CHKDSK [drive specifier] [filename] [/F] [/V]
```

CLS 用来清除屏幕。

```
CLS
```

COMP 用来比较两个文件的内容是否相同。

```
COMP file1 file2
```

COMMAND 用来启动一个新的指令处理程序(Command Processor)。

```
COMMAND
```

COPY 用来拷贝文件内容到另一个文件中。

```
COPY source destination [/V]
```

*CTTY 用来改变标准的输入与输出设备。

```
CTTY device name
```

DATE 用来显示与改变日期

```
DATE [date]
```

*DEBUG 用来帮助程序设计员进行查错的工作。

```
DEBUG
```

DEL 用来删除文件(与 ERASE 指令相同)。

```
DEL filename [/P]
```

DIR 用来显示磁盘文件目录等信息。

```
DIR [filename] [/P] [/W] [/A] [/O] [/S] [/B] [/L]
```

DISKCOMP 用来比较两张磁盘的内容是否相同。

```
DISKCOMP drive1 drive2
```

DISKCOPY 用来拷贝磁盘。

```
DISKCOPY source destination[/v]
```

DOSKEY 用来保留输入的指令行,以便重复使用。

DOSKEY

* DOSSHELL 用来启动 Shell 界面。

DOSSHELL

EDIT 用来启动编辑程序。

EDIT [filename] [/B] [/G] [/H] [/NOHI]

EDLIN 用来启动行编辑程序。

EDLIN filename

EMM386 用来开关支持扩展内存的功能(CPU 为 386 以上的计算机)。

EMM386 [ON|OFF|AUTO] [W=ON/OFF]

ERASE 用来删除文件(与 DEL 指令相同)。

ERASE filename [/P]

* EXE2BIN 用来转换 EXE 文件为二进制格式。

EXE2BIN input file [output_file]

EXIT 用来停止当前的指令处理程序。

EXIT

* EXPAND 用来展开 DOS 5.0 版本上的压缩文件。

EXPAND filename destination

* FASTOPEN 用来存储文件的存储位置以便快速存取所要的文件。其使用格式如下:

FASTOPEN drive specifier [=num] [/X]

其中 num 表示 DOS 所能存储的文件数目,其值介于 10 到 999,设定值 48。

/X 允许你使用扩展的内存。

FASTOPEN 指令只能在你开机后执行一次。下列就是用来存储硬盘上 48 个文件的存储位置。

FASTOPEN C:

* FC 用来比较两个文件并显示二者的差异。

FC [/A] [/C] [/L] [/N] [/T] [/W] [/B]

FDISK 用来对一个硬盘驱动器进行分区。

FDISK

FIND 用来寻找某一字符串是否在文件中。

FIND [/C] [/I] [/N] [/V] "string" file_list

FORMAT 用来格式化磁盘。

FORMAT drive_specifier [/V] [/S] [/B] [/Q] [U]

* GRAFTABL 用来装入额外的符号数据表(Character Table)到存储器中。

GRAFTABL [xxx] [/STATUS]

* GRAPHICS 用来准备打印屏幕上的图形。其使用格式如下:

GRAPHICS [printer] [/R] [/B] [/LCD]

其中 printer 的名称可根据如下的列表来选用：

打印机类型	名称
IBM 彩色打印机(附黑色色带)	COLOR1
IBM 彩色打印机(附 RBG 色带)	COLOR4
IBM 彩色打印机(附 CMY 色带)	COLOR8
任何惠普的 PCL 打印机	HPDEFAULT
惠普的 DeskJet 打印机	DESKJET
IBM 图形打印机、Proprinter 和 Quietwriter 打印机	GRAPHICS
IBM 使用 11 英寸宽卡槽的打印机	GRAPHICSWIDE
惠普 LaserJet 激光打印机	LASEJET
惠普 LaserJet II 激光打印机	LASERJET II
惠普 PaintJet 打印机	PAINTJET
惠普 QuietJet 打印机	QUIETJET
惠普 Quietjet Plus 打印机	QUIETJETPLUS
惠普 RuggedWriter 打印机	RUGGEDWRITER
惠普 RuggedWriterWide 打印机	RUGED- WRITERWIDE
IBM Thermal 打印机	THERMAL
惠普 ThinkJet 打印机	THINKJET

如未指定打印机名称则自动设定为 GRAPHICS，常用的 Epson MX—70、MX—80 MX—100 也是使用 GRAPHICS 的名称。一般在设定上，屏幕白色部分会印成黑色，而黑色的部分会印成白色。

/R 可用来使白色的印成白色，而黑色的印成黑色。

通常屏幕的背景是不会被印出来的，如果你配备彩色打印机则可以使用/B 来印出背景。

/LCD 是用来指定你所使用的是转换式的液晶显示器(Liquid Crystal Display)。

* HELP　用来显示 DOS 指令的辅助说明。

HELP [command]

INSTALL　用来装入 FASTOPEN、KEYB、NLSFUNC 和 SHARE 指令。

INSTALL filename

JOIN　用来连接两个文件系统。

JOIN joining—drive joined—drive\directory [/D]

* KEYB　用来选择键盘的类型，其使用格式如下：

KEYB keyboard_code,code_page path /ID:nnn/E

选项/E 是用来在 386 以上的计算机设定增强型键盘。例如：

KEYB FR /ID：120

就是用来选择法语键盘，并指定 120 为键盘识别码。

LABEL　用来指定、改变或删除磁盘名称。

LABEL [drive specifier] [name]

*LOADHIGH　用来装入程序到上端存储区中执行，以节省内存空间(与 LH 指令相同)。

LOADHIGH filename

MEM　用来显示有关计算机内存的信息，例如：

```
A:\>MEM

655360 bytes total conventional memory
655360 bytes available to MS- DOS
593120 largest executable program siize

393216 bytes total contiguous extended memory
393216 bytes available contiguous extended memory

A:\>
```

MIRROR　用来记录磁盘上的信息，以备 UNDELETE 和 UNFORMAT 两个指令调用。

MIRROR [/U] [/PARTN] [/1]

MKDIR　用来建立一个子目录(与 MD 指令相同)。

MKDIR path

*MODE　用来建立不同的输入与输入设备的操作，常用的格式有：

MODE video—mode

用来设定显示器的类型。

MODE CON RATE = rate DELAY = delay

用来设定键盘自动重复的速率及延迟时间。

MODE LPT# COLS = length LINES = lines per—inch

用来设定打印的宽度及每英寸的打印行数。

MODE COM# boud,parity,data bits,stop—bits

用来设定通信口的通信标准。

MODE LPT# = COM#

用来引导并行式打印机串行式打印机上。

MORE　用来显示整个屏幕的数据，也就是每次显示 23 行后会暂停下来。

MORE < filename

*NLSFUNC　用来指定 CHCP 指令所使用的国家码定义文件。

NLSFUNC [filename]

PATH　用来指定一个寻找的路径。

PATH path [;path...;path]

PRINT　用来控制 DOS 的打印功能。

PRINT filename filename,,,filename [/T] [/C] [/P]

PROMPT　用来改变 DOS 提示符，其使用格式为：

PROMPT prompt

prompt 提示符可从下表任选一个或组合使用：

代码	提示符
$$	货币符
$b	符号|
$d	系统日期
$e	符号←
$g	符号>
$h	不显示任何符号
$l	符号<
$n	当前磁盘代号
$p	当前目录的路径
$q	符号=
$t	系统时间
$v	DOS 版本号
$	自动按 Enter 键

最常用的一种提示符为显示当前目录的路径并紧跟符号>，例如：

PROMPT PG

* QBASIC　用来启动 QBASIC 语言。

QBASIC [/B] [/EDITOR] [/G] [/H] [/MBF] [/NOHI] [/RUN] filename

/B 在彩色屏幕上以黑白显示 QBASIC 的操作窗口。

/EDITOR 调用 EDIT 编辑程序。

/G 在 CGA 屏幕上使用 QBASIC。

/H 使用最大的屏幕列数。

/MBF 将 QBASIC 语言的 MKS$、MKD$、CVS 和 CVD 指令分别变成 MKSMF$、MKDMBF$、CVSMBF$ 和 CVDMBF$。

/NOHI 允许使用未提供高亮度效果的屏幕。

/RUN 直接装入 QBASIC 程序文件并执行它。

RECOVER　用来将受损的文件或磁盘恢复。

RECOVER [driver..specifier] [filename]

RENAME　用来更改文件名称(与 REN 指令相同)。

RENAME old name new name

* REPLACE　用来取代或新增文件。其使用格式如下:

REPLACE source destination [/S] [/A] [/W] [/P] [/U] [/R]

/S 表示所有子目录的全部文件也会被检查和取代。

/A 表示只把目标磁盘上不存在的文件加上去,以防止源文件被覆盖掉。

/W 暂停 REPLACE 指令的执行以便你插入新的磁盘,然后按下任一键开始执行。

/P 用来询问你是否要取代文件?

/U 表示只会取代文件建立时间较早的文件。

/R 表示只读文件要取代。

下例就是以驱动器 B 的文件来取代驱动器 A 上相同名称的文件,而且包括全部的子目录:

REPLACE B: A: /S

RESTORE　用来存回备份文件。

RESTORE backup fixed [/A:date] [/R:date] [/E:time] [/L:time] [/P] [/S] [/M] [/N] [/D]

RMDIR　用来删除一个子目录(与 RD 指令相同)。

RMDIR directory

SET　用来设定一变量名称到指令处理程序的环境中。

SET name = value

下例就是用来设定 APPSDAT 变量名称以便代替路径名称 \PROGRAM\APPS\

SET APPSDAT = \PROGRAM\APPS

如果要删除所设定的名称,只要使用下列的指令即可:

SET APPSDAT =

在批处理文件中使用设定的变量名称时,必须在变量名称前后加一个%号,例如:

COPY %APPSDAT% *.*

那么,当这个批处理文件被执行时,这个指令就变成:

COPY \PROGRAM\APPS*.*

* SETVER　用来设定程序文件的版本号,以便在 DOS 5.0 上执行它。由于某些程序文件必须在特定的 DOS 版本上(例如 4.01 或 3.3)才能执行,如果在 DOS 5.0 上要执行此程序,则必须先以 SETVER 指令来指明它的版本号才能执行。

SETVER [filename n.nn]

SETVER [filename [/DELETE [/QUIET]]

n.nn 表示版本号。

/DELETE 删除先前所设定版本号的程序文件。
/QUIET 删除时不显示提示信息。

* SHARE　用来准备在网络系统上共享文件和锁定的功能。
SHARE [/F:filespace] [/L:locks]

SORT　用来排序数据。
SORT [<input [>output] [/R] [/+num]

SUBST　用来建立不同的名称来取代驱动器或路径名称。
SUBST nickname drive specifier path

* SYS　用来将 DOS 的操作系统拷贝到磁盘上。
SYS drive specifier

TIME　用来显示和改变时间。
TIME [hh:mm:ss]

TREE　用来显示磁盘上的全部目录。
TREE drive specifier [/F] [A]

TYPE　用来显示文本文件的内容。
TYPE filename

UNDELETE　用来恢复先前被删除的文件。
UNDELETE [filename] [/LIST | ALL] [/DOS | DT]

UNFORMAT　用来恢复先前已被格式化的磁盘。
UNFORMAT drive [/J] [/U] [/L] [/TEST] [/P] [/PARTN]

VER　用来显示 DOS 的版本号。
VER

VERIFY　用来确认数据是否正确的被存储到磁盘上。
VERIFY [ON/OFF]

VOL　用来显示磁盘名称。
VOL [drive—specifier]

* XCOPY　用来拷贝一组文件以及较低层次的子目录，其使用格式为：
XCOPY source target [/A] [/D:date] [/E] [/P] [/S] [/V]
[/W] [/M]

/A 用来拷贝存取属性为打开状态的文件，而且不会改变存取字符的状态。
/D:date 用来拷贝与指定日期相同或较早的文件。
/E 用来建立全部的子目录，即使是空的。
/M 用来拷贝存以属性为打开状态的文件，然后设定存取字符为 0。
/P 拷贝任何文件前会显示提示信息。
/S 拷贝子目录内的文件。
/V 确认数据写入是否正确。
/W 等待直到插入磁盘。
下例会把驱动器 A 的全部文件包括全部的子目录拷贝到驱动器 B 上：

XCOPY A：B:\ /S

以下为行编辑程序 EDLIN 的指令摘要：

EDLIN　　用来启动行编辑程序 EDLIN。

EDLIN filename

下表为 EDLIN 所提供的编辑指令：

指令	功能
A	增添文字行
C	复制文字行
D	删除文字行
E	终止 EDLIN 并存储文件
I	插入文字行
L	列出文字行
M	搬移文字行
P	一次显示一个屏幕的文字行
Q	终止 EDLIN 但不存储文件
R	以一字符串取代另一字符串
S	搜寻某一字符串
T	从磁盘拷贝一文件内容
W	写入文字行
line_num	编辑文字行

以下为批处理文件的指令摘要：

CALL　　用来调用另一个批处理文件。

CALL batch file

ECHO　　是否显示控制屏幕。

ECHO [ON] [OFF] [message]

FOR　　反复执行一个 DOS 指令。

FOR %%var IN (argument—list) DO command

GOTO　　无条件地改变执行的流程到某一标记上。

GOTO label

IF　　有条件地执行 DOS 指令。

IF condition command

PAUSE　　暂停批处理文件的执行。

PAUSE [message]

REM　　当 ECHO 指令是 ON 状态时，可用来显示说明的信息。

REM [remark]

SHIFT　　改变变量%0 到%9 的值。

SHIFT

: 用来指定不被 DOS 执行的注解文字。

: comment

以下为配置文件 CONFIG.SYS 的命令摘要：

BREAK 用来改变控制中断(Ctrl+Break)键的检查。

BREAK = [ON | OFF]

BUFFERS 用来指定磁盘缓冲区的大小。

BUFFERS = number

COUNTRY 用来设定时间和日期格式，还有金额和小数点的分隔符号。

COUNTRY = code

DEVICE 用来指定设备驱动程序文件。

DEVICE = device—driver

DEVICEHIGH 用来装入设备驱动程序到上端存储区。

DEVICEHIGH =device· driver

DOS 用来装入部分的 DOS 到上端存储区。

DOS = HIGH/LOW

DRIVEPARM 用来定义参数以取代当前系统的定义。

DRIVEPARM = drive number

FCBS 用来设定文件控制表所能打开的最大文件数。

FCBS — number

FILES 用来设定同时能够打开的最大文件数。

FILES = number

INSTALL 用来装入 DOS 的 FASTOPEN、KEYB、NLSFUNC 和 SHARE 指令。

INSTALL filename

LASTDRIVE 用来设定驱动器代号的最大值。

LASTDRIVE = letter

REM 用来加入注解文字到配置文件中。

REM [comment]

SHELL 用来指定最高层次的指令处理指令。

SHELL = filename

STACK 用来改变堆栈(Stack)资源。

STACK = n,s

SWITCHES 用来强迫增强型的键盘如同一般的键盘功能。

SWITCHES = /K

以下为 DOS 6.0 新增的指令摘要。指令左上角注有"*"号者表示该指令并未在本书中提到，请利用 HELP 或 FASTHELP 联机帮助说明的功能，来取得详细的介绍及使用说明。

CHOICE 用来显示指定的提示信息并暂停等候按键。

*DBLSPACE 用来增加磁盘的存储容量。

*DEFRAG 用来整理存储在磁盘中的文件，使其发挥最佳的存取功能。

DELTREE	用来删除子目录的内容。
* FASTHELP	用来快速显示 DOS 6.0 的全部指令名称及其扼要说明。
* INTERLNK	用来驱动两台已连接成可互为主从的计算机网络，以便进行数据的传送。
* INTERSVE	用来启动 INTERLNK 主端(即服务站端)的网络功能。
* LOADFIX	用来装入程序到常规内存前 64K 上并执行它。
* MAIL	用来传送与接收电子邮递的信息。
MEMMAKER	用来协助常规内存的最优化，使程序的执行效率加快。
MOVE	可在不同子目录间搬移文件。
* MSAV	用来检测及清除计算机病毒。
MSBACKUP	以窗口的操作方式来进行磁盘文件的备份与回存。
* MSCDEX	用来驱动只读式的光盘驱动器。
* MSD	用来诊断计算机的硬件及外围设备。
POWER	可更有效率地保存笔记本计算机的电池电力。
* VSAFE	用来监视计算机病毒。

以下为 DOS 6.0 增强功能的指令清单，这些指令的用法已在本书中介绍过，因此只列出指令名称供你参考。

DEVICEHIGH

EMM386

HELP

LOADHIGH(LH)

MEM

SMARTDRV

UNDELETE

多配置文件命令：

INCLUDE

MENUDEFAULT

MENUCOLOR

MENUITEM

SUBMENU

附录 B EDLIN 指令的使用

B.1 EDLIN 简介

在本附录中，你将学习如何使用行编辑程序 EDLIN。这个程序可用来对源程序(Source Program)或文本文件(Text Files)做建立、改变内容及显示文件等功能。也就是你可使用 EDLIN 来做：

1. 建立新的源文件，并存储它们。
2. 更新已存在的文件，并存储新文件和旧文件。
3. 针对文件中的每行可做删除、编辑、插入及显示的功能。
4. 可在一行到多行内做寻找、删除和取代某些指定的文字。

EDLIN 是以行为单位来处理文本文件或源文件，而每行最多可容纳 253 个字符。在使用此行编辑程序时，将会看到行号显示在每行的开头，此行号是在编辑过程中所产生的，以便用户参阅，而在实际存储文件时不会把它存入磁盘中。

当你做编辑而插入一行或多行文字时，在插入后，所有行号将自动累加所加入的行数目。同样，在删除一行或多行文字时，也会将其后的所有行号依删除的行数而降低。所以不论如何编辑，在文件中的行号必须是连续的。

B.2 如何启动 EDLIN

要启动行编辑程序(EDLIN)，请在 DOS 提示符下输入：

EDLIN 〈filespec〉

如果你要建立一个新文件，那么其上的〈filespce〉将是你所要建立的文件名称。如果行编辑程序在指定的驱动器上找不到你指的文件名，它便自动视为你要建立一个新文件。在建立新文件时，将会出现下面的信息和提示：

```
new file
*
```

注意：在 EDLIN 中的提示符为星号(*)。这时你就可输入新文件的内容，但在此操作前需输入一个插入命令“I”，然后按 Enter 键。这“插入命令”稍后将讨论。

如果你要编辑一个已存在的文件，那么〈filespec〉将是你所要编辑的文件名。当行编辑程序在指定的驱动器上找到你所指定的文件名，则这个文件立刻被装入到内存中，若整个文件不是很大而能完全装入，那么行编辑程序将在屏幕上显示下列信息：

```
End of input file
*
```

这时你就可利用行编辑程序的编辑命令来编辑这个文件了。

如果要装入的文件太大而无法装入内存时，行编辑程序将只装入部分行，直到装满内存的3/4为止，然后显示星号，这时你可编辑那些存在内存中的行。要编辑剩下的文件，务必要先存储已编辑过的行到内存中。此过程请参考稍后描述的 W 及 A 编辑命令。

当你完成了所要的编辑后，可利用结束(END)命令把编辑前与编辑后的两个文件存储起来。结束命令在稍后也会讲到。这两个被存储的文件一个是编辑前的，将会以.BAK 的文件扩展名存储，而编辑后的文件则延用你编辑时给它的完整文件名。原有的.BAK 文件将继续存在磁盘上，当你确定不再需要它时，可以利用 DEL 指令将它删除，以节省磁盘的存储空间。

注意，不可编辑一个文件扩展名为.BAK 的文件，因为行编辑程序认为.BAK 文件为备用的文件。如果因某种原因必须编辑它时，必须重新命名这个文件的扩展名(可用 DOS 中的 RENAME指令)，然后才可编辑此文件。

B.3　EDLIN 的使用

使用 EDLIN 编辑程序时常需配合 DOS 的特殊编辑键，来编辑文本文件或源程序文件。现在让我们实际来看看如何使用键盘来执行 EDLIN，比如要建立一个新文件，其文件名为 MYFIRST.TXT，就可按照以下方法启动 EDLIN：

```
A:\>EDLIN MYFIRST.TXT
```

EDLIN 提示符出现后，输入字母 I，就是输入文字插入命令，目的在于告诉 EDLIN 执行后面插入的文字行工作。到目前为止，文件仍然是空白的，屏幕上应显示如下：

```
New file
*I
     1:*
```

1:* 指 EDLIN 会将你所输入的数据放在新文件的第一行。现在请输入如下各行文字：

```
1:*I AM TESTING THE EDLIN
2:*LINE 1
3:*LINE 2
4:*LINE 3
5:*LINE 4
6:*LINE 5
7:*LINE 6
8:*LINE 7
9:*LINE 8
10:*LINE 9
11:*LINE 10
12:*^C(按 Ctrl+C 键结束插入文字行)

*E(按 E 结束编辑工作)

A:\>DIR MYFIRST.TXT
```

```
Volume in dirve A is BATCH DISK
Directory of A:\

MYFIRST      TXT        106   07-04   92    9:44a
        1 file(s)        106 bytes
                      111616 bytes frec

A:\>TYPE MYFIRST.TXT
I AM TESTING THE EDLIN
LINE 1
LINE 2
LINE 3
LINE 4
LINE 5
LINE 6
LINE 7
LINE 8
LINE 9
LINE 10
A:\>
```

可以看到，文件目录中不但已经建立了一个文件名为 MYFIRST.TXT 的文件，而且其所存储的内容就是我们刚刚输入的各行文字。

B.4 EDLIN 的命令摘要

EDLIN 共有两种基本操作模式。第一种模式是在 EDLIN 提示符(*)出现，而无行号出现时，输入 EDLIN 的命令而产生，例如刚才的 * 后所输入的 I(插入)和 E(结束)命令。

在第二种模式下，EDLIN 所接受的文字会变成编辑文件内容的一部分。如前节所述，在此模式下，EDLIN 提示符前紧跟冒号和行号。

EDLIN 命令是由一个带有参数，或不具有参数单独字符(大写或小写)所组成。此命令在按下 Enter 键后就会被执行。执行 EDLIN 命令时，按下 Ctrl+Break 键，就能终止此命令的执行，并且回到 EDLIN 命令提示符下。按下 Ctrl+Numlock 键，就可暂时停止输出，以便查看显示状况，之后再按任意键，该命令就会继续执行。

通常只有两个行号都为命令参数时，才有必要使用分隔符(如空格或逗号)。否则，分隔符都会被忽略。现将各 EDLIN 命令参数列于下表：

参数	定义
line (行号)	参数"line"代表编辑文件的行号。此行号参数的范围为 1 到 65529 的整数。句点(.)表示当前行号,而#则表示内存中最后一个行号。如果你输入的行号大于文件内的行号,其效果就与输入#号相同。行号前也可附加正负号,正号表示当前所在的行之后的行号,负号表示当前所在行号之前的行号。
n	"n"代表要由磁盘转移到存储器,或由存储器转移到磁盘的文字行数。当编辑文件太大不能全部放入存储器时,此参数就与 APPEND 及 WRITE 命令共同使用,以便执行编辑命令。
string (字符串)	"string"代表字符组成的字符串,此字符串就是搜寻文字(search text,S)及代换文字(replace text,R)命令中,被 EDLIN 当作搜寻或代换的字符串。

EDLIN 命令能够在编辑文件中任意插入文字行、查看某个范围内的文字行、删除文字行、搜寻及置换文字以及将编辑过的文件存回。为方便起见,也可以在一行中使用几个命令,只需用分号(;)将各个命令隔开。各命令的功能详述如下。

1. 插入文字行命令——I(Insert)

插入文字行命令(I)可在文件中插入几行文字。此命令的格式如下:

[line]I

若省略参数 line,则所输入的几行文字就被插入当前所在文字行后,若指定了行号 line,新插入的文字行就被放在指定行号的前面,直到此命令终止为止。在此命令下,若按 Enter 键,就可以将显示的文字行送入编辑文件,并使 EDLIN 产生下一行的行号。

当按下 Ctrl+Break 键,插入文字行命令就会终止其操作。因为在文件中加插了几行文字,所以在其之后的文字行全部都会改变行号。

2. 编辑文字行命令

编辑文字行命令能从文件指定一行文字予以编辑,此命令的格式如下:

[line]

若指定了行号 line,则该行文字就被显示出来,应用 DOS 编辑键就可编辑此行文字,或重新再输入此行。

如果此命令未指定行号 line,也就是在 EDLIN 命令提示符后只按下 Enter 键来响应,则显示下一行文字,以供编辑。

如果你不想变更所指定的一行文字,可让显示光标移到文字行开头,再按下 Enter 键。若按下 ESC 或 Ctrl+Break 键,也能使存在文件中的文字行保持不变。

假设希望在文本文件中加入控制字符,只要先输入 Ctrl+V,而后接着输入大写字母即可。例如,Ctrl+V Z 就可将^Z 加入文件中。

3. 列出文字行命令——L(List)

列出文字行命令(L)的作用,是显示编辑文件内某个范围的文字行。此命令的格式如下:

[line][,line]L

此命令若无参数，就显示出 23 行文字(以当前文字行为中心)。若第一个行号参数省略，也就是输入命令为：

,line L

则显示的文字行就是当前所指行号前面的第 11 行开始，直到指定的行号为止。

若本命令只包含第一个行号参数，也就是输入：

line L

则显示 23 行文字，且以指定行号为起始的第一行。

如果命令中包含两个行号参数，则指定范围内的各行文字就被显示出来。

4. 分页命令——P(Page)

分页命令(P)与列出文字行命令(L)相同，只是执行分页命令后，当前所指行号会有所改变。此命令的格式如下：

[line][,line]P

分页命令执行后，最后一行显示出的文字就成为现在所指行号。此命令一次可显示一页(23 行)文字，因此非常有用。

5. 复制文字行命令——C(Copy)

复制文字行命令(C)可以复制一行或几行文字，并将这些文字放在指定文本文件中的某个地方。此命令的格式如下：

[line],[line],line[,count]C

若未指定以上格式的前两个行号参数，就自动以当前所在行号取代。否则，这两个行号就代表要被复制文件的行号范围。第三个行号参数为被复制的各行文字要放置所在的行号。例如，你想把第 1 行到第 6 行的文字复制到第 10 行，则第 1 行到第 6 行的文字就成为第 10 到第 15 行，而原来的第 10 行就变成第 16 行。

此命令若包含有计数参数"count"，则此复制操作就执行 count 指定的次数。若省略此参数，则复制只执行一次。

6. 删除文字行命令- D(Delete)

删除文字行命令(D)可删除一行或几行文字，其格式如下：

[line],[,line]D

以上两个行号参数用来指定要删除文字的行号范围。若省略其中的第一个参数，就自动以当前所在行号来代替。若省略后者，则只删除第一个参数所指定的文字行。删除完后，跟随在被删除文字行后面的一行文字就成为现在所指行号。

7. 搬移文字行命令- M(Move)

搬移文字行命令(M)可将一行或几行文字从当前位置移到另一个指定位置。此命令的格式如下：

[line],[line],line M

本命令中的前两个行号参数用来指定要被搬移文字的行号范围。若予以省略，就自动设定

为当前所在行号。第三个行号代表文字区段(text block)的目的地行号。比如，你想将第 10 到 12 行的文字移到第 3 行的位置，则各个被移动文字就在其原始位置消失，并在指定的新位置出现，同时因为这种文字搬移，行号也跟着调整，也就是原先第 3 到第 9 行的文字，被调整为第 6 到第 12 行。

8. 搜寻文字命令——S(Search)

搜寻文字命令(S)的作用是在文件中寻找某一特定的字符串行。此命令的格式如下：

[line][,line][?] S string

指定被搜寻字符串的搜寻文字行范围，可以用包含两个行号参数的命令来规定(注意：string 参数并不加前后双引号来标示，但若搜寻字符串为此字符串的一部分时就需加前后双引号)。此命令省略第一个行号参数时就设定行号为 1，若省略第二个行号参数，就设定参数值为文件的最后一个行号。若两个参数都省略，就搜寻整个文件。

若省略命令中的问号，则第一次搜寻到所指定的字符串就停止搜寻，并且将该行文字设定为现在所指行号。若未搜寻到此指定字符串，屏幕上便显示"Not Found(搜寻不到)"的信息，并终止此命令。

如果 S 命令附有问号(?)，则在搜寻到指定字符串时，EDLIN 便会显示字符串所在的行号及信息"OK?"。如果输入 Y 或 Enter 键，搜寻工作便立刻结束，而显示文字行就成为当前文字行。此时若按下任意键，搜寻就继续产生作用。

请注意，指定字符串是由紧随在 S 命令右边的各个字符所组成，此字符串以按下 Enter 键终止。

9. 代换文字命令——R(Replace)

代换文字命令(R)的作用与搜寻文字命令(S)相似，只是在此命令中，指定字符串是以命令中的另一个指定字符串代换。

此命令的格式如下：

[line][,line][?] R string [〈F6〉 string]

规定第一个字符串参数搜寻范围的方式与搜寻文字命令(S)完全相同，如果命令中包含了问号，则在第一次搜寻到指定字符串后，立即停止搜寻。此时若输入 Y 或 Enter 键来响应信息"OK?"，第二个指定字符串参数就会代换第一个字符串。如果搜寻停止后，按下其他键，就会继续搜寻，并且不会代换当前搜寻到的字符串。在上述两种情况下，搜寻会持续到遍布整个指定范围为止。

在此第一个字符串参数都是在 R 后面输入，且以按下 F6 来结束此字符串。同样，代换字符串也是紧跟在 F6 键后立刻输入，按下 Enter 键，此代换字符串就终止。如果省略代换字符串部分，则指定字符串在搜寻到后便立刻删除。

10. 传送文字行命令——T(Transfer)

传送文字行命令(T)能够读取磁盘文件的内容，并合并到当前正在编辑的文本文件中。此命令的格式如下：

[line] T (filespec)

line(行号)参数用来指示新文字行的位置。若将其省略,就自动设定为当前文字行。filespec参数是用来指所要读取的文本文件名称。

11. 写入文字行命令——W(Write)

写入文字行命令(W)是在编辑文件太大而无法全部放入内存时使用,也就是在启动EDLIN,却得不到下列信息时使用:

```
End of input file
*
```

或

```
new file
*
```

我们通常把要编辑的文件装入内存,直到内存被装入75%为止,这时就可对装入的文字进行编辑。编辑过后,就必须使用W命令转移编辑过的数据回到磁盘,以便腾出空间存放其他的文字行。

此命令的格式如下:

[n]w

若省略n,则从1开始的各文件文字行就被写入磁盘,直到内存只剩下25%的空间存有文字时为止。如果命令中有n,则行号1到n的文字就写入磁盘。

12. 增添文字行命令——A(Append)

在写入文字行命令腾出内存空间后,就可以使用增添文字行命令(A)来使文件内容的其他(下一个)部分装入内存。此命令的格式如下:

[n]A

在此若未指明n值,则文字行就持续装入内存,直到内存占满75%或达到文件末尾为止。若指定了n值,则将指定行数的文字增添到内存最后一行之后。

当整个文件都被增添文字行命令选取完毕后,屏幕上会显示出"End of Input File"的信息。

13. 终止编辑命令——E(End)

终止编辑命令(E)能将被编辑的文件写回磁盘中。在进行编辑时,所有编辑的过程都在存储器内部进行。发出终止编辑命令后,源文件的名称作如下变动:其文件名仍然不变,但扩展名改为.BAK。编辑过的文件存入磁盘后,就沿用源文件的整个名称。如果新磁盘放在存取的驱动器内,编辑过的文件就写入此磁盘,其文件名就与启动EDLIN时所指定的名称相同。

简单终止编辑命令的格式如下:

E

当然,你仍需确定磁盘是否有足够的空间来存入文件,否则将使编辑过的某些部分因此丢失。

终止编辑命令能使EDLIN完全结束,并回到DOS系统提示符下。

14. 退出编辑命令　Q(Quit)

退出编辑命令(Q)能结束当前 EDLIN 操作,但不存储编辑过的文件。此命令的格式如下:

Q

在结束编辑前,EDLIN 会询问是否真的放弃先前的编辑:

Abort edit(Y/N)?

若输入 Y,则停止编辑。输入其他键,则回到 EDLIN 命令提示符,以继续编辑。

使用终止编辑命令来停止编辑后,被编辑的文件不留下任何备份文件(但源文件仍保持其原状)。

B.5　EDLIN 的错误信息

当行编辑程序发现有错误时,下面所列错误信息中的一种将被显示出来:

Cannot edit .BAK file...rename file

原因:你企图去编辑一个扩展名为.BAK 的文件。由于.BAK 的文件被视为备用文件,因此不可去编辑它。

解决:如果你为了要编辑而需使用.BAK 文件,则可把该文件另外拷贝一份并更改其扩展名,或者改成另外的文件名或扩展名,然后再编辑这个新文件。

No room in directory for file

原因:当你企图建立一个新文件时,可能是文件目录已满或指定了一个不正确的驱动器,也可能是一个不合规则的文件名,将造成此错误。

解决:检查刚才所输入的命令语句,是否在启动行编辑程序(EDLIN)时,使用了不正确的驱动器名称或不合规则的文件名。如果此命令已没有显示在屏幕上,而你还未输入新的命令语句前,可使用 F3 键把该命令从语句存储区调出来看看。

如果依上面的方法发现语句并没有错误,则可执行 CHKDSK 指令来检查所指定驱动器,若所报告的状态显示磁盘已满,则移走此磁盘,换一张已格式化的新磁盘。

Entry Error

原因:刚才所输入的命令语句有语法上的错误。

解决:以正确的语法重新输入,然后按 Enter 键。

Line too long

原因:在"代换命令(R)"语句中由于取代的字符串加入,造成此行超过 253 字的极限。行编辑程序将终止此取代命令。

解决:把它分成两行(可用 F5 键),再执行一次。

Disk full... file write not completed

原因：当你使用“结束命令(E)”时，磁盘内剩余空间不足以把全部文件容纳进去。在这种情况下，行编辑程序将终止此项命令，然后回到操作系统。但会有部分的文件被写在磁盘上。

解决：这种错误使被写入磁盘的那部分文件完全无用。因此你可删除已写到磁盘上的那部分文件，然后重新开始编辑。当你在编辑后，要把文件写到磁盘前，必须先确定磁盘中有足够的空间可容纳该文件。

Incorrect DOS version

原因：你试图在不是 DOS V2.0 版或更高版本的操作系统下执行行编辑程序(EDLIN)。

解决：你必须确认现在所用的是 DOS V2.0 或更高版本的操作系统。

Invalid drive name or file

原因：在启动行编辑程序时，没有指定一个正确的驱动器名称或文件名。

解决：指定正确的驱动器名称及文件名。

Filename must be specified

原因：当你启动行编辑程序时，没有指定文件名。

解决：指定文件名。

Invalid parameter

原因：当你启动行编辑程序 EDLIN 时，使用了"/B"以外的选项。

解决：改用"/B"选项。

Insufficient memory

原因：当执行 EDLIN 行编辑程序时，已没有足够的内存空间。

解决：需把内存中的一些文件存回磁盘以空出空间，或在启动 EDLIN 时，先删除一些文件。

File not found

原因：使用“传送命令(T)”时在磁盘中找不到所指定的文件名。

解决：当使用“传送命令”时，指定一个存在而且正确的文件名。

Must specify destination number

原因：当执行“复制”或“移动”命令时，未指明目标行。

解决：重新输入有目标行的命令语句。

Not enough room to merge the entire file

原因：在执行“传送命令(T)”时，没有足够的内存空间来容纳要加入的文件。

解决：必须把一些文件从存储器中存回磁盘去，以腾出一些空间，或者把一些文件删除，再去传送文件。

File creation error

原因：没有办法建立一个行编辑程序用的临时文件。

解决：检查并确认是否磁盘目录中有足够的空间来建立临时文件。同时，也要确定是否在磁盘目录中的子目录有相同的名称。

附录 C EDIT 编辑程序的使用

C.1 EDIT 编辑程序简介

EDIT 编辑程序是 DOS 5.0 和 6.0 提供的一个崭新的全屏幕文本编辑程序，它完全是以窗口操作的方式来进行编辑的工作，易学好用，使你在编辑程序或文本时可节省很多时间。

在这个附录中，你将学习如何使用 EDIT 编辑程序。此程序也可用来对源程序(Source Program)或文本文件(Text File)做建立、改变内容及显示文件等功能。也就是说你可以利用 EDIT 来做：

1. 建立新的源文件，并存储它们。
2. 更新已存在的文件，并存储新、旧文件。
3. 针对文件内容做删除、拷贝、插入及显示的功能。
4. 可在整个文件内做寻找、删除和取代某些指定的字符串。

C.2 启动 EDIT 编辑程序

要启动 EDIT 编辑程序，只要在 DOS Shell 界面的主控程序窗口中选用 Editor 这个项目，或者直接在 DOS 提示符下输入 EDIT，然后按 Enter 键，就有如图 C—1 所示的菜单：

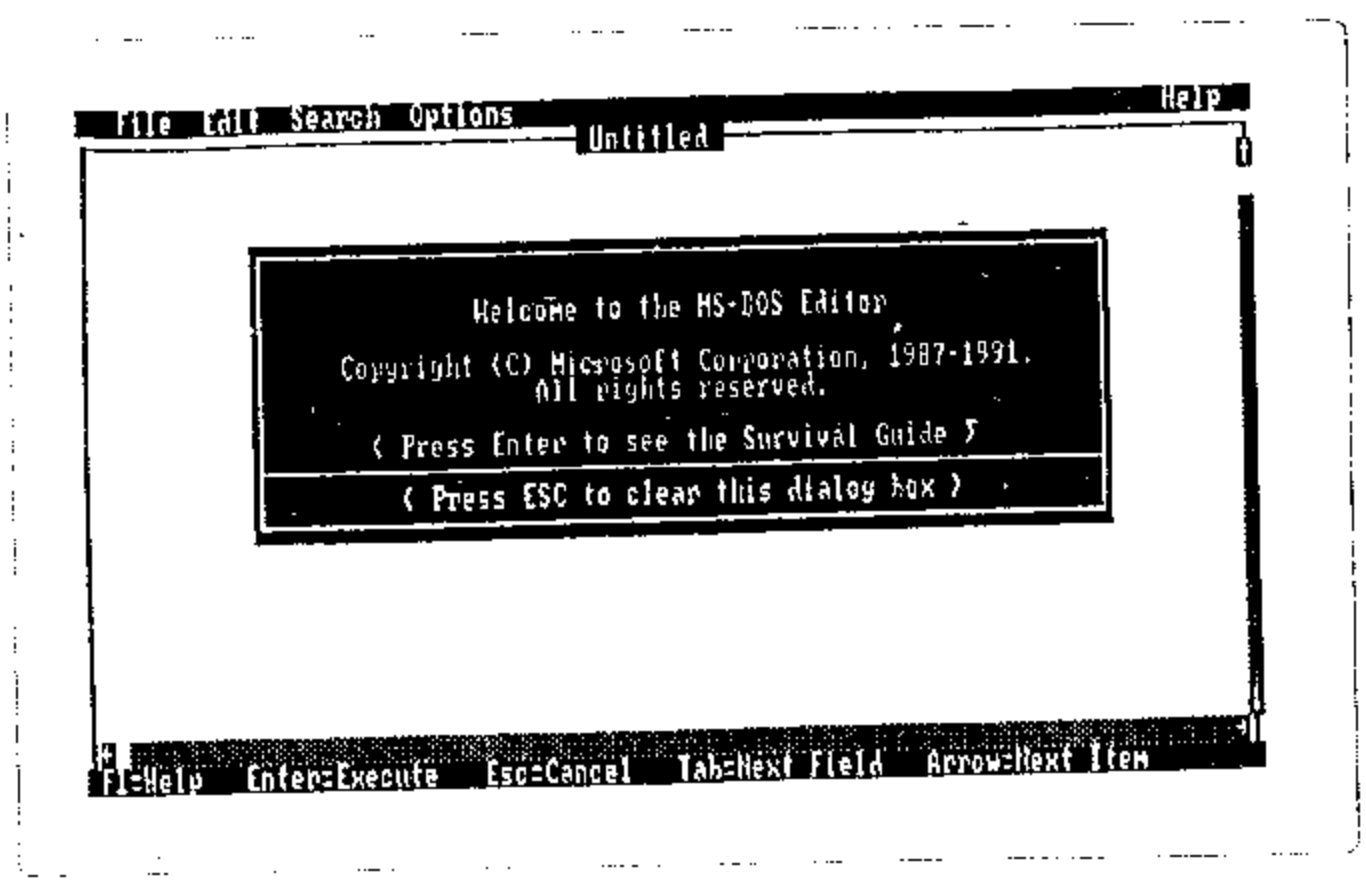


图 C—1

当你按下 ESC 键后，屏幕会显示如图 C－2 的提示方框：

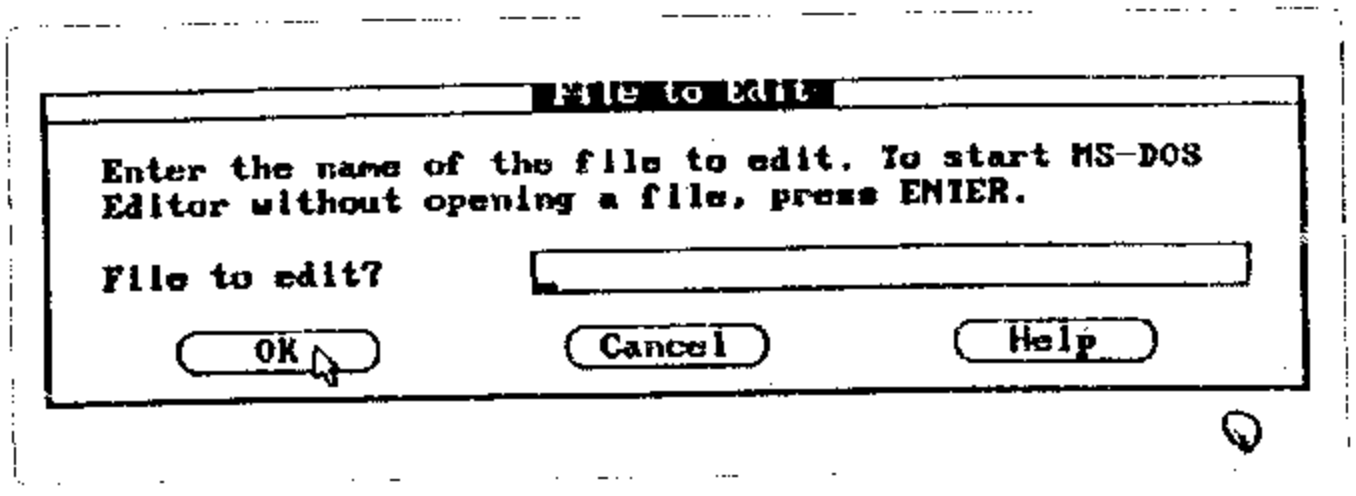


图 C－2

在图 C－2 的方框中，你必须输入你所要编辑的文件名称，你可以输入一个新的文件名来建立新文件，或是输入一个旧的文件名以便进行更新的工作。你也可以不输入文件名直接按 Enter 键，等到编辑完成后再输入文件名来存储你所编辑的新文件，为了说明起见，请你输入 MYTEXT. TXT 的文件名，然后按下 Enter 键，稍后屏幕就会显示如图 C－3 所示的编辑窗口：

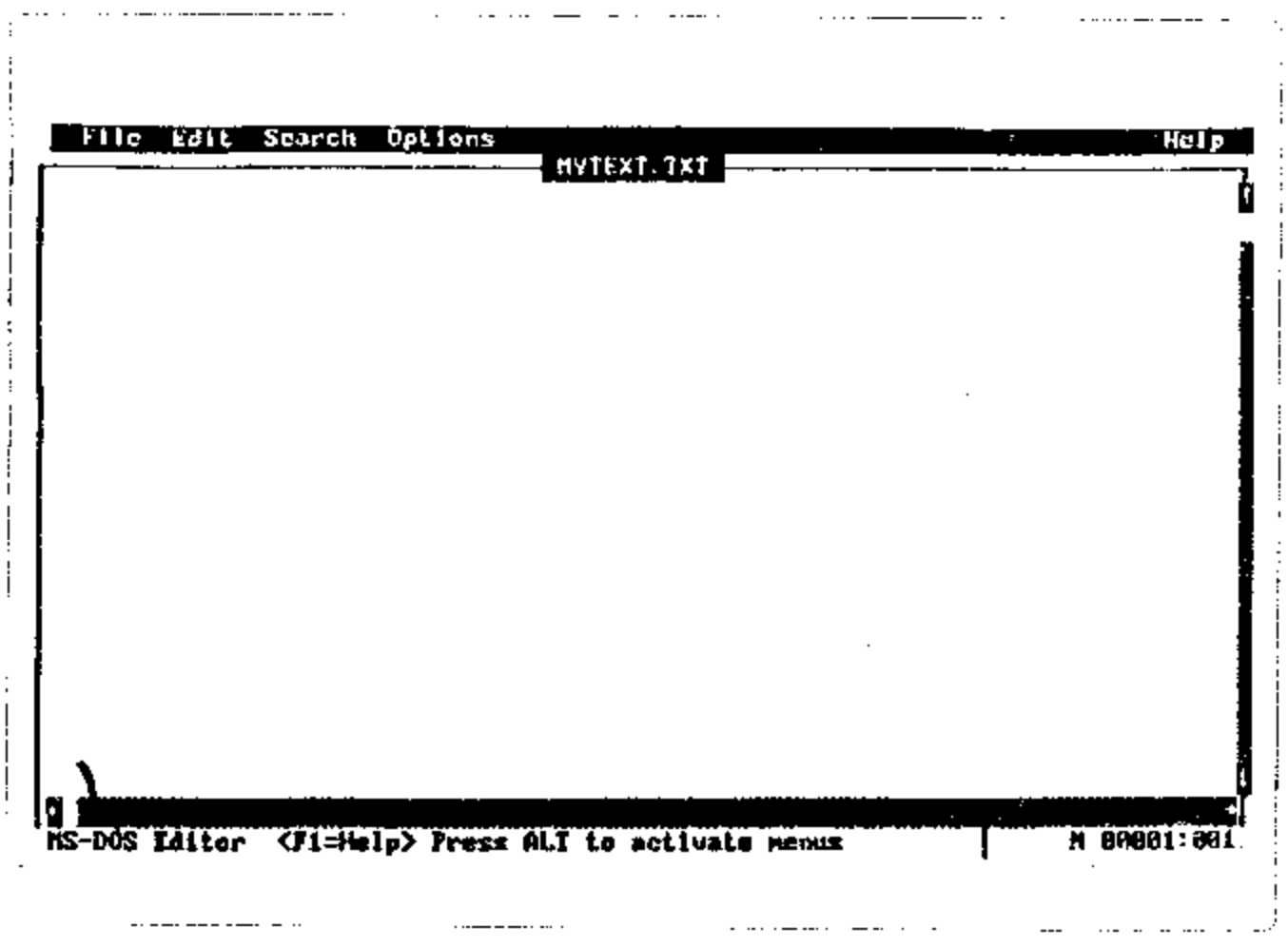


图 C－3

这个编辑窗口和 QBASIC 控制中心的编辑窗口大同小异，使用的技巧也几乎相同，因此操作起来非常容易。

在这个窗口上方就是 EDIT 编辑程序的目录栏，由左到右分别是：File、Edit、Search、Options和 Help，你一样可以利用 Alt 键来启动目录栏，以便选用各目录中的项目来进行编辑的工作。在目录栏下方中间的地方是名称栏，也就是显示你正在编辑的文件名，当前所显示的是 MYTEXT. TXT 这个文件名。屏幕中央的窗口是供你编辑的地方，当前光标闪烁处就在窗口的左上角，也就是你开始输入数据的地方。在窗口的最底端是状态栏，在此栏中会显示当前的操作状态及各功能键和选择项目的功能说明。而在窗口的右边和状态栏的上方则是卷动栏，可以用来上下卷动编辑窗口，以便显示整个文件的内容。

为了以后便于操作说明，目录栏中各个下拉式窗口中的内容分别如图 C—4、图 C—5、图 C—6、图 C—7、图 C—8 所示，并分别说明各个项目的功能。实际的操作将在后面各节中详细的介绍。

1. File—— 文件目录

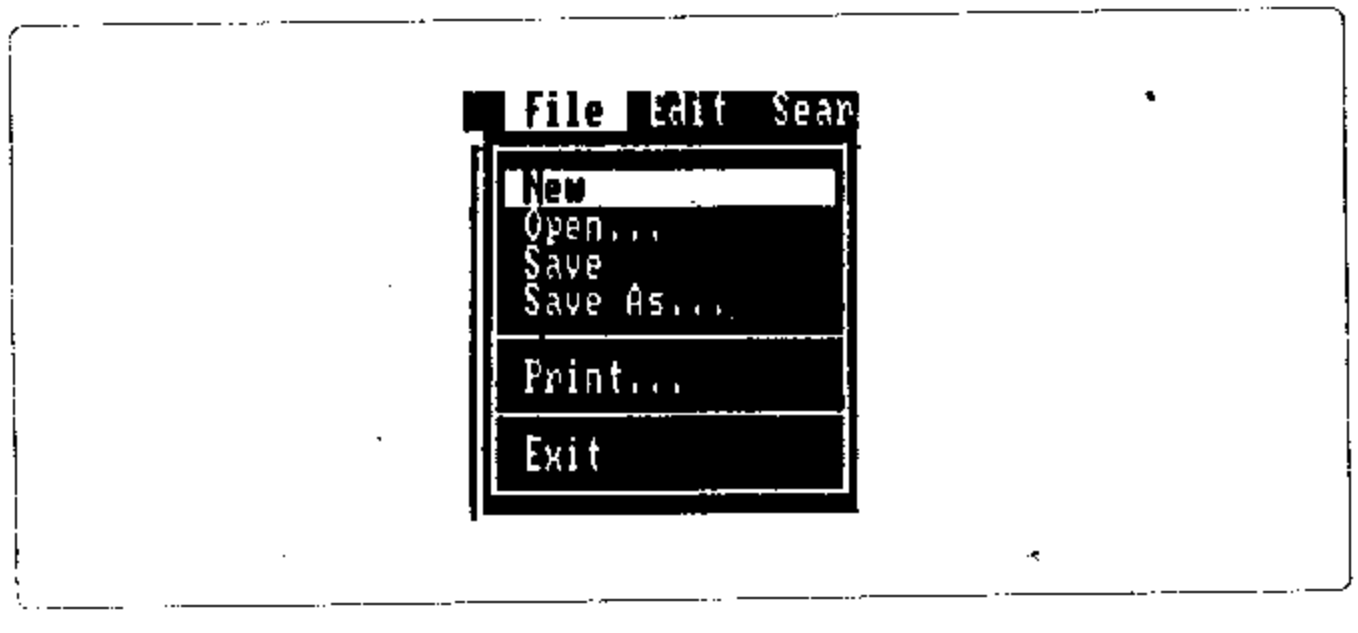


图 C—4

各个项目的功能分述如下：

New —— 开始编辑新文件。

Open —— 打开存储在磁盘上的文件。

Save As —— 将当前正在编辑的文件存储到磁盘文件中。

Print —— 将当前正在编辑的文件输出到打印机上。

Exit —— 结束 EDIT 编辑程序的操作，然后回到 DOS 提示符或 Shell 界面。

2. Edit——编辑目录

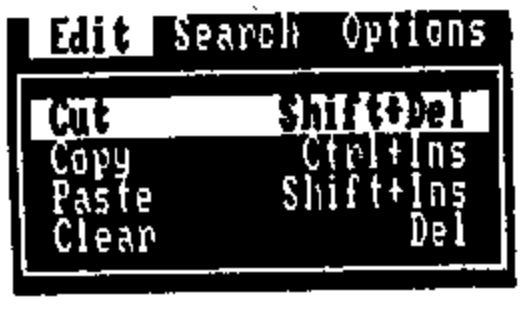


图 C-5

各个项目的功能分述如下：

Cut　—— 删除选定的数据块。

Copy　—— 拷贝选定的数据块。

Paste　—— 插入选定的数据块。

Clear　—— 删除选定的数据块。

3. Search——寻找目录

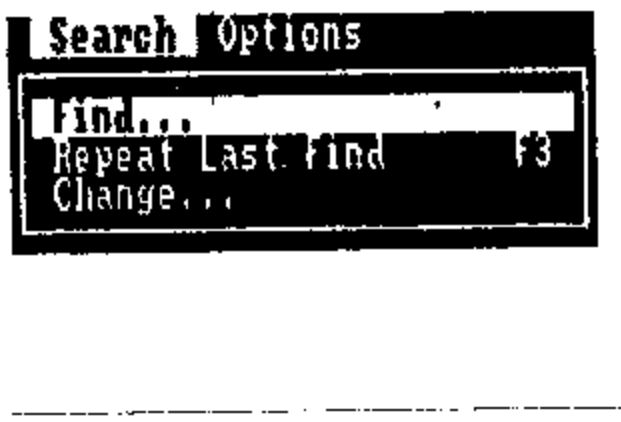


图 C-6

各个项目的功能分述如下：

Find　—— 寻找所要的字符串。

Repeat Last Find　—— 继续上次所指定的寻找。

Change　—— 寻找所要的字符串并修改它。

4. Options——选择目录

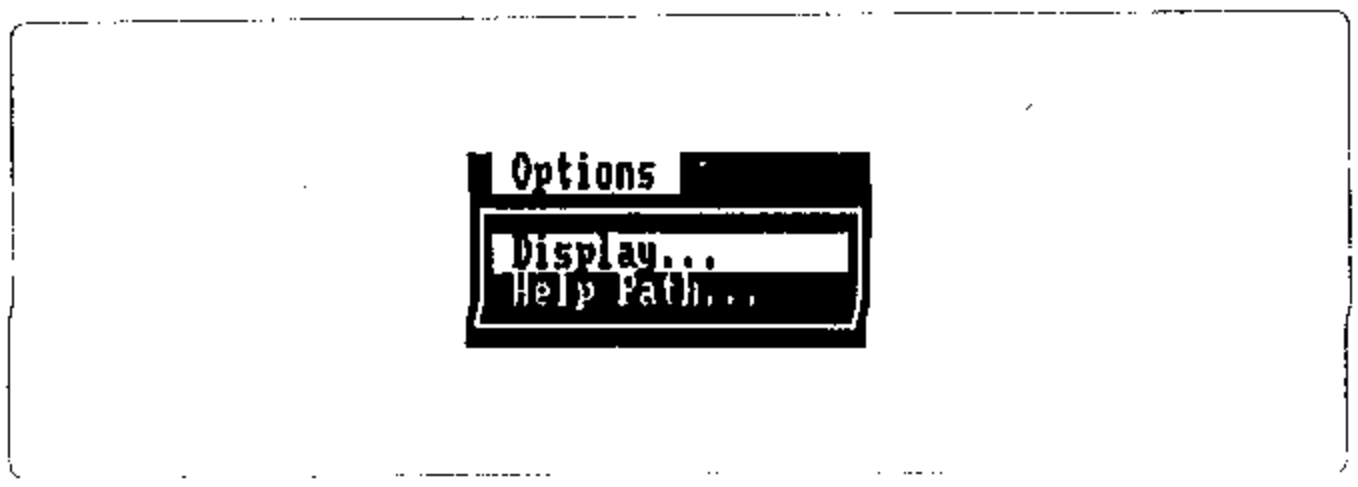


图 C—7

各个项目的功能分述如下：

Display —— 设定屏幕的显示特性，例如前景和背景的颜色，是否使用滚动条以及 TAB 键的定位。

Help Path —— 指定 EDIT.HLP 文件的路径。

5. Help 辅助目录

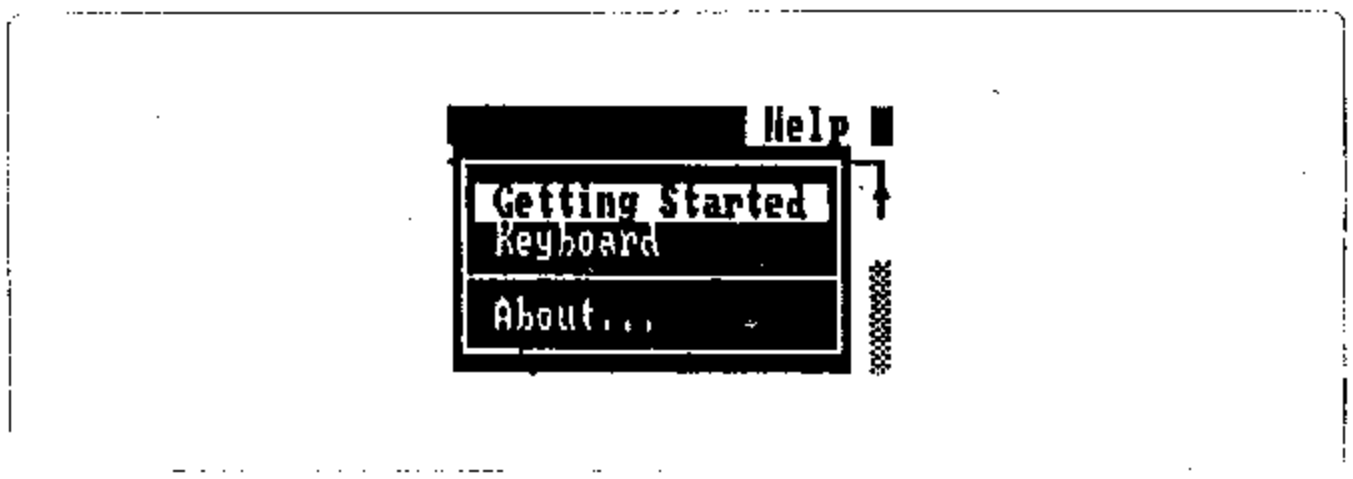


图 C—8

各个项目的功能分述如下：

Getting Started —— EDIT 编辑程序的操作说明。

Keyboard —— EDIT 编辑程序各按键的功能。

About —— EDIT 编辑程序的版权菜单。

C.3 EDIT 指令的操作

现在，请你从光标闪烁处开始输入如下文字：

This is a

test of the

DOS 5 editor.

记住，在每次输入一行后要按 Enter 键，如果发现你所输入的文字有误，可用光标移动键来修改错误的地方。当你确定无误后，屏幕就会显示如图 C—9 的菜单：

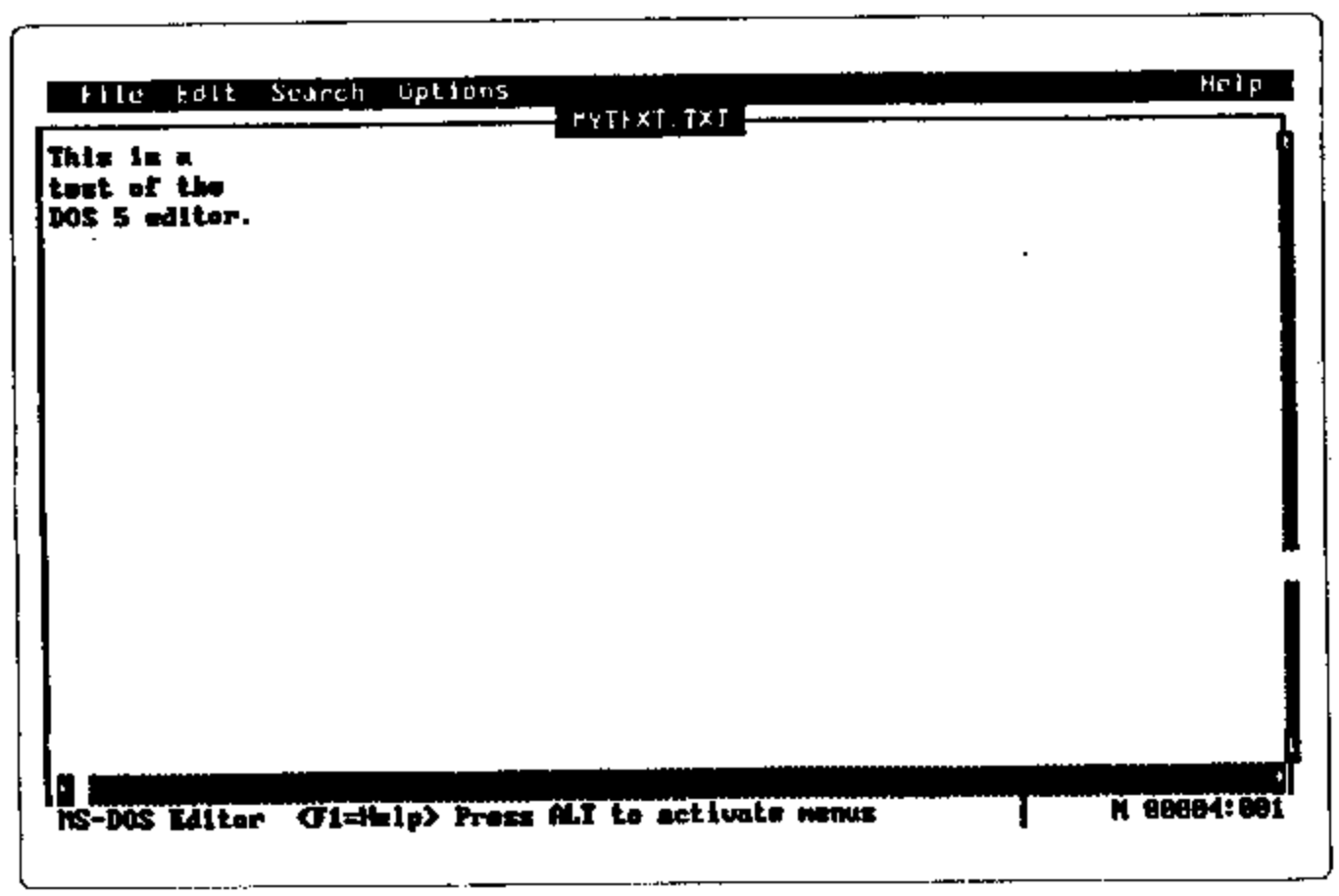


图 C—9

EDIT 编辑指令的键盘设定模式为插入状态，你可以利用 INS 键来选择修改的模式，或是恢复为插入的模式。现在请你将光标移到第二行的行首 t 字母上，然后输入 very small，最后按上 Enter 键，屏幕就会显示如图 C—10 的菜单：

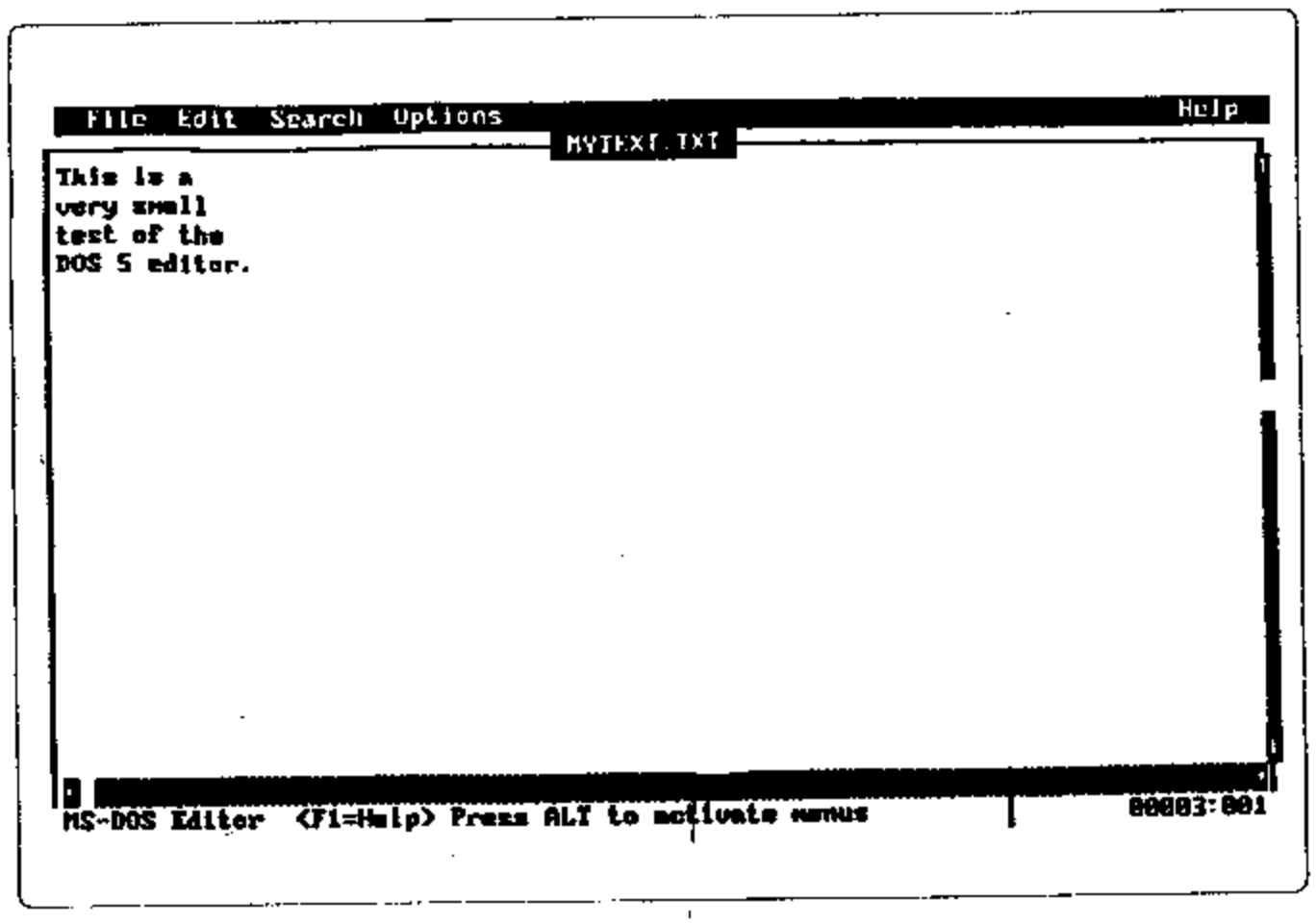


图 C—10

可看到 very small 这一行已被插入，而原来的一行则自动向下移动一行。要插入新行的方式有很多，你也可以先向下移动原来的一行，然后再从空出的一行上输入新的一行。无论你使用哪种方式，都是利用光标移动键来移动光标到你要修改的地方，然后再使用 EDIT 指令所提供的编辑键来进行编辑工作。

除了你所熟悉的上下左右移动键、退位键和 INS 键以外，EDIT 指令所提供的按键支持，也可以让你轻松愉快地完成编辑的工作。这些按键的功能说明列表如下：

按键	同义键	功能说明
←	Ctrl+S	左移一个字符
→	Ctrl+D	右移一个字符
↑	Ctrl+E	上移一行
↓	Ctrl+X	下移一行
Ctrl+←	Ctrl+A	左移一个短语
Ctrl+→	Ctrl+F	右移一个短语
Ctrl+↑	Ctrl+W	向上卷动
Ctrl+↓	Ctrl+Z	向下卷动
PgUp	Ctrl+R	上移一页
PgDn	Ctrl+C	下移一页
Ctrl+PgUp		向左卷动
Ctrl+PgDn		向右卷动
Home	Ctrl+Q+S	移到行首
End	Ctrl+Q+D	移到行尾
Ctrl+Home	Ctrl+Q+R	移到文件开头
Ctrl+End	Ctrl+Q+C	移到文件结尾
Ctrl+Q+E		移到窗口顶部
Ctrl+Q+X		移到窗口底部
Ctrl+Enter		移到下一行的行首

C.4 EDIT 编辑目录

当编辑的文件或程序越来越大时，除了可以利用上列的按键外，你也可以利用 EDIT 目录内各项功能来进行删除、拷贝或移动整段文件的工作。现在，请利用先前你所输入的文字来实际练习一下这些项目的用法，首先将光标移到第三行的行首 t 字母上，然后用左手按住 Shift 键不放，用右手按↓键两次来选用最后两行的文字。如果你的主机安装了鼠标，只要移动鼠标指针到你所要的位置上，然后按住鼠标左边的按钮，并向下移动鼠标到选用你所要的文字行，最后松开按钮即可。此时编辑窗口如图 C--11 所示：

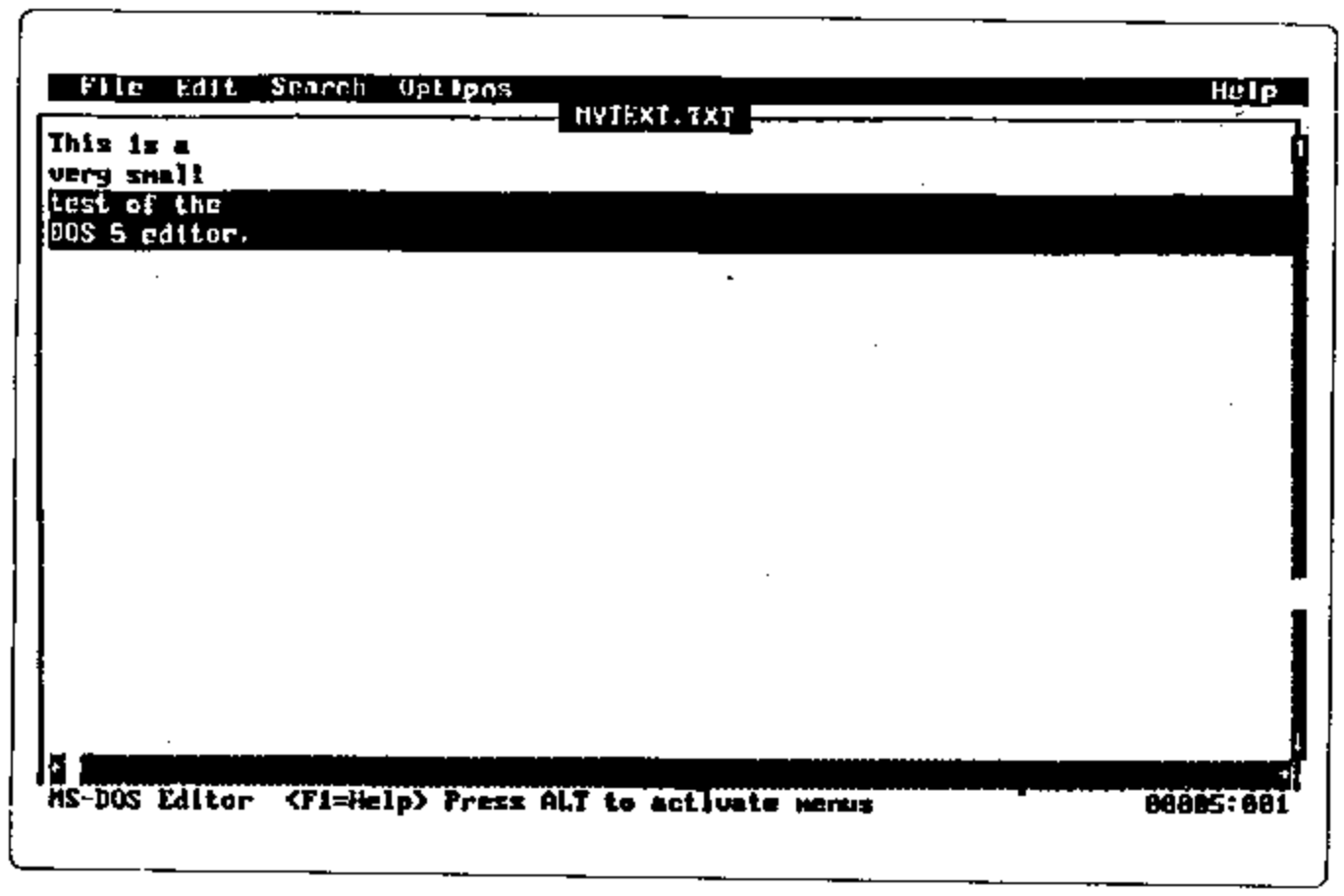


图 C　11

当前窗口中以反相效果显示的这两行文字，就是你刚才利用 Shift＋↓键所选用的。现在，请你按 DEL 键就可将这两行删除掉，然后再输入这两行文字以便后面的练习。除了 Shift＋↓键外，你也可以配合下列的按键来选用你所要的文字行：

按键	选用的文字行
Shift＋←	向左逐隔选用
Shift＋→	向右逐隔选用
Shift＋Ctrl＋←	向左逐字选用
Shift＋Ctrl＋→	向右逐字选用
Shift＋↓	向下逐行选用
Shift＋↑	向上逐行选用
Shift＋PgUp	向上逐屏选用
Shift＋PgDn	向下逐屏选用
Shift＋Ctrl＋Home	向上全部选用
Shift＋Ctrl＋End	向下全部选用

现在，请再利用 Shift＋↓键来选用前两行文字，然后选用 Edit 目录中的 Cut 项目，当你按下 Enter 键后，这两行文字就被删除掉。这个功能和你使用 DEL 键来删除选用的文字块是一样的，唯一的差别是 Cut 项目会将删除的文字块存储起来，因此仍然可以利用 Paste 项目重新将这些已被删除的文字块插入到原处或其他地方。现在，请你选用 Paste 项目就可发现原先被

删除的两行文字又出现在原来的地方。然后利用 Ctrl+End 键将光标移动到文件结尾，再选用 Paste 项目将你原先所选用的两行文字插到最后，此时你的编辑窗口如图 C—12 所示：

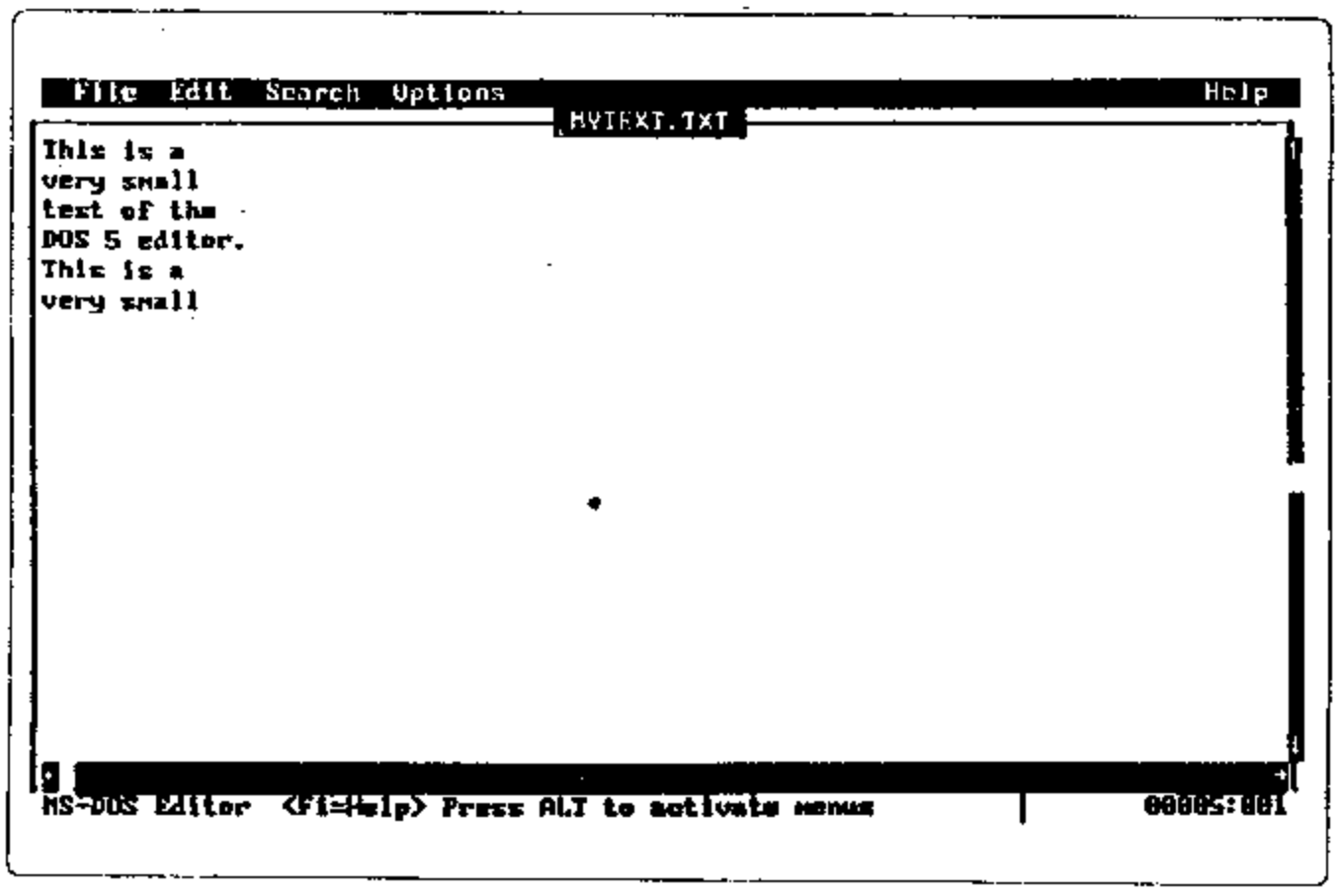


图 C—12

Edit 目录中的 Copy 项目的功能是将你所选用的文字块拷贝一份，但不会像 Cut 项目会将选用的文字块删除掉。因此，你也可以再利用 Paste 项目将它们插到程序或文件中的任何地方，以达到复制文字块的目的。Cut、Copy 和 Paste 这三个项目不但可以在同一个文件中进行编辑的工作，而且也可以在不同的文件间进行拷贝的工作，使文件或程序的编辑工作更加容易。

C. 5 Search 寻找目录

Search 寻找目录的功能使你在编辑文件或程序时，可以很方便地寻找你所要的字符串并加以修改。Search 寻找目录中含有三个选项，其中 Find 和 Change 这两个项目之后还紧接着三个小点，这表示当你选用这两个项目时，EDIT 编辑指令还会再显示一个提示框，要求你输入一组相关的数据。当你选用 Find 这个项目后，就会显示如图 C—13 所示的提示方框：

在 Find What 的小方框中，就是让你输入所要寻找字符串的地方，目前所要寻找的是 This 这个字。在 Find What 小方框下面还有两个选择项目，你可以利用 TAB 键来移动光标到各个项目上，然后按空格键来选用它。被选用的项目，它左边的中括号内会出现一个 X 符号，如果你再按一次空格键，原来被选用项目上的 X 符号就会消失，因此空格键具有开关的功能。左边这个项目的功能是用来指定寻找时，大小写要完全符合才会找到。而右边这个项目的功能则是用来指定寻找时，只要独立而且符合的字符串才会找到，例如寻找 PUT 时，INPUT 或 PUTS

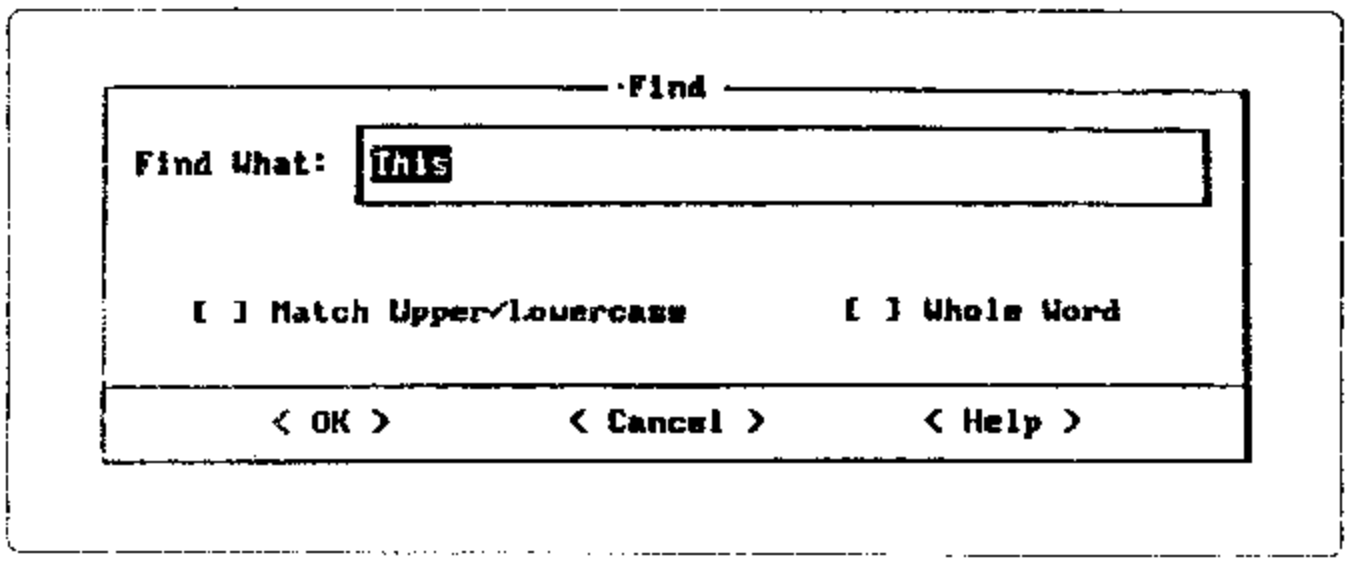


图 C—13

这些类似的字符串都不符合，只有 PUT 才会被找到。

当你确定提示方框中的选择项无误后，按下 Enter 键就可开始寻找。如果找到你所指定的字符串，光标就会移到该字符串的开头，并以反相的效果来显示它，如果无法找到，则会显示"Match not found"的错误信息来提醒你。如果你想要继续寻找相同的字符串，则只要选用 Search 目录中的 Repeat Last Find 项目或直接按 F3 键即可。当找到你所要的字符串后，就可进行修改或删除的编辑工作。如果你要做自动修改的工作，则可以选用 Search 目录中的 Change 项目，此时会显示如图 C—14 所示的提示方框：

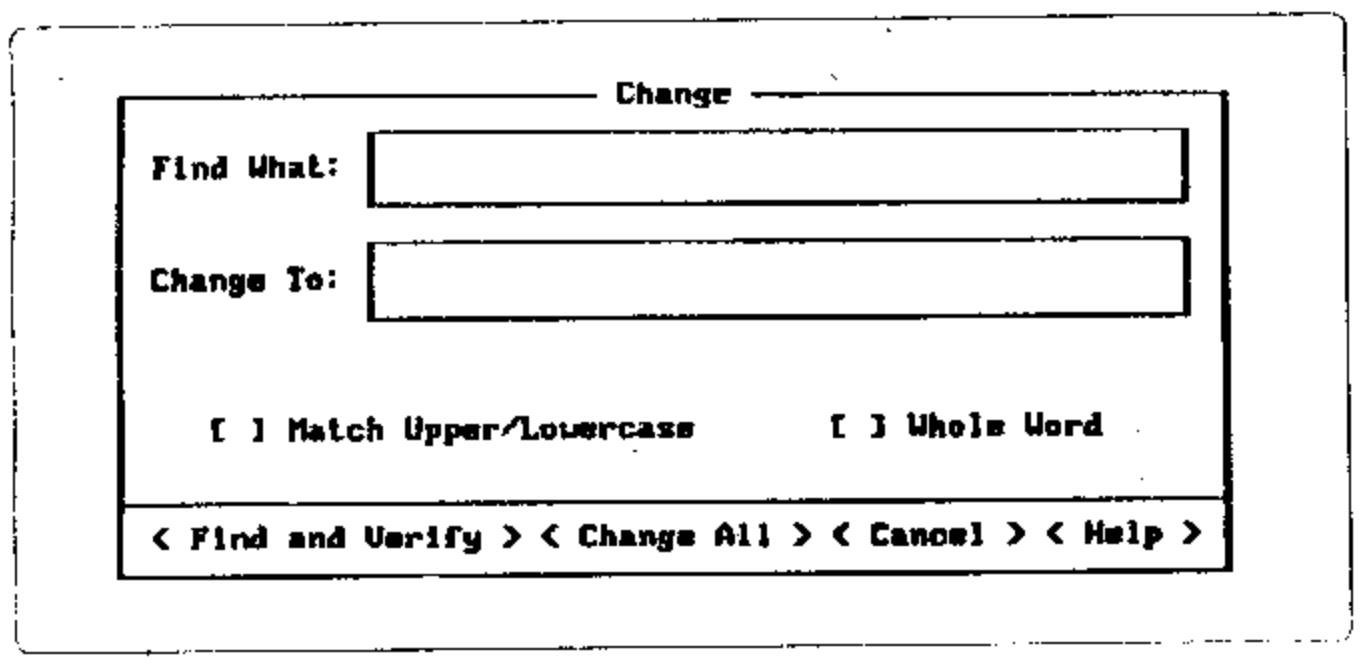


图 C—14

你一样可以在 Find What 小方框中输入你所要寻找的字符串，在 Change to 小方框中输入你所要改变的字符串，然后在方框底端选用你所要的处理方式，其中 Find and Verify 项目在修改之前会再显示一个提示方框，来询问你是否修改；而 Change All 这个项目则直接修改不再提醒你，因此选用这个项目时要特别小心。

C.6 存储和装入文件

当你完成编辑的工作后，可以选用 File 目录中的 Save 项目，来存储你所编辑的文件或程序到磁盘上。如果你在进入 EDIT 编辑指令时未指定你所要编辑的文件名，就必须选用 Save As 这个项目来存储你当前所编辑的文件或程序，当你选用 Save As 这个项目后会再显示如图 C—15 所示的提示方框：

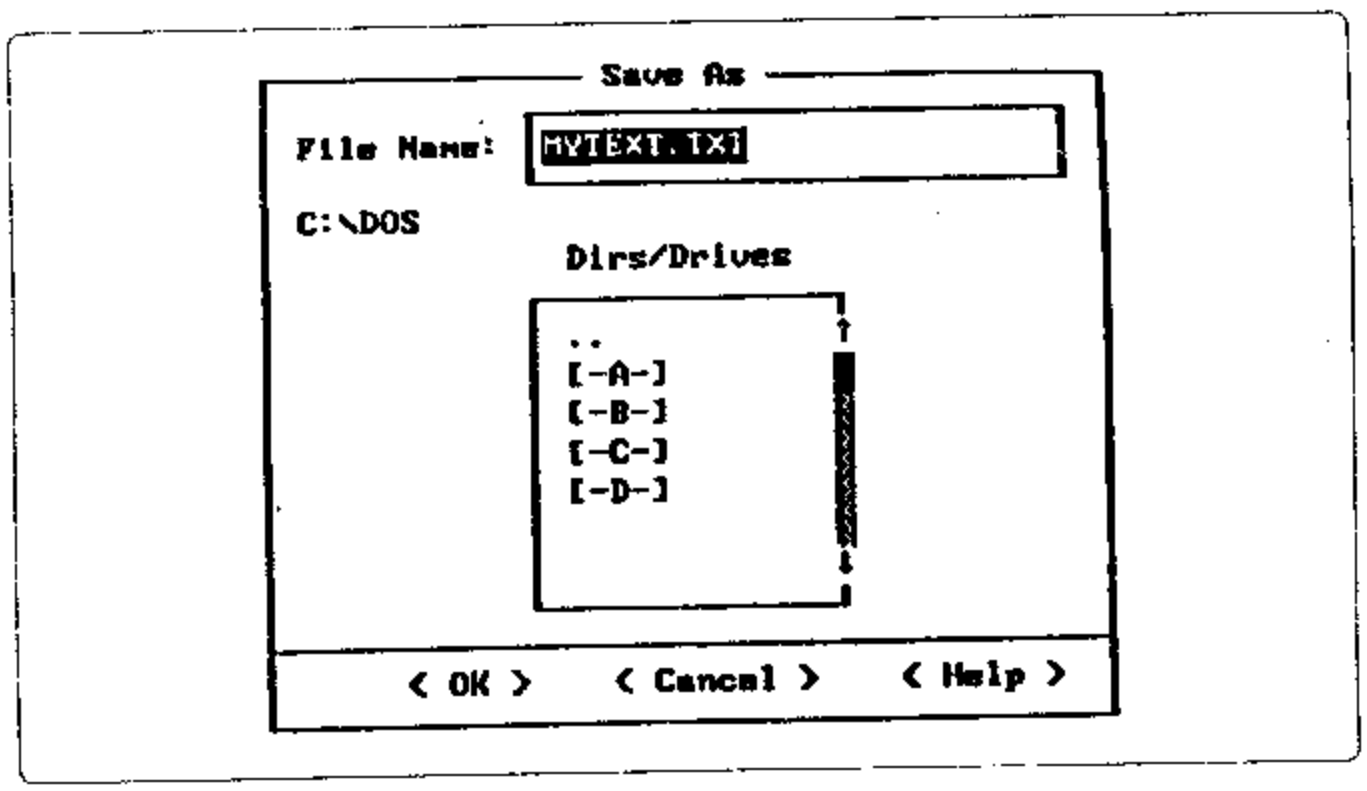


图 C—15

目前 File Name 小方框中所显示的是你先前所指定的文件名，如果你要沿用这个文件名，按下 Enter 键即可，那么原来存储在文件中的内容就被取代。如果你要保留原文件的内容，则可输入一个新的文件名来存储当前你正在编辑的文件或程序。

另外，File 目录中的 Open 项目可以让你打开一个旧的文件，以便进行编辑的工作，而 New 项目则是让你重新进行文件或程序的编辑，直到完成后再选用 Save As 项目来存储它。如果你想打印编辑窗口中的内容，也可以选用 Print 项目将数据输出到打印机上，当你选用 Print 这个项目时，屏幕会显示如图 C—16 所示的提示方框：

方框中含有两个选择项目供你选用，当前是选定第二项来打印全部的文件内容，你也可以选用第一项来打印你所选定的文字行。

最后，你可以选用 File 目录中的 Exit 项目来结束编辑的工作，在结束之前，如果你当前所编辑的内容未存储到文件中，则还会显示一个提示方框来提醒你，以免发生意外。

C.7 Options 和 Help 目录

EDIT 编辑程序的目录栏中，也包括了 Options 和 Help 两个目录供你选用，Help 目录提供你一个非常简明的联机帮助系统，你可通过它来取得有关 EDIT 编辑程序的使用说明。Options

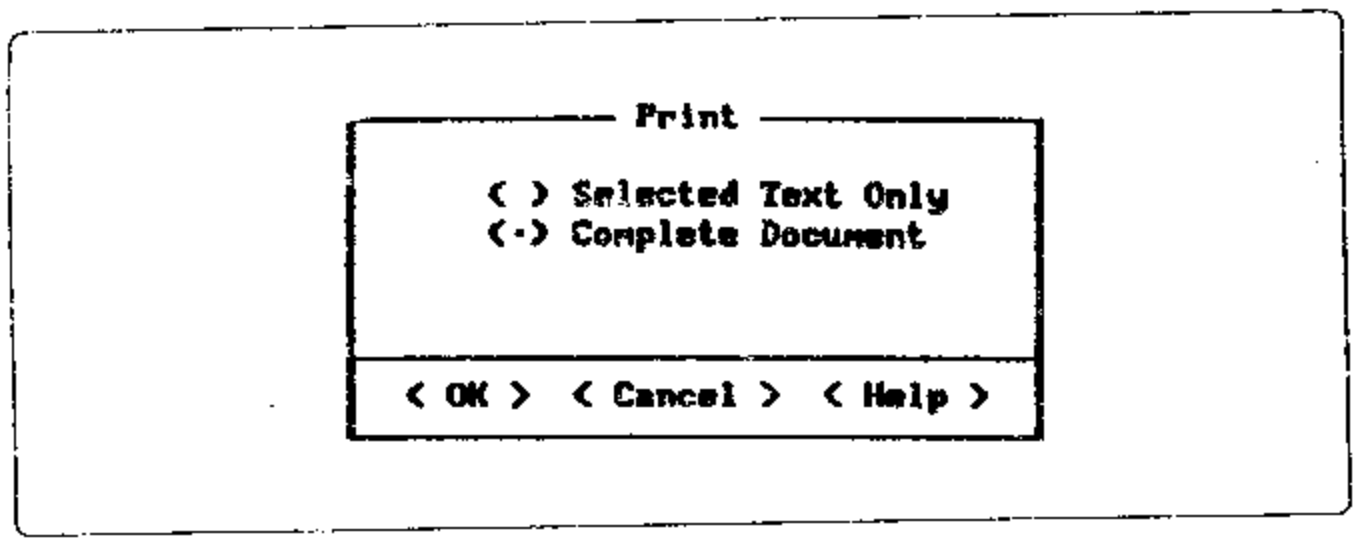


图 C　16

这个目录则是供你设定屏幕的显示特性和指定存储联机帮助系统的 EDIT.HLP 文件路径。选用 Options 目录中的 Display 项目后会显示如图 C--17 所示的菜单：

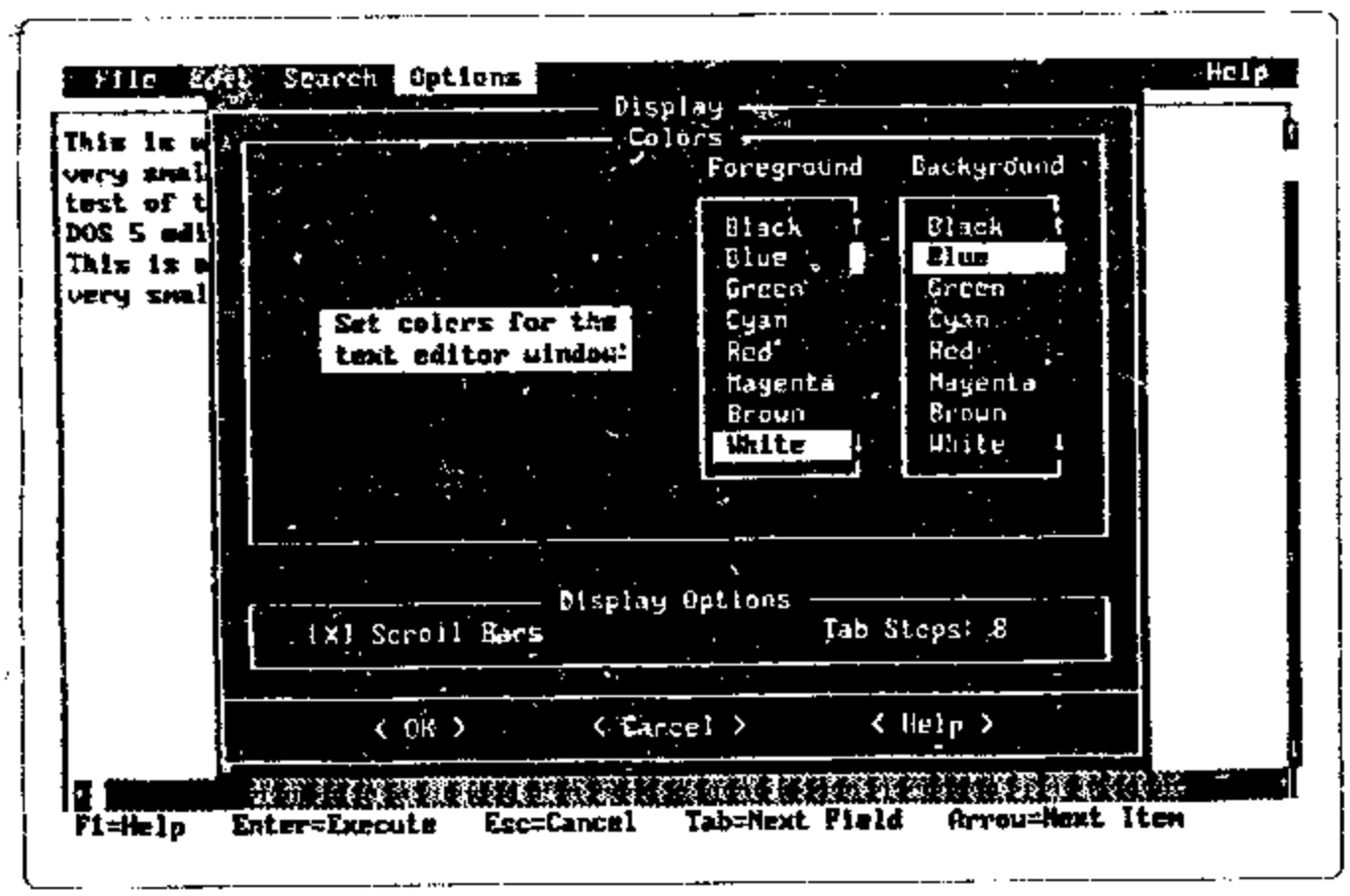


图 C　17

你可以利用这个窗口来设定屏幕前景和背景的颜色，还有选择是否使用滚动条及 TAB 键的定位。

附录 D 美国国家标准数据码(ASCII CODE)

D.1 控制字符

	十进制	十六进制	二进制
^@ NUL	0	00h	00000000
^A SOH	1	01h	00000001
^B STX	2	02h	00000010
^C ETX	3	03h	00000011
^D EOT	4	04h	00000100
^E ENQ	5	05h	00000101
^F ACK	6	06h	00000110
^G BEL	7	07h	00000111
^H BS	8	08h	00001000
^I HT	9	09h	00001001
^J LF	10	0Ah	00001010
^K VT	11	0Bh	00001011
^L FF	12	0Ch	00001100
^M CR	13	0Dh	00001101
^N SO	14	0Eh	00001110
^O SI	15	0Fh	00001111
^P DLE	16	10h	00010000
^Q DC1	17	11h	00010001
^R DC2	18	12h	00010010
^S DC3	19	13h	00010011
^T CD4	20	14h	00010100
^U NAK	21	15h	00010101
^V SYN	22	16h	00010110
^W ETB	23	17h	00010111
^X CAN	24	18h	00011000
^Y EM	25	19h	00011001
^Z SUB	26	1Ah	00011010
^[ESC	27	1Bh	00011011
^\ FS	28	1Ch	00011100
^] GS	29	1Dh	00011101
^^ RS	30	1Eh	00011110
^_ US	31	1Fh	00011111

D.2　标准字符

	32	20h	00100000
!	33	21h	00100001
"	34	22h	00100010
#	35	23h	00100011
$	36	24h	00100100
%	37	25h	00100101
&	38	26h	00100110
'	39	27h	00100111
(	40	28h	00101000
)	41	29h	00101001
*	42	2Ah	00101010
+	43	2Bh	00101011
,	44	2Ch	00101100
-	45	2Dh	00101101
.	46	2Eh	00101110
/	47	2Fh	00101111
0	48	30h	00110000
1	49	31h	00110001
2	50	32h	00110010
3	51	33h	00110011
4	52	34h	00110100
5	53	35h	00110101
6	54	36h	00110110
7	55	37h	00110111
8	56	38h	00111000
9	57	39h	00111001
:	58	3Ah	00111010
;	59	3Bh	00111011
<	60	3Ch	00111100
=	61	3Dh	00111101
>	62	3Eh	00111110
?	63	3Fh	00111111
@	64	40h	01000000
A	65	41h	01000001
B	66	42h	01000010
C	67	43h	01000011
D	68	44h	01000100
E	69	45h	01000101
F	70	46h	01000110
G	71	47h	01000111
H	72	48h	01001000
I	73	49h	01001001
J	74	4Ah	01001010
K	75	4Bh	01001011
L	76	4Ch	01001100
M	77	4Dh	01001101
N	78	4Eh	01001110
O	79	4Fh	01001111
P	80	50h	01010000
Q	81	51h	01010001
R	82	52h	01010010
S	83	53h	01010011
T	84	54h	01010100
U	85	55h	01010101
V	86	56h	01010110
W	87	57h	01010111
X	88	58h	01011000
Y	89	59h	01011001
Z	90	5Ah	01011010
[	91	5Bh	01011011
\	92	5Ch	01011100
]	93	5Dh	01011101
^	94	5Eh	01011110
_	95	5Fh	01011111
`	96	60h	01100000
a	97	61h	01100001
b	98	62h	01100010
c	99	63h	01100011
d	100	64h	01100100
e	101	65h	01100101
f	102	66h	01100110
g	103	67h	01100111
h	104	68h	01101000
i	105	69h	01101001
j	106	6Ah	01101010
k	107	6Bh	01101011
l	108	6Ch	01101100
m	109	6Dh	01101101
n	110	6Eh	01101110
o	111	6Fh	01101111
p	112	70h	01110000
q	113	71h	01110001
r	114	72h	01110010
s	115	73h	01110011
t	116	74h	01110100
u	117	75h	01110101
v	118	76h	01110110
w	119	77h	01110111
x	120	78h	01111000
y	121	79h	01111001
z	122	7Ah	01111010
{	123	7Bh	01111011
|	124	7Ch	01111100
}	125	7Dh	01111101
~	126	7Eh	01111110
Δ	127	7Fh	01111111

D.3 扩充字符

Ç	128	80h	10000000
ü	129	81h	10000001
é	130	82h	10000010
â	131	83h	10000011
ä	132	84h	10000100
à	133	85h	10000101
å	134	86h	10000110
ç	135	87h	10000111
ê	136	88h	10001000
ë	137	89h	10001001
è	138	8Ah	10001010
ï	139	8Bh	10001011
î	140	8Ch	10001100
ì	141	8Dh	10001101
Ä	142	8Eh	10001110
Å	143	8Fh	10001111
É	144	90h	10010000
æ	145	91h	10010001
Æ	146	92h	10010010
ô	147	93h	10010011
ö	148	94h	10010100
ò	149	95h	10010101
û	150	96h	10010110
ù	151	97h	10010111
ÿ	152	98h	10011000
Ö	153	99h	10011001
Ü	154	9Ah	10011010
¢	155	9Bh	10011011
£	156	9Ch	10011100
¥	157	9Dh	10011101
₧	158	9Eh	10011110
ƒ	159	9Fh	10011111
á	160	A0h	10100000
í	161	A1h	10100001
ó	162	A2h	10100010
ú	163	A3h	10100011
ñ	164	A4h	10100100
Ñ	165	A5h	10100101
ª	166	A6h	10100110
º	167	A7h	10100111
¿	168	A8h	10101000
⌐	169	A9h	10101001
¬	170	AAh	10101010
½	171	ABh	10101011
¼	172	ACh	10101100
¡	173	ADh	10101101
«	174	AEh	10101110
»	175	AFh	10101111
▒	176	B0h	10110000
▓	177	B1h	10110001
▒	178	B2h	10110010
│	179	B3h	10110011
┤	180	B4h	10110100
╡	181	B5h	10110101
╢	182	B6h	10110110
╖	183	B7h	10110111
╕	184	B8h	10111000
╣	185	B9h	10111001
║	186	BAh	10111010
╗	187	BBh	10111011
╝	188	BCh	10111100
╜	189	BDh	10111101
╛	190	BEh	10111110
┐	191	BFh	10111111
└	192	C0h	11000000
┴	193	C1h	11000001
┬	194	C2h	11000010
├	195	C3h	11000011
─	196	C4h	11000100
┼	197	C5h	11000101
╞	198	C6h	11000110
╟	199	C7h	11000111
╚	200	C8h	11001000
╔	201	C9h	11001001
╩	202	CAh	11001010
╦	203	CBh	11001011
╠	204	CCh	11001100
═	205	CDh	11001101
╬	206	CEh	11001110
╧	207	CFh	11001111
╨	208	D0h	11010000
╤	209	D1h	11010001
╥	210	D2h	11010010
╙	211	D3h	11010011
╘	212	D4h	11010100
╒	213	D5h	11010101
╓	214	D6h	11010110
╫	215	D7h	11010111
╪	216	D8h	11011000
┘	217	D9h	11011001
┌	218	DAh	11011010
█	219	DBh	11011011
▄	220	DCh	11011100
▌	221	DDh	11011101
▐	222	DEh	11011110
▀	223	DFh	11011111
α	224	E0h	11100000
ß	225	E1h	11100001
Γ	226	E2h	11100010
π	227	E3h	11100011

Σ	228	E4h 11100100	≥	242	F2h 11110010
σ	229	E5h 11100101	≤	243	F3h 11110011
μ	230	E6h 11100110	⌠	244	F4h 11110100
τ	231	E7h 11100111	⌡	245	F5h 11110101
Φ	232	E8h 11101000	÷	246	F6h 11110110
Θ	233	E9h 11101001	≈	247	F7h 11110111
Ω	234	EAh 11101010	°	248	F8h 11111000
δ	235	EBh 11101011	∙	249	F9h 11111001
∞	236	ECh 11101100	·	250	FAh 11111010
φ	237	EDh 11101101	√	251	FBh 11111011
∈	238	EEh 11101110	n	252	FCh 11111100
∩	239	EFh 11101111	²	253	FDh 11111101
≡	240	F0h 11110000	■	254	FEh 11111110
±	241	F1h 11110001		255	FFh 11111111

[General Information]
书名=DOS批处理文件设计技巧
作者=BEXP
页数=205
下载位置=http://book7.ssreader.com/diskjsj/l/0l02/!00001.pdg

目录
正文