MS-DOS® 6.2 批处理文件高级指南

[美] Kris Jamsa 著
唐晓非 张 勇 译
张 勇 审校

清华大学出版社

(京)新登字 158 号

MS-DOS® 6.2 批处理文件高级指南
Concise Guide to MS-DOS Batch Files

Kris Jamsa

图书在版编目(CIP)数据

MS-DOS® 6.2 批处理文件高级指南/(美) Kris Jamsa 著；唐晓非，张勇译. -北京：清华大学出版社，1994
ISBN 7-302-01605-4

Ⅰ. M… Ⅱ. ①K… ②唐… ③张… Ⅲ. 批量处理-磁盘操作系统-手册 Ⅳ. TP316.3

中国版本图书馆 CIP 数据核字(94)第 08868 号

出版者：清华大学出版社(北京清华大学校内，邮编 100084)
印刷者：化学工业出版社印刷厂
发行者：新华书店总店北京科技发行所
开　本：787×1092　1/16　印张：11.25　字数：267 千字
版　次：1994 年 9 月第 1 版　1994 年 9 月第 1 次印刷
书　号：ISBN 7-302-01605-4/TP・677
印　数：0001--5000
定　价：18.50 元

引 言

多年来，有经验的 MS-DOS 用户都使用 MS-DOS 批处理文件来节省时间和减少击键；不幸的是，从批处理文件中常常能受益最多的新用户却从未真正被教会如何使用这些批处理文件，因此，本书将介绍 MS-DOS 批处理文件，且从基础开始。随着你的理解力和信心的不断增强，本书将逐渐介绍一些高级的技术，使你能充分发挥计算机的潜能。有经验的批处理文件用户，将学习如何用他们的批处理文件给屏幕显示加上色彩和划线字符。本书还将介绍 MS-DOS 6 中引入的新的批处理文件命令，此外，你还将学到如何通过菜单来选择 CONFIG.SYS 中的项以控制你的系统配置，以及如何同 Microsoft Windows 一起使用批处理文件。

简单地说，一个 MS-DOS 批处理文件是一个内含 MS-DOS 命令的文件，当你在命令提示符后面敲入批处理文件名时，MS-DOS 就会执行批处理文件中的命令。许多用户喜欢每天利用一些简单的批处理文件，缩写一些常用命令。然而，众所周知，多数用户不能充分利用批处理文件提供的潜在功能。

尽管 MS-DOS 命令功能非常强大，可是有些时候命令还是不能提供你所需要的所有功能。因此，本书将就如何创建自己的批处理文件，提供一切所需的资料。从非常基础的知识开始，介绍用 MS-DOS 编辑工具创建自己的批处理文件，然后引导你熟悉每条批处理文件命令，每一条命令均有一个详细的说明，及用于描述该命令的一个实例。你还会学到一些高级批处理文件命令，如用 IF ERRORLEVEL 语句测试出口条件，用 MS-DOS 6 中的命令 CHOICE 获取用户的应答，及使用有名参数等。然后，随着你开始建立自己的批处理文件库，你将会学到如何将一些命令组合在一起来完成一些有用的任务。其中最大的优点是，这些程序只需几分钟就可以编写出来，你不需要任何特殊的编程语言——所需的只是 MS-DOS。

如果你是 MS-DOS 5.0(或更高版本)的用户，本指南将教你如何用 DOSKEY 命令建立宏定义。一个宏类似于一个 MS-DOS 批处理文件，它包含一个或多个命令名，你简单地敲入宏名就可以执行它们。与 MS-DOS 批处理文件不同的是，宏驻留在计算机的高速电子 RAM(随机存储器)中，而不是磁盘上。如果你是 MS-DOS 5.0 的用户，DOSKEY 命令的知识会使你的工作效率更高。

计算机系统上的一个最重要的批处理文件是 AUTOEXEC.BAT，该文件用于配置你的系统，本书将详细讨论它。从 MS-DOS 6 开始，MS-DOS 提供了一系列可在 CONFIG.SYS 文件中设置的项，这些项能更好地控制 MS-DOS 如何将自身加载到内存中，本书将向你展示如何使用每一项。参照提供的例子，你可以快速地建立起预置菜单，在每次计算机启动时 MS-DOS 会显示这一菜单并且让你选择所需配置。正如你将看到的，像 MS-DOS 批处理文件一样，一旦你了解了其中的秘密，就会很容易地使用 CONFIG.SYS

目 录

第 1 章

批处理文件入门

1.1 理解批处理文件

MS-DOS 命令现在已经超过 80 个了，然而，我们中间的多数人还只是使用其中的一个很小的子集，如 COPY、REN、DEL、TYPE、DISKCOPY 和 FORMAT——及可能的子目录命令。我们不可能复习许多其他命令的格式，更不用说它们的命令行参数了。

如果要完成诸如移到一个特殊目录中或运行一个字处理软件等功能，新用户通常要执行一系列命令。随着你必须的击键量的增加，错误的概率也不断增大。错误可能是丢掉了一个重要命令，或者是印刷错误。

为了简化 MS-DOS 的使用并减小错误的概率，MS-DOS 提供了批处理文件。一个批处理文件是由你创建的、内有一个或几个 MS-DOS 命令的文件。要执行批处理文件中的所有命令，你只要简单地在命令提示符下敲入批处理文件名并按一下 Enter 键即可，就好像批处理文件是一个 MS-DOS 单命令一样。接着，MS-DOS 执行所有的批处理文件命令，先执行第一条命令，直到最后一条命令。为了了解批处理文件如何工作，让我们先研究一下批处理文件 TIMEDATE.BAT：

其中有三条 MS-DOS 的基本命令：

```
CLS
TIME
DATE
```

以后，你可以创建一个类似的批处理文件；现在，我们将简单地探讨它是如何执行的。

CLS 清屏，TIME 显示当前时间并提示你输入一个新时间；DATE 显示当前日期并提示你输入一个新日期。

要注意，文件的扩展名是 BAT。文件的扩展名可以描述一个文件中包含信息的类型。在 TIMEDATE.BAT 这个例子中，扩展名 BAT 不仅告诉 MS-DOS，而且告诉你，该文件是一个批处理文件。要运行这个批处理文件，只需像下面那样在命令提示符后面敲入批处理文件名，然后按下 Enter 键：

```
C:\>TIMEDATE
```

当 MS-DOS 遇到批处理文件名时，就打开该批处理文件并执行文件中的第一条命令——在本例中是 CLS。CLS 命令执行完后，MS-DOS 执行批处理文件中的下一条命令——TIME。这时，MS-DOS 显示以下信息：

```
C:\>TIME
Current time is 3:26:46.03P
Enter new time:
```

现在你可以输入正确时间并敲一下 Enter 键，或者你可以敲一下 Enter 键，不修改时间就离开。TIME 执行完后，MS-DOS 接着执行 DATE 命令并显示如下信息：

```
C:\>DATE
Current date is Thu 04-07-1994
Enter new date (mm dd-yy):
```

现在你可以输入正确的日期并敲一下 Enter 键，或者不作任何修改敲一下 ENTER 键离开。当 DATE 命令执行完时，MS-DOS 搜寻批处理文件中的下条命令，因为不存在其他命令了，所以批处理文件就执行完了，并且 MS-DOS 显示其提示符。

注意：低于 4.0 版的 MS-DOS 不支持 12 小时时钟，这些版本不支持用 P 代表 P.M.，像上面显示的那样。相反，这些版本将显示 15:26:46.03，如果没有其他注释，本书均基于 MS-DOS 6。

我们来研究一下另外一个批处理文件 DISKINFO.BAT，其中有三条 MS-DOS 命令：

```
VOL
CHKDSK
DIR
```

该批处理文件将显示有关当前驱动器的信息，要执行该文件，敲入批处理文件名并敲一下 Enter 键：

```
C:\>DISKINFO
```

注意：在执行该批处理文件时，如果 MS-DOS 显示 Bad command or file name，那么就是 MS-DOS 找不到外部命令 CHKDSK.EXE。外部命令必须在当前目录中或可通过在 AUTOEXEC.BAT 文件中的 MS-DOS PATH 命令定义的路径找到。

1.2 批处理文件的优点

MS-DOS 批处理文件能节省用户的时间、减少击键次数和错误，并且可以简化复杂命令的执行。随着你逐渐适应了批处理文件，批处理文件会变成使计算机效率更高的重要工具。若要了解如何才能做到，我们来看几个批处理文件例子。

1.2.1 节省时间

假定你每天早晨要运行四个货存控制程序。第一个程序，CALCINV.EXE，计算你现

令时的紧张感。批处理文件减少了 MS-DOS 新用户必须记住的复杂命令并成功地执行它们，这样就帮助用户减轻了这种紧张感。例如，我们多数都将字处理文档文件放在一个唯一的子目录中，为了运行字处理器并方便地存取那些文件，你必须先用 CHDIR（改变目录）选择正确的目录，然后选择字处理器。例如，在 MICROSOFT WORD 中，命令的顺序就变成如下形式：

```
C：\>CHDIR \DOCS

C：\>WORD
```

不论你是新用户还是能帮助新手的有经验的用户，你都应考虑创建一个名为 DOCS.BAT 的批处理文件，里面有两个命令。新用户必须记住的命令越少，他们就越觉得计算机使用起来舒服，结果，他们将学得更快。

1.3 命名批处理文件

你应该总是想为你创建的每一个文件取一个有意义的文件名，批处理文件也不例外。MS-DOS 批处理文件必须有 BAT 扩展名，这样，你就只有八个字符长的文件名来区分每个批处理文件。批处理文件名应能清晰地说明批处理文件执行的过程，例如，前面我们研究过批处理文件 TIMEDATE.BAT，该文件设置你的系统时间和日期；在我们的货存的例子中，批处理文件为 GETINV.BAT，这两个批处理文件名说明了文件的执行过程。尽管将批处理文件命名为 X.BAT 或 Z.BAT 很容易，但它们均不能描述批处理文件是干什么的。在你创建这样一个批处理文件几天之后，你可能就记不起它的功能了。

要避免使批处理文件同 MS-DOS 的内部或外部命令（以 EXE 或 COM 扩展名标识的）重名。尽管批处理文件也被 MS-DOS 看作命令，若 MS-DOS 先遇到一个内部或外部命令，就执行那个命令而不是批处理文件。MS-DOS 永远不会执行一个与 MS-DOS 内部命令同名的批处理文件，并且只在非常特殊的情况下执行与外部命令同名的批处理文件。

每当你在命令提示符下输入一个命令时，MS-DOS 首先检查它是不是一个内部命令（如 CLS、DATE 或 TIME），内部命令一直在内存中。若你的命令是一个内部命令，MS-DOS 就执行它（而不是什么外部命令，包括你的批处理文件）。例如，你的批处理文件名是 TIME.BAT，MS-DOS 将查找 TIME 命令并且只执行这个内部命令。

若 MS-DOS 发现那个命令不是内部命令，接着就检查该命令是否为当前目录中的外部命令。一个外部 MS-DOS 命令（如 DISKCOPY 或 FORMAT）是存在盘上的 MS-DOS 命令，其他外部命令包括那些用于启动程序的命令（如 WORD.EXE 或 WIN.COM）。如果 MS-DOS 找到了你输入的外部命令，就执行那个命令。

最后，如果你的命令不是内存中的内部命令或盘上的外部命令，MS-DOS 就查看你的命令是否对应于当前目录中的 MS-DOS 批处理文件。MS-DOS 只有在当前驱动器和当前目录中没有找到匹配的内部或外部命令时才会执行批处理文件。

假定在当前目录中没有找到匹配文件名的命令，MS-DOS 接着就开始在你的搜索路

径所指定的目录中查看。它检查每一目录(以指定的顺序),先查找外部命令,然后找批处理文件。只有在MS-DOS找到一个同名的外部命令之前先找到一个与指定文件名匹配的批处理文件,才会执行批处理文件(在第2章,我们将讨论如何在你的AUTOEXEC.BAT文件中设置你的搜索路径)。

总之,如果你为批处理文件赋了一个内部MS-DOS命令名,批处理文件就永远不会被执行。如果你为批处理文件赋了一个外部命令名,MS-DOS可能执行批处理文件,也可能执行外部命令,这取决于MS-DOS先找到哪个文件。

1.4 创建批处理文件

你所选择的创建批处理文件的方法因其文件长度而变。对于短的批处理文件最简单的和最快的方法是从键盘上拷贝批处理文件。要做到这一点,你可以用CON(键盘的设备名)作为批处理文件的输入的源,执行一个拷贝操作。作为示例,我们将创建TIMEDATE.BAT批处理文件,它用来设置系统的时间和日期。若要开始批处理文件拷贝操作,可在命令提示符后输入如下命令:

```
C:\>COPY CON TIMEDATE.BAT
```

MS-DOS接着就用COPY命令执行一个拷贝操作,在这种情况下,要拷贝的数据的源是键盘,拷贝操作的目的是批处理文件TIMEDATE.BAT。当你按下Enter键开始拷贝时,MS-DOS将光标置于命令下一行的行首。在这时,MS-DOS等待输入第一行。输入命令TIME,即批处理文件中的第一个命令,并按下Enter键:

```
C:\>COPY CON TIMEDATE.BAT
TIME
```

再输入命令DATE,并按下Enter键:

```
C:\COPY CON TIMEDATE.BAT
TIME
DATE
```

DATE命令是批处理文件中的最后一条命令,你必须告诉MS-DOS文件没有输入了,为了做到这点,按下F6功能键(文件结束),然后按下Enter键。MS-DOS在批处理文件底部显示字符^Z(注意:不要按SHIFT-8产生尖号,可供选择的另一种方法是按下Ctrl键和Z键产生^Z)。

字符^Z,发音为Control Z,在MS-DOS中指示文件结束。当你在字符^Z后面按Enter键时,就告诉MS-DOS你已经做完拷贝操作,这样MS-DOS就创建了批处理文件,并且显示如下:

```
C:\>COPY CON TIMEDATE.BAT
TIME
DATE
```

```
^Z
    1 File(s) copied

C:\>
```

要执行该批处理文件，只需输入 TIMEDATE：

```
C:\>TIMEDATE
```

当你按 Enter 键时，MS-DOS 先执行 TIME 命令，然后执行 DATE 命令。

现在用 COPY CON 技术创建批处理文件 DISKINFO. BAT，其中有命令 VOL、CHKDSK 和 DIR。像前面一样，输入 COPY 命令行，用 CON 设备名作输入源并按 Enter 键：

```
C:\>COPY CON DISKINFO.BAT
```

接着输入每一处理文件命令，在每一命令名后面按 Enter 键：

```
C:\>COPY CON DISKINFO.BAT
VOL
CHKDSK
DIR
```

为了告诉 MS-DOS 你已输入完命令了，按 F6 键，然后按 Enter 键。MS-DOS 就创建了那个批处理文件，并且提示已经拷贝了一个文件：

```
C:\>COPY CON DISKINFO.BAT
VOL
CHKDSK
DIR
^Z
    1 File(s) copied
C:\>
```

总之，你可以按如下步骤由键盘上创建小的批处理文件：

1. 执行一个 COPY 命令，用 CON 作源，用一个有意义的批处理文件名作目的(target)，然后按 Enter 键，例如：

```
C:\>COPY CON DATEPRNT.BAT
```

2. 每次输入一个批处理文件命令，在每一命令后面按 Enter 键。

3. 在输入最后一个批处理文件命令后，按 F6 键，然后按 Enter 键(F6 为 MS-DOS 标志文件的结束)。

1.4.1 创建较大的批处理文件

随着你的批处理文件的不断增大，你将需要用一个字处理器，即全屏幕 MS-DOS 编辑器(MS-DOS 5.0 及以上版本提供)，或行编辑器 Edlin(MS-DOS 5.0 以下版本提供)。

如果你使用的是一个字处理器，一定要以正文(ASCII)格式保存文件。正如你可能知道的，字处理器会允许你以几种方式格式化正文，如改变字体或特定字符的大小或特定段落的排列方式，为了达到这些目的，字处理器就在你的文件中嵌入字符，尽管这些嵌入的字符对于字处理器是有意义的，但 MS-DOS 却无法理解它们。如果这些字符出现在你的批处理文件中，将会引起错误。如果你以 ASCII 格式保存你的字处理文件，那么字处理器就不会在文件中嵌入这些字符。

1.4.2 用 Edit 创建批处理文件

MS-DOS 5.0 及更高版本提供了 MS-DOS 编辑器，也称为 Edit。Edit 是一个全屏幕编辑器(意味着你可以使用整个屏幕)，可以让你创建和修改文件。为了理解 Edit 如何工作，我们来用它创建一个名为 SHORTDIR.BAT 的批处理文件，内有命令 CLS 和 DIR/W。

首先，在命令提示符下运行 Edit，并说明相应的文件名。如果该文件尚不存在，它就会被创建。如果没有说明文件名，就创建一个无名字文件，在保存该文件时，必须说明其名字。例如，命令行可为：

```
C:\>EDIT SHORTDIR.BAT
```

当按 Enter 键时，Edit 就显示如图 1-1 所示的屏幕。

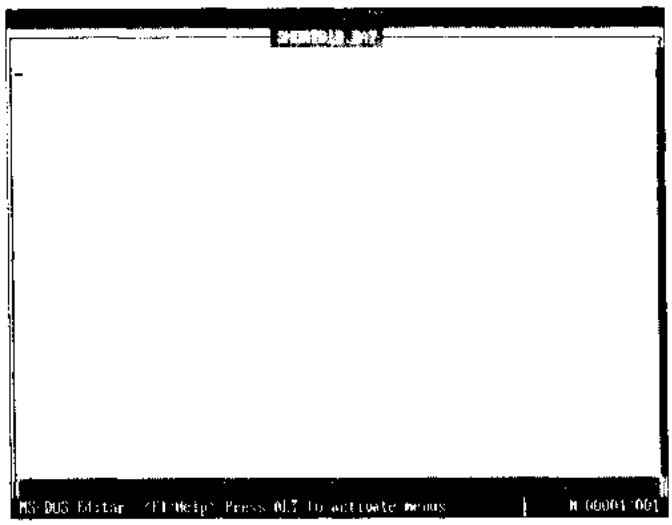


图 1-1 Edit 屏幕

在左上角显示的下划线是光标，它说明字符在被键入时会出现的位置。在编辑区域上面中间显示的是文件名 SHORTDIR.BAT(如果没有在命令行上说明文件名则会看见该位置上显示 Untitled)。在文件的左上方是四个用来向 Edit 发出命令的下拉菜单。在此例中，我们将只用两个菜单命令(为获取有关 Edit 任一性质的信息，可按 F1 键以进入 Edit 的联机帮助系统)。

为创建 SHORTDIR.BAT，敲入下列命令，在每行的结束要按 Enter 键：

```
CLS
DIR/W
```

如果敲错了字符，可用 Backspace 键来删除错误，然后再重新敲入此行的内容。如果光标不在出错的那行上，可用箭头键来将光标移至有错的那一行上，然后再用 Backspace 键来改正错误。

为在磁盘上保存 SHORTDIR.BAT，可先按 Alt 键，再按 F 键以打开 File 菜单。接着，按 S 以选择 File 菜单的 Save 命令（如果在一开始执行 Edit 时没有说明文件名，Edit 会显示一对话框以便输入文件名。只需敲入 SHORTDIR.BAT，再按 Enter 就可命名并保存该文件）。

注意：如果使用的是鼠标，那么可以方便地激活 Edit 菜单中的命令。将鼠标指示符的位置放至菜单名上，按下鼠标的左按钮就可打开此菜单。然后可再按按钮以激活所选择的命令。例如，为保存一文件，先激活 File 菜单，然后再激活 Save 命令。

在运行所建立的批处理文件前，必须先退出 Edit 程序而返回至命令提示符。为此，按 Alt，F（以打开文件菜单），然后再按 X（以选择 File 菜单的 Exit 命令）。

为运行新创建的批处理文件，敲入 SHORTDIR：

```
C:\>SHORTDIR
```

然后再按 Enter 键，MS-DOS 将会清除屏幕，然后在屏幕上用文件名显示目录（如同用/W 开关显示目录那样）。

Edit 还可用来修改一现存文件。例如，可修改 SHORTDIR.BAT 文件以让 MS-DOS 只显示带 EXE 扩展名的文件，如先前那样，敲入 Edit 并说明文件名：

```
C:\>EDIT SHORTDIR.BAT
```

由于该文件已经存在，Edit 就在其窗口中显示其与以前所敲入的完全一样的内容。

为修改此文件，只需将光标移至想要修改的行上，对该行进行修改，然后再次保存该文件。在本例中，是想将 *.EXE 文件扩展名加至第二行，所以开始时可以按一次下箭头键来将光标下移一行。然后按 END 键以将光标移至行尾。接着按两次左箭头键以将光标移至/下面（使用鼠标时只需对着/按下按钮即可）。最后，敲入 *.EXE，后面再跟上一空格。已在行上的/W 会在敲入上述内容时向右移，结果看上去类似于图 1-2（如果你的批处理文件与图 1-2 的不同，现在就应进行必要的修改）。

为保存修改后的文件，按 Alt，F，再按 S。最后，按 Alt，F 以及 X 可从 Edit 程序中退出并返回至命令提示符。现在运行该批处理文件时，MS-DOS 就会清除屏幕，然后只显示带 EXE 扩展名的文件。

如果不想再在 Edit 中编辑一文件，并且不保存对文件所进行的修改而返回至命令提示符，可选择 FILE 菜单的 Exit 命令。当不对一新文件或一旧文件的修改进行保存而使用

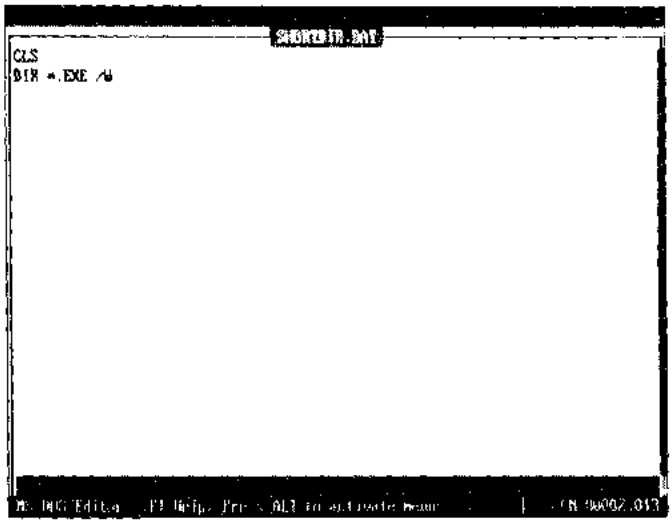


图 1-2　编辑后的 shortdir.bat

Exit 命令时，Edit 会显示如图 1-3 所示的对话框以避免在对文件进行保存之前意外地退出。

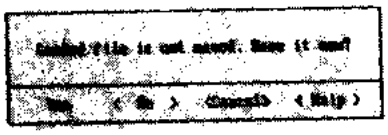


图 1-3　Exit 对话框

如果按了 Enter 键从而选择 Yes，Edit 就在返回至命令提示符之前保存对文件所做的任何修改。如果先按了 Tab 键，再按 Enter 键从而选择了 N，那么 Edit 就不保存所做的任何修改而返回至命令提示符。如果按 Esc 键而选择了 Cancel，Edit 就取消 Exit 命令而返回至文件中，可以继续对文件进行编辑。

注意：与许多其他正文编辑器不同，Edit 在编辑文件及保存改变了的内容时不对文件原先的内容进行备份。原先的文件内容完全被覆盖了。所以，一般来说，在修改重要的批处理文件的内容之前，一定要用 COPY 命令对文件进行备份。第四章将说明如何建立一个批处理文件来自动地完成此任务。

1.4.3　用 Edlin 创建批处理文件

如果使用的是 MS-DOS 4.0 或更早的版本，那么可以用 EDLIN 来建立批处理文件。Edlin 是一个行编辑器(这意味着每次只能对一行进行编辑)，它可用来创建及修改文件。为了解 Edlin 是如何工作的，让我们用它来创建一个含 CLS 及 DIR/W 命令的名为 SHORTDIR.BAT 的批处理文件。

先从命令提示符下运行 Edlin，并说明想要编辑的文件名。必须在 Edlin 命令行中说明文件名。在本例中，命令行如下：

```
C:\>EDLIN SHORTDIR.BAT
```

当按 Enter 时，Edlin 显示下列内容：

```
C:\>EDLIN SHORTDIR.BAT
New file
*I
   1:*
```

星号 * 是 Edlin 的提示符。Edlin 有若干单字母的命令。对此批处理文件，我们将只使用其中的几个命令。有关 Edlin 的全面讨论，可参见 MS-DOS 的用户手册。

为在一批处理文件中插入一命令，必须使用 Edlin 的插入命令 I，并按 Enter 键：

```
C:\>EDLIN SHORTDIR.BAT
New file
*I
   1:*CLS
2:*
```

现在，Edlin 提示输入第二行。敲入 DIR/W 并按 Enter 键：

```
C:\>EDLIN SHORTDIR.BAT
New file
*I
   1:*CLS
   2:*DIR/W
```

由于此批处理文件只需两行内容，所以必须告诉 Edlin 已经输入完正文了。为此，按住 Ctrl 键，再按 C 键。当按 Ctrl-C 时，Edlin 退出插入方式，并显示其提示符：

```
C:\>EDLIN SHORTDIR.BAT
New file
*I
   1:*CLS
   2:*DIR/W
   3:*^C
*
```

为保存该文件，退出 Edlin 而返回至 MS-DOS，可输入 Edlin 的结束命令 E，并按 Enter。结果如下：

```
C:\>EDLIN SHORTDIR.BAT
New file
*I
   1:*CLS
   2:*DIR/W
   3:*^C
*E
```

```
C:\>
```

为运行刚才建立的批处理文件，敲 SHORTDIR 即可。

```
C:\>SHORTDIR
```

并按 Enter 键，MS-DOS 会清除屏幕，然后在屏幕上以列的形式显示文件名的目录清单（如/W 开关所示）。

如前所述，Edlin 还可用来对一个已存在的文件进行修改。例如，可对 SHORTDIR.BAT 批处理文件进行修改以使 MS-DOS 只显示带有 EXE 扩展名的文件。同前面一样，敲入 Edlin，说明文件名：

```
C:\>EDLIN SHORTDIR.BAT
```

由于 SHORTDIR.BAT 文件已经存在，Edlin 就显示下列信息：

```
C:\>EDLIN SHORTDIR.BAT
End of input file
*
```

这表明 Edlin 已读入了整个文件，并准备好进行编辑了。在 Edlin 提示符下敲入数字 1，并按 Enter。Edlin 就会显示该文件的第一行：

```
C:\>EDLIN SHORTDIR.BAT
End of input file
*1
    1:*CLS
    1:*
```

屏幕上出现的第二个 1:* 是 Edlin 的对第一行进行修改的提示符。如果不想对第一行进行修改，按 Enter；否则，敲入第一行的新的内容。在本例中，按 Enter，不对第一行进行修改：

```
C:\>EDLIN SHORTDIR.BAT
End of input file
*1
    1:* CLS
    1:*
*
```

在 Edlin 提示符下，敲入数字 2，并按 Enter。Edlin 会显示第二行的内容，并允许对之进行修改：

```
C:\>EDLIN SHORTDIR.BAT
End of input file
*1
    1:* CLS
```

```
    1:*
*2
    2:* DIR/W
    2:*
```

在本例中，需将第二行原先的内容 DIR/W 改成 DIR *.EXE/W。敲入新的内容，并按 Enter：

```
C:\>EDLIN SHORTDIR.BAT
End of input file
*1
    1:* CLS
    1:*
*2
    2:* DIR/W
    2:* DIR *.EXE/W
*
```

同样，敲入 Edlin 的结束命令 E，并按 Enter，就可以保存修改过的文件内容，并退出至 MS-DOS：

```
C:\>EDLIN SHORTDIR.BAT
End of input file
*1
    1:* CLS
    1:*
*2
    2:* DIR/W
    2:* DIR *.EXE/W
*E

C:\>
```

如果现在运行该批处理文件，MS-DOS 就会先清屏，然后只显示带 EXE 扩展名的文件。

如果终止对文件的编辑，并且不保存所作的任何修改，那么可使用 Edlin 的不保存退出命令 Q。当输入该命令时，Edlin 会显示：

```
Abort Edit(y/n)?
```

如果敲入 y 并按 Enter，则 Edlin 会放弃编辑而退出至 MS-DOS，文件未被作任何修改。如果敲入 N 并按 Enter，则 Edlin 就继续编辑任务。

在本书后面，我们将使用这些编辑方法来建立可改变屏幕布局、并将常用的 MS-DOS 命令赋给键盘上的功能键的批处理文件。我们还将继续讨论 MS-DOS 批处理文件和批处理文件命令。

第 2 章

系统设置

2.1 用 AUTOEXEC.BAT 预置系统

无论你是否知道，你都可能一直在用一 MS-DOS 批处理文件来对具体系统进行预置。每一次系统启动，MS-DOS 都在启动盘的根目录下搜索一特定的名为 AUTOEXEC.BAT 的文件。如果该批处理文件存在，MS-DOS 就执行它所含的命令。否则，MS-DOS 就执行 DATE 和 TIME 命令。

过去，不得不按照本书第 1 章中说明的创建批处理文件的步骤来建立自己的 AUTOEXEC.BAT。但在安装 MS-DOS 的 5.0 版或更高版时，安装软件会自动地创建 AUTOEXEC.BAT 文件。如果从一较早版本升级至 MS-DOS 的 5.0 或更高版本，安装软件则会修改已存在的 AUTOEXEC.BAT 以保证其中含有某些必需的命令。

要记住 AUTOEXEC.BAT 也是一普通的批处理文件，其内容可被修改以满足特定需要。但是，由于 MS-DOS 在每次启动系统时都执行 AUTOEXEC.BAT，所以应该对 AUTOEXEC.BAT 上的任何修改的后果有一全面了解。如果对后果如何无把握，那么就应该在读完本书后再对之进行修改。第 12 章详细讨论了可能会在 AUTOEXEC.BAT 中使用的若干命令。

2.1.1 常用的 AUTOEXEC.BAT 命令

为查看 AUTOEXEC.BAT 文件的内容，敲入如下 TYPE 命令：

```
C:\>TYPE \AUTOEXEC.BAT
```

虽然你的 AUTOEXEC.BAT 文件可能含有其他一些命令，但几乎每一个 AUTOEXEC.BAT 文件都有类似于下面这样的 PATH 和 PROMPT 命令：

```
PATH C:\DOS
PROMPT $P$G
```

MS-DOS 5.0 和更高版的安装程序命令会保证这两行出现在你的 AUTOEXEC.BAT 文件中。如果原先不存在，安装程序会将它们增加进去。

PATH 命令说明 MS-DOS 为了确定外部 MS-DOS 命令和其他可执行程序（如批处理文件和应用程序）而应搜索的目录。上述的 PATH 命令告诉 MS-DOS 在 C 盘的 DOS 目录中搜索。当安装诸如 Windows 这样的软件程序时，其安装程序经常更改 PATH 命令以加入其自身目录。如果加入 Windows，PATH 命令会变成：

```
PATH C:\DOS;C:\WINDOWS
```

注意在 PATH 命令中说明的目录名之间用分号(;)隔开。如果你建立了一个名为 BATCH 的目录用来在其中保存你的批处理文件,可以如下这样地将该目录名加入搜索路径:

```
PATH C:\DOS;C:\WINDOWS;C:\BATCH
```

在上面的 PROMPT 命令中用字符 $p 来告诉 MS-DOS 在命令提示符中显示当前驱动器和目录,字符 $g 表明显示大于号(>)。第 12 章将详细介绍 PROMPT 命令,包括预置你的命令提示符的几种方法。

2.1.2 AUTOEXEC.BAT 的其他用途

除了定义路径名和命令提示符,AUTOEXEC.BAT 还经常被用来装载内存驻留程序(如 DOSKEY),或者用来激活图形用户界面(如 Windows 或 MS-DOS shell)。在本书后面,将陆续说明加至 AUTOEXEC.BAT 文件中可立即提高系统性能的若干命令。

2.2 用 CONFIG.SYS 预置系统

在系统启动时,MS-DOS 还会执行一个名为 CONFIG.SYS 的文件中的语句。对该文件中命令的执行是在执行 AUTOEXC.BAT 文件中的所有命令之前进行的。与含可执行的 MS-DOS 命令名的 AUTOEXC.BAT 文件不同,CONFIG.SYS 的内容是用来控制 MS-DOS 在内存中是如何配置其自身的。此外,CONFIG.SYS 还可装入设备驱动程序(特殊的软件程序)以让 MS-DOS 支持诸如鼠标或 CD-ROM 这样的不同的硬件组成。不要混淆了 CONFIG.SYS 和 AUTOEXC.BAT。它们唯一的相似之处是 MS-DOS 在系统启动时都要用到这两个文件。CONFIG.SYS 中的语句不是能在命令符下敲入执行的命令;它们只能在 CONFIG.SYS 中使用。

过去,建立自己的 CONFIG.SYS 文件必须使用诸如 Edit 这样的编辑器从 MS-DOS 5.0 版开始,安装程序会自动为你创建一个 CONFIG.SYS 文件,或者自动对一已存在的 CONFIG.SYS 文件进行修改。为查看 CONFIG.SYS 文件的内容,可敲入下列 TYPE 命令:

```
C:\>TYPE \CONFIG.SYS
```

下面的表简单地描述了可在 CONFIG.SYS 文件中使用的若干语句:

语 句	功 能
BREAK=	允许、禁止扩展 Ctrl-BREAK 检查。
BUFFERS=	说明磁盘 I/O 缓冲区的数目。
COUNTRY=	设置特定的某国信息(2.1 版本及以后)。
DEVICE=	在常规内存(小于 640KB)中安装设备驱动程序。
DEVICEHIGH=	在 640KB 上安装设备驱动程序(5.0 版及以后版本)。

续表

语 句	功 能
DOS=	设置 MS-DOS 将把其自身装入其中的 RAM 区(640KB 上或之下)，允许使用高内存块(5.0 及以后版本)。
DRIVPARM=	说明块设备特征(3.2 版及以后版本)。
FCBS=	说明 MS-DOS 为较早程序所支持的文件控制块的数目(3.0 版及以后版本)。
FILES=	说明 MS-DOS 一次能打开的文件数。
INCLUDE=	将一组相关的启动语句放入另一组中(6.0 及以后版本)。见第 12 章。
INSTALL=	允许在 CONFIG.SYS 处理中装入内存驻留程序(TSR)(4.0 及以后版本)。
LASTDRIVE=	说明 MS-DOS 支持的最后一个驱动器符(3.0 及以后版本)。
MENUCOLOR=	说明 MS-DOS 启动菜单的颜色(6.0 及以后版本)。见第 12 章。
MENUDEFAULT=	说明 MS-DOS 启动菜单的缺省选项和超时周期(6.0 及以后版本)。见第 12 章。
MENUITEM=	说明 MS-DOS 启动菜单上的一个选项，参见第 12 章(6.0 及以后版本)。
NUMLOCK=	控制启动时 NumLock 键(因此就包括数字小键盘)的状态(6.0 及以后版本)。
REM	允许在 CONFIG.SYS 文件中出现注释行(4.0 及以后版本)。
SET	对一个 MS-DOS 环境变量赋值(6.0 及以后版本)。
SHELL=	定义一个非 MS-DOS 的命令处理程序。
STACKS=	为一次出现多硬件中断的系统提供更多的栈空间(3.2 及以后版本)。
SUBMENU=	为 MS-DOS 启动菜单定义第二级菜单选项(6.0 及以后版本)。见第 12 章。
SWITCHES=	控制交互式启动的可能性以及在安装一扩展键盘时说明常规键盘的功能(5.0 及以后版本)。

在 MS-DOS 6 及以后版本中，CONFIG.SYS 还支持几条新的并且功能很强的语句。第 12 章将详细讨论这些 CONFIG.SYS 语句。

2.3 修改 AUTOEXEC.BAT 及 CONFIG.SYS

在使用 MS-DOS 时，可能需要对 AUTOEXEC.BAT 和 CONFIG.SYS 文件进行修改。值得注意的是，在对这些文件作备份前，不要对它们进行修改。这样，在修改引起了系统启动出错时，就始终能恢复原先的文件。例如，下列命令将 AUTOEXEC.BAT 的内容拷贝至名为 AUTOEXEC.SAV 的文件中：

```
C:\>COPY \AUTOEXEC.BAT \AUTOEXEC.SAV
```

当然，本书的整个核心是为了说明如何自动完成这类任务。在第 4 章，我们将建立一个创建一备份，然后打开 Edit 正文编辑器的批处理文件——只需一条命令。

第 3 章

批处理文件的基本命令与概念

3.1 不显示批处理文件中的命令名

缺省情况下，当执行一个批处理文件时 MS-DOS 会显示它所执行的每个命令的名字。例如，在执行了含 CALCINV、SORTINV、PRINTINV 和 ORDERINV 命令的批处理文件后，屏幕上会出现下列内容：

```
C:\>GETINV

C:\>CALCINV
…Calculating current product inventory

C:\>SORTINV
…Sorting inventory by quantity

C:\>PRINTINV
…Printing inventory

C:\>ORDERINV
…Generating purchase orders

C:\>
```

有时会希望 MS-DOS 在执行批处理文件中的命令时不要显示命令名，以减少屏幕的凌乱或不让批处理文件的用户知道 MS-DOS 正在执行的是什么命令，根据 MS-DOS 版本的不同，可以用两种方法来消除批处理文件中命令名的显示。

从 MS-DOS 3.3 版开始，可以将@字符放在一命令名之始，以消除该命令名的显示。例如下面的批处理文件 VERVOL.BAT 会显示 MS-DOS 的当前版本号以及当前盘的卷号，但是由于 VER 和 VOL 这两条批处理文件的命令前面都有@字符，所以，MS-DOS 就不显示命令名：

```
@VER
@VOL
```

当运行此批处理文件时，MS-DOS 会显示如下所示的 MS-DOS 的版本以及盘的卷号：

```
MS-DOS version 6.2

  Volume in drive C is DOS 6 DISK
  Volume Serial Number is 3921-18D3

C:\>
```

我们可将此输出与另一个批处理文件 VERVOL2.BAT 的输出相比较。在 VERVOL2.BAT 中，VER 和 VOL 命令没有消除命令名的显示。当运行此批处理文件时，MS-DOS 显示下列信息：

```
C:\>VER

MS-DOS version 6.2

C:\>VOL

  Volume in drive C is DOS 6 DISK
  Volume Serial Number is 3921-18D3

C:\>
```

在较早的 MS-DOS 版本中，为消除命令名的显示，必须使用批处理文件命令 ECHO。当第一个批处理文件命令是 ECHO OFF 时，MS-DOS 在执行批处理文件的命令时不显示命令名。例如，下面的批处理文件 VERVOL3.BAT 就使用了 ECHO OFF：

```
ECHO OFF
VER
VOL
```

当运行此批处理文件时，MS-DOS 显示如下：

```
C:\>ECHO OFF

MS-DOS version 3.2

  Volume in drive C is DOS_DISK
  Volume Serial Number is 3921-18D3

C:\>
```

如上所示，MS-DOS 没有显示 VER 和 VOL 命令名。但是，MS-DOS 却显示了下列信息：

```
C:\>ECHO OFF
```

从 MS-DOS 3.3 版开始，可以在 ECHO OFF 命令前加上@字符而消除 ECHO OFF 信息的显示。

如果用的是 MS-DOS 3.3 以前的版本，可以在批处理文件中的 ECHO OFF 命令后立即加上 CLS 命令，以便在显示 VER 和 VOL 命令的输出前清屏：

```
ECHO OFF
CLS
VER
VOL
```

我们将此文件命名为 VERVOL4.BAT。当运行此文件时，MS-DOS 显示 ECHO OFF 信息，清屏，然后再显示下列信息：

```
MS-DOS version 6.2

 Volume in drive C is DOS 6 DISK
 Volume Serial Number is 3291-18D3

C:\>
```

在本章后面，我们将在批处理文件中大量使用 ECHO 命令以向用户输出信息。在第八章中，我们将用 ECHO 命令来设置屏幕颜色及重定义键盘上的键。但是，目前只需知道 ECHO OFF 命令会阻止 MS-DOS 在执行一批处理文件时对命令名的显示。

如果在执行命令时想要显示其中一些的名字（以后会看到这样做的目的），那么可以使用 ECHO ON 命令（这也是 MS-DOS 缺省时的设置）。下面的批处理文件 VERVOL5.BAT 说明了如何使用 ECHO ON 和 ECHO OFF。

```
ECHO OFF
CLS
VER
ECHO ON
VOL
```

当运行此文件时，MS-DOS 会显示下列信息：

```
MS-DOS version 6.2

C:\>VOL

 Volume in drive C is DOS 6 DISK
 Volume Serial Number is 3921-18D3

C:\>
```

可见，MS-DOS 没有显示出 CLS、VER 和 ECHO ON 命令名。在 ECHO ON 命令被

执行后，MS-DOS 就允许显示命令名，因此在这里显示了 VOL 命令名。

注意：许多用户喜欢在 AUTOEXEC.BAT 中的命令前加上@字符，而另外的许多用户则喜欢加上 ECHO OFF 命令。这两种方法都可在 MS-DOS 执行 AUTOEXC.BAT 中的命令时消除对命令名的显示。

ECHO 批处理文件命令

功能：

在 MS-DOS 执行一批处理文件中的命令时消除或允许对命令名的显示。

格式：

ECHO ON

或

ECHO OFF

注释：

ECHO OFF 命令阻止在 MS-DOS 执行批处理文件中的每条命令时对命令名的显示。这样，屏幕凌乱就会得以减少，用户也就不易迷惑。缺省时，MS-DOS 使用的是 ECHO ON，即在每条命令被执行时显示其名字。MS-DOS 在每个批处理文件结束时将 ECHO 重新设置成 ON。从 MS-DOS 3.3 版以后，可以将一批处理文件中的每一命令前加上@字符以消除对命令名的显示。在较早的版本中，可以用 CLS 命令在关闭 ECHO 之后清屏。

除了能允许及阻止命令名的显示，ECHO 批处理文件命令还可让批处理文件显示一些信息。

例子：

假定你的批处理文件中含 TIME 和 DATE 命令：

```
TIME
DATE
```

那么当运行此文件时，MS-DOS(缺省地)在执行每一命令时显示各命令名：

```
C:\>TIME
Current time is 11:26:45.03a
Enter new time:

C:\>DATE
Current date is Thu 04-07-1994
```

```
Enter new date (mm-dd-yy):
```

可以将@ECHO OFF 命令如下那样地加入到批处理文件中：

```
@ECHO OFF
TIME
DATE
```

当运行此文件时，MS-DOS 将消除对命令名的显示，然后显示下列信息：

```
Current time is 11:26:46.03a
Enter new time:
Current date is Wed 04-07-1994
Enter new date (mm-dd-yy):
```

3.2 批处理文件执行的中断

在任何时候想要中断对一批处理文件中命令的执行，都可以先按住 Ctrl 键，再按 C 键。当 MS-DOS 发现 Ctrl-C 被按下后，就会显示 ^C 及下列信息：

```
Terminate batch job(Y/N)?
```

如果按 Y，MS-DOS 就立即终此批处理文件的执行，如果按 N，MS-DOS 就结束当前正在执行的命令，继续从批处理文件中的下一命令开始执行。

为更好地了解此过程，让我们重新考察一下含 TIME 和 DATE 命令的批处理文件 TIMEDATE.BAT。当运行此批处理文件时，MS-DOS 首先执行 TIME 命令：

```
C:\>TIMEDATE
Current time is 11:39:22.25a
Enter new time:
```

如果不是输入了一新的时间，而是按了 Ctrl-C，MS-DOS 就会问你是否想要结束此批处理文件，如下所示：

```
C:\>TIMEDATE
Current time is 11:39:22.05a
Enter new time: ^C

Terminate batch job(Y/N)?
```

在此例中，按 Y。MS-DOS 会结束对整个批处理文件的执行，而将控制立即返回到命令提示符。

再运行一次 TIMEDATE.BAT，并再在 TIME 提示下按 Ctrl-C：

```
C:\>TIMEDATE
Current time is 11:39:22.05a
```

```
Enter new time: ^ C

Terminate batch job(Y/N)?
```

这次按 N。MS-DOS 不是结束该批处理文件，而只是中断当前命令，并继续执行该批处理文件中的下一条命令，在这里是 DATE：

```
C : \>TIMEDATE
Current time is 11 : 39 : 22.25a
Enter new time: ^ C

Terminate batch job(Y/N)? N
Current date is Wed 04-07-1994
Enter new date (mm-dd-yy):
```

3.3 对批处理文件输出的重定向

在命令提示符下敲入一批处理文件名而执行该文件时，不能用 I/O 重定向来将批处理文件的输出送至屏幕之外的某处。但是，在批处理文件中却可以使用 I/O 重定向。例如：DIRAB. BAT 批处理文件可有下列内容：

```
DIR A: *. *
DIR B: *. *
```

可以敲入下列命令来试着执行此文件：

```
DIRAB > DIRAB. OUT
```

但是，在批处理文件中的 DIR 命令的输出不会被定向至文件 DIRAB. OUT。为将 DIR 的输出定向至一文件，必须在批处理文件中将 I/O 重定向加至一特定的 MS-DOS 命令行上。换句话说，DIRAB. BAT 的内容应为：

```
DIR A: *. * > DIRAB. OUT
DIR B: *. * >> DIRAB. OUT
```

当运行此修改过后的 DIRAB. BAT 时，MS-DOS 会将两条 DIR 命令的输出都重定向至文件 DIRAB. OUT 中。单个大于号>创建一新文件。两个大于号>>则将信息附加至一已存在的文件中。

在执行此批处理文件时，该文件会在屏幕上显示两条 DIR 命令。为消除命令名的显示，可使用 ECHO OFF 命令或用@字符：

```
@DIR A: *. * > DIRAB. OUT
@DIR B: *. * >> DIRAB. OUT
```

3.4 批处理文件中的注释

为一批处理文件所起的名字应该反映出该文件的用途。但是，随着批处理文件复杂性的增加，要想记住批处理文件的命令序列，以及每一命令行所使用的开关是困难的。为帮助人们记住或理解批处理文件的内容，MS-DOS 提供了批处理命令 REM，REM 可用来在批处理文件中放置说明或注释。当 MS-DOS 在批处理文件中遇到 REM 时，就跳过它继续执行下一条命令。

试看 REM 命令是如何提高了 GETINV.BAT 批处理文件的可读性的：

```
@ECHO OFF
REM  Name: GETINV.BAT
REM  Function: Executes the commands for
REM  inventory processing.
REM
REM  Written By: K. Jamsa 4/01/93
REM
REM  Use the CALCINV program to determine
REM  the current inventory status.
CALCINV
REM  Use the SORTINV program to generate
REM  a sorted listing of the current inventory.
SORTINV
REM  Use the PRINTINV program to print
REM  hard copies of the current inventory.
PRINTINV
REM  Use the ORDERINV program to
REM  initiate inventory purchase orders.
ORDERINV
```

初看起来，该批处理文件的长度有点可怕。但是，在你读了注释后，该批处理文件的功能就一目了然了。

第一条命令@ECHO OFF 使该批处理文件在执行时不显示 MS-DOS 的命令名（另外也不显示 ECHO 命令本身）。如果没有此 ECHO OFF 命令，MS-DOS 会在执行批处理文件时在屏幕上显示每条 REM 命令（REM 是用来帮助人们理解批处理文件的，但在屏幕上显示 REM 命令会让用户感到迷惑）。

第 2 行到第 6 行解释了批处理文件的目的，编写此文件的人名及时间。给出作者名是为了便于用户就该批处理文件与作者进行交流。给出文件的建立日期可用来确定所使用的文件是否是最新的版本。

如果有谁改变了此批处理文件，他就应该在文件中加上一说明以表明修改的日期及原因。

```
@ECHO OFF
REM Name: GETINV.BAT
REM Function: Executes the commands for
REM inventory processing.
```

若要单步运行批处理文件，需要在 COMMAND 命令后面跟上/Y 和/C 开关以及文件名。MS-DOS 确认用户是否要运行批处理文件，然后停于批处理文件中的每条命令前，并提示用户键入 Y 运行该命令，或键入 N 跳过该命令。下面的例子说明了如何用 COMMAND 命令单步运行批处理文件，该例跳过了 TIME 命令，执行 DATE 命令：

```
C:\>COMMAND /Y /C TIMEDATE
TIMEDATE[Y/N]? Y

C:\>TIME
TIME[Y/N]? N
C:\>DATE
DATE[Y/N]? Y
Current date is Thu 04-07-1994
Enter new date (mm-dd-yy):

C:\>
```

3.6 暂时挂起批处理文件的处理

许多批处理文件都是从头到尾，没有用户干预地执行其中的所有命令。但是，有时需要让批处理文件在你准备好打印机的联机或插入软盘至驱动器之前稍微等待一下，这时可使用批处理文件命令 PAUSE 来显示一信息，并在你按下一键以继续操作之前暂时停止批处理文件的处理。

当 PAUSE 命令被执行时，它显示在其命令行中说明的信息，后面接着下列所示的一行提示：

```
Press any key to continue…
```

当按下任何键，批处理文件就从下一条命令开始继续执行。欲在 PAUSE 提示下结束批处理文件的运行，可按 Ctrl-C。

例如，一个用于打印当前目录中文件清单的名为 PRINTDIR.BAT 的批处理文件可能会在打印前显示下列信息：

```
PAUSE Place the printer on line
Press any key to continue…
```

该批处理文件含下面两条命令：

```
PAUSE Place the printer on line
DIR > PRN
```

当执行此批处理文件时，PAUSE 会在继续执行第二行的 DIR 命令之前等待用户按下任何键。>PRN 会将输出作为一 ASCII 正文文件而定向至打印机上。

下面显示的另一个批处理文件 PAUSETWO.BAT 两次使用了 PAUSE 命令以打印 A 驱动器中的文件清单。第一条 PAUSE 让用户将软盘放至 A 驱动器中，第二条 PAUSE 则提示将打印机联机：

```
PAUSE Put disk with directory to print indrive A
PAUSE Place the printer on line
DIR A : > PRN
```

当运行此文件时，MS-DOS 先执行第一条 PAUSE 命令，并显示下列内容：

```
PAUSE Put disk with directory to print in drive A
Press any key to continue…
```

当将打印机联机并按一键后，批处理文件就执行 DIR 命令，将目录输出重定向至打印机上。

注意，上面两个批处理文件中都没有使用 ECHO OFF 命令来消除对批处理文件命令名的显示。如果加上一条 ECHO OFF 命令，就不会显示在 PAUSE 命令行上说明的信息，这样输出就只有下列内容：

```
Press any key to continue…
```

例如，下面的 NOMSG.BAT 文件就说明了这样的结果：

```
ECHO OFF
PAUSE Place the printer on line
DIR > PRN
```

当运行此批处理文件时，MS-DOS 显示下列信息：

```
C : \>ECHO OFF
Press any key to continue…
```

MS-DOS 不再显示 Place the printer on line。

使用 PAUSE 命令的批处理文件非常类似于用 ECHO ON 以及 ECHO OFF 命令来只显示批处理文件的某些命令名而不显示其他的命令名。下面这个名为 ECHOTEST.BAT 的批处理文件没有消除 PAUSE 信息：

```
ECHO OFF
VER
VOL
ECHO ON
PAUSE Just turned ECHO ON
Press any key to continue…
ECHO OFF
VER
VOL
```

当运行此批处理文件时，MS-DOS 显示下列内容：

```
C:\>ECHO OFF

MS-DOS version 6.20

  Volume in drive C is DOS 6 DISK
  Volume Serial Number is 3921-18D3

C:\>PUASE Just turned ECHO ON
Press any key to continue . . .

C:\>ECHO OFF

MS-DOS version 6.2

  Volume in drive C is DOS 6 DISK
  Volume Serial Number is 3921-18D3

C:\>
```

记住，在使用 MS-DOS 3.3 以后的版本时，可以在 ECHO OFF 命令行的行首加一个@符以不显示 ECHO OFF 信息。

3.6.1 用 PAUSE 引起用户注意

取决于批处理文件中命令的数目以及运行时间的长短，你可能会启动一个批处理文件，然后离开计算机去做别的事而让该文件在无人照看的状态下自行运行。如果该批处理文件执行了一条 PAUSE 命令，那么在你按下一键以继续批处理文件的处理之前可能已过了很多时间。

你可能不只是希望用 PAUSE 来显示一信息，还想要用该命令来在计算机的内部扬声器中产生一个响铃声。马上就能看到，当对计算机用来在屏幕上显示字母、数字和符号的 ASCII 字符集有些了解之后，产生计算机的响铃声是很容易的。

计算机显示或打印机打印的每个字符都是用独一无二的某个值来表示的。ASCII 字符集由从 0 到 127 的 128 个值组成，包括大小写字母、数字 0－9、常用标点符号、以及对计算机有特殊意义的特殊值。例如，ASCII 值 7 就是用来让计算机将响铃声送至内部扬声器的(在第八章，我们将使用从 128 到 255 的一组值——所谓 IBM 扩展字符集——以让 IBM PC 及其兼容机显示出杠和数学字符)。

为了让 PAUSE 命令产生计算机的响铃声，PAUSE 命令行中的信息必须含 ASCII 字符 7。有三种方法可将此字符放入 PAUSE 命令行中。

首先，我们可从键盘上拷贝一名为 BELL.BAT 的批处理文件，如下所示：

```
C:\>COPY CON BELL.BAT
```

敲入 PAUSE，后跟一空格，但不要按回车。再在 PAUSE 命令行中键入 ASCII 字符 7，方法是按住 Alt 键，再按下数字小键盘上的 7。注意：必须用数字小键盘，而不是键盘顶部的那一排数字键。当放开 Alt 键时，MS-DOS 就在屏幕上显示字符 ^G：

```
C:\>COPY CON BELL.BAT
PAUSE ^G
```

再按两次 Alt-7 键组合（在连续敲入 ASCII 值之间，Alt 键必须被放开）。

```
C:\>COPY CON BELL.BAT
PUSE ^G^G^G
```

接着再敲入信息 BELL BELL BELL 并按 F6 以完成此批处理文件的建立：

```
C:\>COPY CON BELL.BAT
PAUSE ^G^G^G BELL BELL BELL
^Z
    1 File(s) copied

C:\>
```

当运行此批处理文件时，MS-DOS 显示 PAUSE，在计算机上响铃三次，显示信息 BELL BELL BELL，然后在另一行上显示 Press any key to continue…，并等待用户的反应。

MS-DOS 在屏幕上用特殊字符 ^G 来表示 ASCII 字符 7。所以，第二种输入可让计算机发出响铃声的 ASCII 字符的方法是用 Ctrl-G 键组合，如前所示，在键盘上拷贝建立 BELL.BAT 文件（这样做会覆盖掉盘上先前的文件版本）。敲 PAUSE，后跟一空格：

```
C:\>COPY CON BELL.BAT
PAUSE
```

现在按 Ctrl-G。MS-DOS 会如下显示：

```
C:\>COPY CON BELL.BAT
PAUSE ^G
```

重复按两次 Ctrl-G，然后敲入 BELL 三次：

```
C:\>COPY CON BELL.BAT
PAUSE ^G^G^G BELL BELL BELL
^Z
    1 File(s) copied

C:\>
```

当运行此文件时，MS-DOS 显示 PAUSE，在计算机上发出三次响铃声，显示 BELL BELL BELL，然后在新的行上再显示信息 Press any key to continue…，并等待你的反

应。建立含响铃字符的文件的第三种方法是使用 Edit。为建立 BELL2.BAT 文件，可用 Edit：

```
C:\>EDIT BELL2.BAT
```

首先敲入 PAUSE 和一空格。

诸如 ^G 这样的字符通常被称作控制符，因为它们是被用于控制程序的进行的。如果在 Edit 中只按了 Ctrl-G，不会有什么效果。如果想在正文中插入一控制字符，就必须首先告诉 Edit 要将该字符当作其本身而非命令。为此，可使用 Ctrl-P 命令。Ctrl-P 可让 Edit 将下一字符看成正如所敲入的那样，而不是试图将它解释为一命令。

为输入 ^G 字符，先按 Ctrl-P，再按 Ctrl-G。方法可以是按住 CTRL 键，按下字母 P，放开 Ctrl 键，再次按住 Ctrl 键，再按下 G 键；或者可以是按住 Ctrl 键，按下 P 键，然后不放开 Ctrl 键，再按 G 键。第一种方法的优点是它强调了所输入的是两个不同的字符组合。第二种方法的优点则是较为简单。

用第二种方法按住 Ctrl 键，再按 P 和 G。这时会看见图 3-1 所示的屏幕。

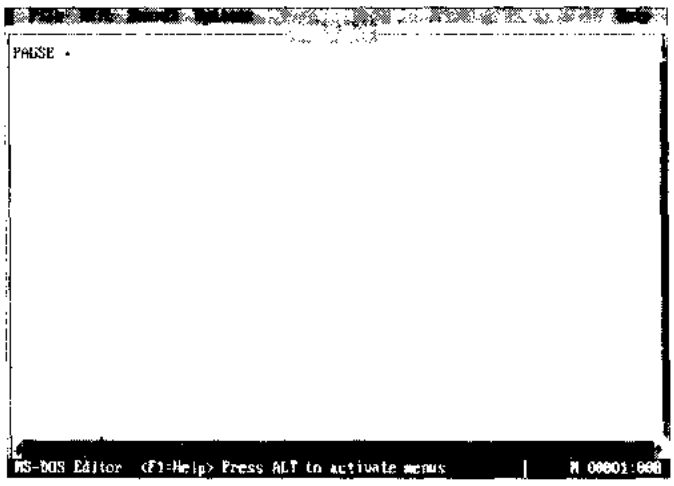


图 3-1 控制字符的屏幕表示(小圆点)。

Edit 所显示的小圆点表示是控制字符。为完成此批处理文件，再键入两次 Ctrl-G 字符，后跟 BELL BELL BELL(在键入每个 Ctrl-G 字符前都必须使用 Ctrl-P)。结束后屏幕如图 3-2 所示。

为保存此新批处理文件至已有名字(BELL2.BAT)中，选择 FILE 菜单，然后选择 SAVE 命令(ALT-F,S)。最后，为返回至命令提示符，选择 FILE 菜单，然后选择 Exit 命令(ALT-F,X)。

当运行此批处理文件时，MS-DOS 应显示 PAUSE，在计算机上发出三次响铃声，在屏幕上显示 BELL BELL BELL，移至一新行，显示

Press any key to continue…

然后等待用户的反应。

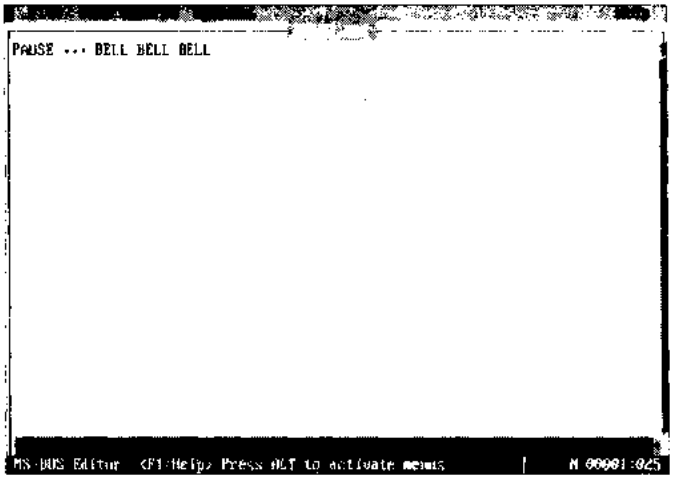


图 3-2 完整的响铃命令行

注意：在 MS-DOS 5.0 以前的版本中，应使用 EDLIN 而不是 Edit。在 EDLIN 中，可以直接输入任何控制符，就如同第二个 COPY CON 例子中那样。

注意：如果用某个字处理器来创建批处理文件，可参看该字处理器的文档来了解如何键入 ASCII 字符。

PAUSE 批处理文件命令

功能：

在显示一可选信息后，暂时挂起对一批处理文件的处理。当用户按了任何键后，就继续批处理文件的执行。

格式：

PAUSE [消息]

注释：

当批处理文件执行 PAUSE 时，MS-DOS 在 PAUSE 命令行中显示可选信息，后跟下列信息：

Press any key to continue…

如果用户按了任何键，批处理文件就从文件中的下一条命令处继续执行。如果用户不想继续执行该批处理文件，就可按 Ctrl—C 并对信息 Terminate batch job(Y/N)? 回答以 Y 而结束此文件。

ECHO OFF 命令可用来消除对 PAUSE 命令行中说明的可选信息的显示。如果文件

中说明了 ECHO OFF，PAUSE 命令仍会显示信息 Press any key to continue…，并挂起批处理的文件的运行，直到用户按下了一键。

例子：

下列批处理文件提示用户将含有名为 PAYROLL 的应用程序的软盘插入 A 驱动器中，并按任何键继续：

```
Pause insert payroll disk in drive A
A:PAYROLL
```

当运行此批处理文件时，MS-DOS 显示下列信息：

```
C:\>Pause insert payroll disk in drive A
Press any key to continue…
```

3.7 用 ECHO 命令显示信息

PAUSE 命令允许批处理文件向用户显示信息。但是，在 PAUSE 显示的每一条信息后，用户必须按一键才能继续。有时会希望批处理文件显示不需要用户按键就能继续的信息或提示。批处理文件命令 ECHO 就能处理这种情形。除了能显示及消除显示命令名，ECHO还可让批处理文件显示一行信息。在用 ECHO 来显示信息前，大多数批处理文件都先通过使用 ECHO OFF 命令来禁止命令名的显示。例如，一名为 MESSAGE.BAT 的批处理文件显示了下列信息：

```
First message
Second message
Third message
```

该批处理文件的命令是：

```
@ECHO OFF
ECHO First message
ECHO Second message
ECHO Third message
```

该批处理文件首先通过使用 ECHO OFF 而禁止对命令名的显示。没有此 ECHO OFF 命令，文件的输出会显得杂乱：

```
C:\>ECHO FIRST MESSAGE
First mseeage

C:\>ECHO SECOND MESSAGE
Second msessage
```

```
C:\>ECHO THIRD MESSAGE
Third message
```

ECHO 命令的信息显示功能有许多用途，包括显示批处理文件菜单、设置屏幕颜色，以及重定义键盘上的键(后两者要使用 ANSI.SYS 设备驱动程序)。注意 ECHO 命令是如何在下列批处理文件中显示版本信息的：

```
@ECHO OFF
REM  Name: GETINV.BAT
REM  Function: Executes the commands for
REM  inventory processing.
REM
REM  Written by: K. Jamsa 4/01/93
REM
REM  Display copyright information.
ECHO GETINV.BAT (Copyright 1993. KAJ Software)
ECHO All rights reserved.

REM  Use the CALCINV program to determine
REM  the current inventory status.
CALCINV

REM  Use the SORTINV program to generate
REM  a sorted listing of the current inventory.
SORTINV

REM  Use the PRINTINV program to print
REM  hard copies of the current inventory.
PRINTINV

REM  Use the ORDERINV program to
REM  initiate inventory purchase orders.
ORDERINV
```

当运行此批处理文件时，MS-DOS 首先显示：

```
GETINV.BAT (Copyright 1993. KAJ Software)
All rights reserved
```

可见，ECHO 可以不中断批处理文件的执行而向用户显示有意义的信息。

3.7.1 ECHO 和 PAUSE 的组合使用

有时会需要显示一特定的信息，然后再给用户作答的机会。你可能想向用户提供比标准的 Press any key to continue…信息更多的信息，或者可能想完全替换掉此信息。为做到这一点，可使用后跟 PAUSE 命令的 ECHO 命令。

下列名为 MYTEXT.BAT 的批处理文件就用了该方法来为用户提供比标准 PAUSE 信息更多的信息：

```
@ECHO OFF
ECHO Press Ctrl-C to cancel execution or
PAUSE
```

运行此文件时，MS-DOS 显示下列内容：

```
Press Ctrl-C to cancel execution or
Press any key to continue…
```

MS-DOS 然后等待用户作答。第一行@ECHO OFF 阻止了批处理文件命令名的显示。第二行显示有关用户选择的进一步信息。第三行也是最后一行，显示了标准的 PAUSE 信息，并接着停止批处理文件的执行，以等待用户反应。

利用这种方法，可以对批处理文件中信息的显示有更大的控制。这些行可被加入任何你希望能在 PAUSE 命令处为用户提供可使用的选项(Ctrl-C 结束，任何其他键继续)这类信息的批处理文件中。

你可能想将 PAUSE 命令产生的信息完全消除掉。为此，必须将输出重定向至 NUL 设备上。下面的批处理文件 MYTEXT2.BAT 就将 PAUSE 信息用一 ECHO 命令产生的信息完全替代了。

```
@ECHO OFF
ECHO Press any key other than Ctrl-C to continue
PAUSE > NUL
```

该批处理文件显示来自 ECHO 命令的信息，然后暂停。标准的 Press any key to continue …信息被重定向至 NUL 设备而不再出现(NUL 设备在存放无用输出时特别有用)。

ECHO 显示信息的批处理文件命令

功能：

显示单行信息。

格式：

ECHO 信息

注释：

除了在批处理文件中用来允许和禁止命令名的显示，ECHO 批处理文件命令还可在批处理文件中显示一行信息。该信息可以很简单，如告诉用户没有找到某文件，也可以含 ANSI 换码序列以清屏或设置屏幕颜色。在本书中，将讨论如何用 MENU 来生成菜单、让计算机发出响铃声，以及重定义键盘上的键。从 MS-DOS 5.0 版开始，可以让 ECHO 显示一空行，方法是在 ECHO 后加上一句点：

```
ECHO.
```

如果用的是 MS-DOS 5.0 之前的版本，可以通过在 ECHO 后跟着由 ALT－255 键组合而创建的 IBM 扩展字符来显示一空行。

例子：

下面的批处理文件使用 ECHO 来让计算机发出响铃声，从而告诉用户货存处理结束了。

```
ECHO OFF
CALCINV
SORTINV
PRINTINV
ORDERINV
ECHO ^G^G^G Inventory processing is complete
ECHO Inventory status information is printing
```

记住，在命令行上按 Ctrl-G 显示 ^G。按 Ctrl-P，然后再按 Ctrl-G 可将响铃声插入 Edit 中。当此批处理文件结束执行时，计算机会发出三声响铃声，并且 MS-DOS 会显示下列信息：

```
Inventory processing is complete
Inventory status information is printing
```

3.7.2 加入空行

有时需要在信息中加入空行。缺省时，如果不带命令行参数(如 ON 或 OFF，或某一信息)地执行 ECHO，则 ECHO 就显示其当前状态：ECHO ON 或 ECHO OFF。例如，如果在命令行上输入下面这样的 ECHO 命令，会看见：

```
C:\>ECHO
ECHO IS ON

C:\>
```

下列名为 SHOWECHO.BAT 的批处理文件调用 ECHO 以显示在批处理文件执行过程中的当前状态：

```
@ECHO Displaying default state
@ECHO
@ECHO Turning ECHO OFF
@ECHO OFF
@ECHO
```

当运行此批处理文件时，MS-DOS 显示：

```
C:\>SHOWECHO
Displaying default state
ECHO IS ON
Turning ECHO OFF
```

```
ECHO IS OFF
```

即使在ECHO命令后放上若干空格，ECHO仍然显示其当前状态。从MS-DOS的5.0版开始，可以在ECHO后紧跟一个句点(ECHO.)来显示一空行。例如，下示批处理文件显示三个空行：

```
@ECHO OFF
ECHO START
ECHO.
ECHO.
ECHO.
ECHO DONE
```

对于MS-DOS 5.0以前的版本，为显示一空行，必须使用IBM的扩展字符值255。ASCII字符集用值32来表示按键盘空格时产生的空字符；如果ECHO命令只包含这种类型的空字符，则ECHO只显示其状态。IBM扩展字符值255也表示空字符，但是，ECHO命令不会将该值视作空字符，所以将IBM扩展字符值255放入ECHO命令行中会显示一空行。

例如，可从键盘上拷贝命令以建立名为BLANK.BAT的批处理文件：

```
C:\>COPY CON BLANK.BAT
```

首先，设置ECHO OFF并清屏：

```
ECHO OFF
CLS
ECHO Skip one line
```

然后，为产生一空行，敲入ECHO后跟一空格，然后按住ALT键，敲入数字小键盘上的255。当放开ALT键时，光标会右移一个位置，这样ECHO命令行就已含IBM扩展字符值255。

用同样的过程可以建立该批处理文件的其他内容：

```
C:\>COPY CON BLAND.BAT
ECHO OFF
cls
ECHO Skip one line
ECHO <ALT-255>
ECHO Skip two lines
ECHO <ALT-255>
ECHO <ALT-255>
ECHO Last line
^Z
    1 File(s) copied
```

```
C:\>
```

当运行此批处理文件时，MS-DOS 显示：

```
Skip one line

Skip two lines

Last line
```

如果在运行此批处理文件时，ECHO 显示了其当前状态，那就说明建立该文件时没有正确输入 ALT-255 组合，这样就不能产生所期望的空行。记住，必须用数字小键盘。如果用 EDLIN 来建立显示一空行的批处理文件，也可以按住 ALT 而敲入 255（数字小键盘上的）以产生所需要的空字符。

显然，用 ECHO。显示一空行比用 ALT-255 要容易。

3.8 在磁盘上保存批处理文件

许多人一开始都将他们的批处理文件放在与 MS-DOS 外部命令相同的目录中，这是因为该目录通常在 AUTOEXC.BAT 文件中被用 PATH＝定义成搜索路径中的一部分。一个更好的方法是为你常使用的批处理文件建立一专门的目录。可以用下列命令建立一名为 BATCH 的目录：

```
C:\>MKDIR \BATCH
```

在将你的批处理文件拷贝至此目录中后，可以在 AUTOEXC.BAT 文件中修改 PATH 命令以将此目录加入至搜索路径中。例如，下面的 PATH 命令就将 BATCH 目录和 DOS 目录都放在了搜索路径中：

```
PATH C:\DOS;C:\BATCH
```

如果要将你的批处理文件目录加至搜索路径中，要记住 MS-DOS 会在搜索批处理文件时检查目录中的每个文件名。如果目录中含有一些很少用到的批处理文件，那么花在检查文件名上的时间就是不必要的，所以应当只把经常使用的批处理文件放入该目录中。

注意：如果像下面这样把目录的顺序反过来，那么有与外部命令相同名字的批处理文件就会代替 MS-DOS 命令而被使用：

```
PATH C:\BATCH;C:\DOS
```

第 4 章

使用参数来提高批处理文件的灵活性

4.1 批处理文件参数的入门

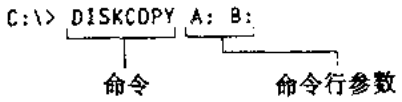
使用批处理文件的目的是为了节省时间和击键次数。让我们来看一个名为 P.BAT，可用来打印你的 AUTOEXEC.BAT 文件备份的批处理文件。该文件内容很简单，只含下列命令：

```
PRINT \AUTOEXEC.BAT
```

当你想打印出你的 AUTOEXEC.BAT 文件的内容时，敲入 P 再按 ENTER 即可。

虽然 P.BAT 能使你少敲几次键，但它并不很有用，因为它只能用来打印出一特定文件的内容。一个更灵活的批处理文件应能打印出任一文件的内容。

当在命令提示符上敲入一命令时，该命令行通常由两部分组成：诸如 DISKCOPY 这样的命令名，诸如磁盘驱动器标识符 A：和 B：这样的命令行参数：



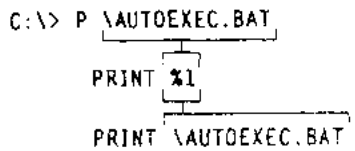
在批处理文件中也可使用参数。例如，通过修改上述 P.BAT，可以很快建立起能打印任一文件的批处理文件：

```
PRINT %1
```

该批处理文件包含 PRINT 命令，但现在该命令让 MS-DOS 打印%1，而不是 AUTOEXC.BAT。为运行此批处理文件，可敲入 P。后跟一文件名：

```
C:\>P \AUTOEXC.BAT
```

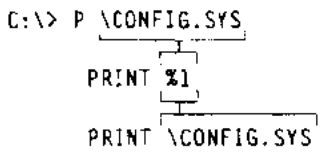
MS-DOS 会将第一个命令行参数——在本例中，是\AUTOEXEC.BAT——赋给变量%1。



当用下列命令行运行此批处理文件时：

```
C:\>P \CONFIG.SYS
```

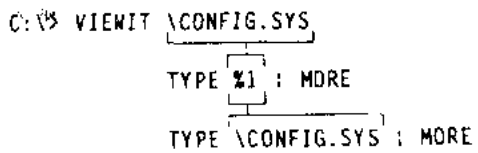
MS-DOS 会将\CONFIG.SYS 赋给%1：



类似地，可以建立名为 VIEWIT.BAT 的可用来一屏一屏地查看一文件内容的批处理文件：

```
TYPE %1 | MORE
```

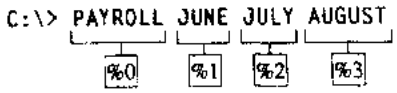
假定用 CONFIG.SYS 文件名激活此批处理文件，则该批处理文件将完成下列参数替换：



通过使用了一个批处理文件的参数，就大大提高了批处理文件的灵活性。为将处理能力提高到更高的程度，MS-DOS 支持从%0 到%9 的批处理文件参数(每次运行一批处理文件时，MS-DOS 就将该文件名赋给%0 变量)。MS-DOS 把第一个命令行参数赋给%1 变量。如果某一批处理文件命令行含几个参数，MS-DOS 就将这些参数赋给连续变量，从%1 开始直至%9(如果有那么多的变量的话)：

```
C:\>PAYROLL JUNE JULY AUGUST
```

在本例中，赋值的情形如下：



下列名为 SHOWNAME.BAT 的批处理文件使用 ECHO 批处理文件命令来显示自己的名字：

```
@ECHO OFF
ECHO %0
```

当运行此批处理文件时，MS-DOS 显示：

```
SHOWNAME
```

通过使用 ECHO 命令，可以显示从%0 到%9 的批处理文件参数的值。例如，如果某个批处理文件需要在其处理期间使用自己的名字(这种情形比较少见)，那么可以用%0 来让 MS-DOS 提供该名字。

名为 SHOWVAR.BAT 的批处理文件使用了 ECHO 命令来显示%0 到%9 的每一个参数：

```
@ECHO OFF
```

```
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
```

可以用下列命令行运行此批处理文件：

```
C:\>SHOWVAR ONE TWO THREE
```

MD-DOS 然后就显示：

```
C:\>SHOWVAR ONE TWO THREE
SHOWVAR ONE TWO THREE
```

类似地，可以用下列命令行来运行该批处理文件：

```
C:\>SHOWVAR A B C D E F G H
```

MS-DOS 就显示：

```
C:\>SHOWVAR A B C D E F G H
SHOWVAR A B C D E F G H
```

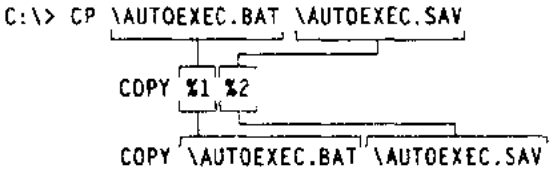
可见，如果想建立功能较强的批处理文件，就必须使用批处理文件参数。下列名为 CP.BAT 的批处理文件使用了%1 和%2 批处理文件参数来简写了 COPY 命令：

```
COPY %1 %2
```

可以用下面的命令行来运行此批处理文件：

```
C:\>CP \AUTOEXEC.BAT \AUTOEXEC.SAV
```

MS-DOS 就会如下这样地给参数赋值：



前面的几个例子假定在你激活批处理文件时说明了文件名及其扩展名。下面的批处理文件 BATEDIT.BAT 假定你只说明了文件名：

```
COPY %1.BAT %1.SAV
EDIT %1.BAT
```

在第一章中，我们讲到过可以用 MS-DOS 提供的 Edit 命令来创建和编辑批处理文件。当用 Edit 修改一个现存文件时，不会保存原文件的备份。BATEDIT.BAT 批处理文件完成两件任务：首先，它对将要编辑的批处理文件作备份，即将该文件的内容拷贝至扩展名为 SAV 的文件中；其次，该批处理文件调用 Edit 命令，这样就能对文件进行修改了。假定想要编辑 CP.BAT 文件，可以这样运行 BATEDIT：

```
C:\>BATEDIT CP
```

在此例中，该批处理文件进行下列的参数替换：

COPY %1.BAT %1.SAV ——→ COPY CP.BAT CP.SAV
EDIT %1.BAT ——————→ EDIT CP.BAT

通过用 BATEDIT.BAT 来编辑已存在的批处理文件，就能保证在你作出的某一修改使批处理文件不能执行的情况下，总有该文件先前内容的一个备份。在此例中，BATEDIT.BAT 提供了 Edit 没有提供的一种功能。但是，该批处理文件仍不完善。如果用 BATEDIT.BAT 来创建一个新的批处理文件，那么当该批处理文件执行时，就会在屏幕上出现 File not found 的错误信息。这是因为 COPY 命令试图对一个并不存在的文件进行拷贝。第五章将讨论在批处理文件中如何检查某一文件是否存在并因此作相应处理。

本书后面还将讨论如何使用批处理文件命令 IF 和 FOR。我们将会看到参数可以很好地提高使用这些命令的批处理文件的功能。

使用批处理文件参数

每次运行一个批处理文件时，MS-DOS 就将该文件的命令行(包括批处理文件名和所有参数)的各组成成分赋给各变量。MS-DOS 可允许让批处理文件用%1 至%9 的变量来存取最多为 9 个的参数。例如，下列命令行运行批处理文件 SHOWIT.BAT，并将 AUTOEXEC.BAT 和 CONFIG.SYS 文件名作为参数传递给批处理文件：

```
C:\>SHOWIT AUTOEXEC.BAT CONFIG.SYS
```

MS-DOS 将第一个参数 AUTOEXEC.BAT 赋给%1 变量，将第二个参数 CONFIG.SYS 赋给%2 变量。因此在此批处理文件中，有关命令就能使用这些参数了：

```
@ECHO OFF
TYPE %1
TYPE %2
```

MS-DOS 始终将批处理文件自己的名字(在这里是 SHOWIT)赋给%0 变量。本书后面还将讨论如何用 SHIFT 命令来访问多于 9 个的参数，以及如何在批处理文件中进行检查以保证用户对不同的参数有所说明的值。

第 5 章

批处理文件的编程

5.1 在批处理文件中进行条件测试

我们至今所讨论的所有批处理文件都是从文件中的第一条命令开始，从头到尾逐条执行命令。随着批处理文件复杂性的增加，可能会有必要根据一组特定条件而控制 MS-DOS 执行哪些命令。批处理文件命令 IF 就可以让批处理文件能够只在一给定条件为真时才执行一 MS-DOS 命令。IF 命令可让一个批处理文件对六个特殊的条件进行测试：三个 IF 条件——EXIST，串 1==串 2，以及 ERRORLEVEL；另外三个条件是它们的非，一共是六个条件。

5.1.1 对文件是否存在进行测试

第一个 IF 条件测试在盘上是否存在一特定的文件。该命令的格式如下：

```
IF EXIST FILENAME.EXT DOS 命令
```

当 MS-DOS 遇到 IF 命令的此格式时，它就在盘上查找所说明的文件。文件名参数可以含一完整的 MS-DOS 路径名——磁盘驱动器号、子目录名和文件名——或者可以是 MS-DOS 将在当前目录中搜索的文件名。不管哪种情况，如果文件存在，MS-DOS 就执行所说明的命令。如果文件不存在，MS-DOS 就从批处理文件中的下一条命令开始执行。

5.1.2 对文件是否不存在进行测试

正如可以用 IF EXIST 来确定某一文件是否存在，你也能用 IF NOT EXIST 来确定一文件是否不存在。此时的命令格式如下：

```
IF NOT EXIST filename.ext DOS 命令
```

IF 命令的此形式的含意与上一条正好相反。如果该文件不存在，MS-DOS 就执行所说明的命令。如果该文件存在，MS-DOS 就从批处理文件中的下一条命令继续执行。

5.1.3 具体使用 IF EXIST

为更好地了解 IF EXIST 命令，考察在前一章中建立的名为 T.BAT 的批处理文件。该批处理文件显示一个文件：

```
TYPE %1 | MORE
```

在运行该文件时，如果在命令行说明了一个不存在的文件。那么 MS DOS 会显示下列信息：

```
File not found - filename.ext
```

通过引入一条 IF EXIST 命令首先测试所说明的文件在盘上是否存在可以大大改进该批处理文件的用途。为此，必须去掉至 MORE 的重定向。如果该文件存在，MS-DOS 就执行 TYPE 命令。如果该文件不存在，MS-DOS 就不执行 TYPE 命令。这样，就不会出现错误信息。新的 T.BAT 文件内容是：

```
IF EXIST %1 TYPE %1
```

你可以分别用存在的文件以及不存在的文件来运行该批处理文件，看看效果如何。

在第 4 章中，建立了批处理文件 BATEDIT.BAT，它用下列命令来将所说明的一批处理文件的内容拷贝至带 SAV 扩展名的文件中，然后再将此批处理文件装入 EDIT：

```
COPY %1.BAT %1.SAV
EDIT %1.BAT
```

如前面我们简单讨论过的那样，如果所说明的批处理文件不存在，COPY 命令就显示一条 FILE NOT FOUND 出错信息。但是，使用 IF EXIST 命令就能在只有存在所说明的批处理文件时才进行 COPY 操作，具体方法是将原来的批处理文件内容作如下修改：

```
@ECHO OFF
IF EXIST %1.BAT COPY %1.BAT %1.SAV
EDIT %1.BAT
```

如你所知，在 MS-DOS 6 及早期版本中如果某存在的文件不是只读、系统或隐含文件，那么 COPY 命令会将此文件内容覆盖掉。为防止 COPY 不小心覆盖掉一个重要文件，可以建立一个名为 MYCOPY.BAT 的简单的批处理文件，它用 IF EXIST 命令来确定目标文件在盘上是否存在。如果存在，批处理文件就能用 PAUSE 批处理文件命令来发出警告。按任何键才可以进行文件拷贝操作。或者，可以按 CTRL-C 来结束该批处理文件的运行。

例如，如果运行一个含下列命令的名为 MYCOPY.BAT 的批处理文件：

```
@ECHO OFF
IF EXIST %2 ECHO Target file already exists
ECHO Press Ctrl-C to cancel the copy
PAUSE
COPY %1 %2
```

如果所说明的文件存在，该批处理文件就显示下列内容：

```
Target file already exists
Press Ctrl-C to cancel the copy
Press any key to continue . . .
```

注意：如果目标文件是只读、系统或隐含的，那么即使用户继续，该文件拷贝操作的结果

也会是 ACCESS DENIED-FILENAME.EXT 信息；除了此三种情形之外，目标文件都将被覆盖。

还可以进一步在此批处理文件显示的信息中引入%2参数：

```
@ECHO OFF
IF EXIST %2 ECHO Target file %2 already exists
ECHO Press Ctrl-C to cancel the copy
PAUSE
COPY %1 %2
```

在此例中，如果将 AUTOEXEC.BAT 作为所说明的文件而运行此批处理文件，则会显示：

```
Target file AUTOEXEC.BAT already exists
Press Ctrl-C to cancel the copy
Press any key to continue . . .
```

该批处理文件显示上述信息的原因是显而易见的。

IF EXIST 批处理文件命令

功能：

测试一文件是否存在，并在存在时，执行所说明的MS-DOS命令。

格式：

IF EXIST filename.ext DOS 命令

注释：

IF EXIST 条件在盘上寻找所说明的文件。文件名可以包含一完整的MS-DOS路径名或者是在当前目录中的一文件名。

如果MS-DOS找到了所说明的文件，MS-DOS就执行所说明的命令。如果该文件不存在，MS-DOS就从批处理文件中的下一条命令处开始执行。

例子：

下列批处理文件测试AUTOEXEC.BAT文件在根目录中是否存在。如果存在，该批处理文件就显示它的内容，然后继续执行VER命令。如果该文件不存在，MS-DOS就从VER命令处继续执行此批处理文件：

```
IF EXIST \AUTOEXEC.BAT TYPE \AUTOEXEC.BAT
VER
```

5.1.4 对字符串是否相等进行测试

第三个 IF 条件测试两个字符串是否相等。一个字符串是一个或多个字符序列。该 IF 条件的格式如下：

```
IF 串1==串2  DOS 命令
```

如果在等号两边的字符串完全相等，MS-DOS 就执行所说明的命令。如果串不同，MS-DOS 就从批处理文件中的下一条命令继续执行。

对串进行的比较很精确，包括大小写字母。

此 IF 命令格式的主要用途是测试赋给一个批处理文件参数的值。例如，下列批处理文件 IF-TEST.BAT 测试第一个参数(%1)的值是否是 AUTOEXEC.BAT：

```
@ECHO OFF
IF %1==AUTOEXEC.BAT ECHO Parameter is AUTOEXEC.BAT
```

运行该文件：

```
C:\>IF-TEST AUTOEXEC.BAT
```

由于该参数等于 AUTOEXEC.BAT，所以此批处理文件显示下列信息：

```
Parameter is AUTOEXEC.BAT
```

随着你的批处理文件复杂性的增加，你会常常用到此条 IF 命令。

IF 串1==串2 批处理文件命令

功能：

测试两个字符串是否相等，并在相等时执行所说明的 MS-DOS 命令。

格式：

```
IF 串1==串2  DOS 命令
```

注释：

IF 串1==串2 条件逐个字母地比较两个字符串。如果它们完全匹配(包括大小写)，MS-DOS 就执行后面跟着的命令。如果有一个或多个字母不同，MS-DOS 就从批处理文件中下一条命令处继续执行。

如果在 IF 命令行中没有说明两个字符串，MS-DOS 会显示出错信息 Syntax error。该错误在测试一个无值的批处理文件参数时常常出现，如：

```
IF %1==MONTHLY BACKUP ECHO MONTHLY
```

如果没有为%1 说明值，IF 条件就成为：

```
IF ==MONTHLY_BACKUP ECHO MONTHLY
```

由于该命令只含一个串，MS-DOS 就显示 Syntax error 信息。为防止这种情况，可将字符串用双引号括起，如下：

```
IF "%1"=="MONTHLY_BACKUP" ECHO MONTHLY
```

然后，在没有为%1 说明一值时，MS-DOS 会将空串与串 MONTHLY_BACKUP 比较，如下：

```
IF""=="MCONTHLY_BACKUP" ECHO MONTHLY
```

由于该命令行现在有了两个串，MS-DOS 就不会再生成 Syntax error 信息。

例子：

下列批处理文件测试命令行参数的值是否是\AUTOEXEC.BAT。如果是，批处理文件就显示文件 AUTOEXEC.BAT 的内容：

```
IF "%1"=="\AUTOEXEC.BAT" TYPE \AUTOEXEC.BAT
```

5.1.5 对字符串是否不相等进行测试

正如 MS-DOS 可测试两个串是否相等，也可以用 IF 来确定两个串是否相异。第四个 IF 条件用 NOT 算符来确定两个串是否相异。格式如下：

```
IF NOT 串1==串2 DOS 命令
```

如果两个串不等，MS-DOS 就执行所说明的命令。如果两个串相等，MS-DOS 就从下一条批处理文件命令处继续执行。

这两个串中的任一个都可以由命令行参数来提供。例如，下列批处理文件 IFTEST2.BAT 在命令行参数不等于 AUTOEXEC.BAT 时显示一信息：

```
@ECHO OFF
IF %1==AUTOEXEC.BAT ECHO Parameter is AUTOEXEC.BAT
IF NOT %1==AUTOEXEC.BAT ECHO Parameter is not AUTOEXEC.BAT
```

5.1.6 具体使用串测试

在第八章中，我们将讨论如何用 ANSI.SYS 设备驱动程序来设置屏幕颜色。在这里，可以在一批处理文件中使用 IF 命令以确定在该批处理文件命令行上说明的颜色。

下面的批处理文件 COLOR.BAT 显示被说明成第一个命令行参数的颜色名：

```
@ECHO OFF
IF %1==RED ECHO Color is Red
IF %1==BLUE ECHO Color is Blue
IF %1==WHITE ECHO Color is White
```

例如，可以用下列命令行运行 COLOR.BAT 批处理文件：

```
C:\> COLOR BLUE
```

该批处理文件然后显示：

```
COLOR is Blue
```

还可以用下述命令行运行此批处理文件：

```
C:\> COLOR RED
```

MS-DOS 就如下匹配此表达式：

```
IF %1==RED ECHO Color is Red
```

该批处理文件接着就显示：

```
Color is Red
```

为了让 IF 命令认为两个串相等，这两个串必须逐个字母(包括大小写)都匹配。例如，可以用下述命令行运行此批处理文件：

```
C:\> COLOR Blue
```

MS-DOS 找不到匹配的颜色，因为 IF 命令认为 Blue 和 BLUE 不相等。

为更好地理解 MS-DOS 处理 IF 命令的方式，将 ECHO OFF 命令从前面的批处理文件中去掉。这时再用下列命令行运行此批处理文件，就会看见 MS-DOS 完成的实际比较：

```
C:\> COLOR BLUE
```

然后再不带命令行参数地运行此文件：

```
C:\>COLOR
```

每次 MS-DOS 试图执行一条 IF 命令，如果该 IF 命令的格式或语法不正确，它就会显示下列信息：

```
Syntax error
```

记住，IF 命令的此格式比较两个串。如果没有对%1 参数赋一个值，IF 命令就只有一个串用于比较，%1 参数不存在。由于 MS-DOS 需要两个串，MS-DOS 就会显示一个语法出错信息。

为防止该问题的出现，需要确保 IF 命令始终有两个串用于比较，方法是在串上加双引号。这样就在%1 不存在时产生一空串""。此批处理文件的内容成为这样：

```
IF "%1"=="RED" ECHO Color is Red
IF "%1"=="BLUE" ECHO Color is Blue
IF "%1"=="WHITE" ECHO Color is White
```

然后用此命令行运行此文件：

```
C:\> COLOR BLUE
```

MS-DOS 匹配条件：

```
"BLUE"=="BLUE"
```

如果不带参数地运行此批处理文件，比较就变为：

```
IF "" =="RED" ECHO Color is Red
```

其中无任何东西的一对双引号即是所谓的空串。由于 IF 命令能将空串与颜色值比较，所以就消除了语法错误 。

在许多批处理文件中，还需测试%1 的值是否在命令行上被说明了。下面的批处理文件 TEST%1.BAT 就用了空串来完成此任务：

```
IF "%1"=="" ECHO No parameter value specified
```

可以用下列命令行来运行此批处理文件：

```
C:\>TEST%1
```

这样表达式就变成：

```
IF ""==="" ECHO No parameter value specified
```

MS-DOS 显示下列信息 ：

```
No parameter value specified
```

5.1.7 对返回状态值进行测试

许多 MS-DOS 命令和程序都提供返回状态值以说明它们是否被成功地执行，如果没有，就说明错误原因。例如，DISKCOPY 命令就返回下列返回状态值中的一个：

值	含　义
0	成功的磁盘拷贝
1	由于非致命磁盘错误引起的拷贝失败
2	由于用户按了 CTRL-C 而引起的拷贝未完成
3	由于致命磁盘错误而引起的拷贝失败
4	内存不够或驱动器不对

IF ERRORLEVEL 命令可让批处理文件检查任一命令的返回状态值，并相应地继续处理。此命令的格式如下：

```
IF ERRORLEVEL 值 DOS 命令
```

当 MS-DOS 遇到一条 IF ERRORLEVEL 命令，它就检查前一条命令的返回状态值。如果返回状态值大于或等于所说明的值，MS-DOS 就执行所说明的命令。如果返回状态值小于所说明的值，MS-DOS 就从批处理文件中的下一条命令处继续执行。

格式：

IF ERRORLEVEL 值 DOS 命令

注释：

许多程序和 MS-DOS 命令都返回一个说明它们是否成功地完成了任务的返回状态值。例如，FORMAT 命令就返回下列返回状态值：

值	含　意
0	成功的格式化
3	由于用户按了 CTRL-C 而导致格式化未完成
4	由于出错而导致格式化未完成
5	根据提示而导致格式化的终止

并不是所有的 MS-DOS 命令都提供了返回状态值。

IF ERRORLEVEL 命令让批处理文件测试一程序的返回状态值，然后根据测试结果作相应的进一步处理。当 MS-DOS 遇到一条 IF ERRORLEVEL 命令时，它就将前一条 MS-DOS 命令的返回状态值与在 IF 命令中所说明的值相比较。如果返回状态值大于或等于 IF 命令所说明的值，MS-DOS 就执行相应的命令。如果返回状态值小于所说明的值，MS-DOS 就从批处理文件中的下一条命令继续执行。如果用的是 IF NOT ERRORLEVEL 命令，就正好相反。

例子：

下列批处理文件执行 FORMAT 命令，并显示基于 FORMAT 的返回状态值的完整状态信息：

```
@ECHO OFF
FORMAT A:
IF ERRORLEVEL 5 GOTO NO_RESPONSE
IF ERRORLEVEL 4 GOTO ERROR
IF ERRORLEVEL 3 GOTO USER_CTRLC
ECHO Successful FORMAT operation
GOTO DONE
:NO_RESPONSE
ECHO Fixed disk will not be formatted
GOTO DONE
:ERROR
ECHO Error in processing, FORMAT incomplete
GOTO DONE
:USER_CTRLC
ECHO FORMAT incomplete due to Ctrl-C
:DONE
```

注意：批处理文件首先测试最大的返回状态值。记住，如果返回状态值大于或等于由 IF 命令所说明的值，MS-DOS 就执行所说明的命令。如果该批处理文件首先测试返回状态值是否为 3，那么 MS-DOS 会始终执行 :USER_CTRLC 标号后面的处理，而不论返回状

态值是3、4或5，这是因为这些返回状态值都是大于或等于3的。通过将测试次序倒过来，该批处理文件就能对各返回状态值分别执行正确的那组命令。

通过组合IF ERRORLEVEL和IF NOT ERRORLEVEL，可以测试一特定的返回状态值。例如，可以用下列命令测试返回状态值是否为3：

```
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 GOTO USER_CTRLC
```

5.1.9 对返回状态值是否小于一特定值进行测试

将NOT算符用于IF ERRORLEVEL命令，批处理文件就能测试一返回状态值是否小于一具体值。如果是，MS-DOS就执行所说明的命令；否则，它就从批处理文件中的下一条命令处继续执行。IF NOT ERRORLEVEL命令的格式如下：

IF NOT ERRORLEVEL 值 DOS命令

5.1.10 对特定ERRORLEVEL值进行测试

当一个批处理文件变得较为复杂时，可能会需要将一条IF ERRORLEVEL命令与一条IF NOT ERRORLEVEL命令组合在一起以测试特定的返回状态值。例如，下列批处理命令在DISKCOPY命令后测试从0至4的返回状态值：

```
@ECHO OFF
DISKCOPY A: B:
IF ERRORLEVEL 0 IF NOT ERRORLEVEL 1 ECHO EXIT status 0
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 ECHO EXIT status 1
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 ECHO EXIT status 2
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 ECHO EXIT status 3
IF ERRORLEVEL 4 IF NOT ERRORLEVEL 5 ECHO EXIT status 4
```

在第一条IF命令中，第一个测试(IF ERRORLEVEL 0)保证返回状态值至少为0；第二个测试(IF NOT ERRORLEVEL 1)保证返回状态值小于1。将这两个测试组合起来，该命令就保证返回状态值为0。类似地，第二条IF命令保证返回状态值为1，等等。

测试及使用返回状态值的功能是在本书后面将出现的提高批处理文件功能的关键。建立批处理文件的目标是自动执行一组MS-DOS命令。记住，你的批处理文件应该能够处理意外的错误，IF ERRORLEVEL命令就正好能让批处理文件做到这一点。在本章后面，我们还将介绍MS-DOS 6中的CHOICE命令，它能根据批处理文件用户在键盘上的应答而评价不同的返回状态值。

5.2 NOT 算 符

我们刚刚说过，IF命令可让批处理文件在只有特定条件被满足时才执行某一MS-DOS命令。该条件可以是是否存在一特定文件，两个串是否相等，或者一命令的返回状态值是否大于或等于一具体值。我们还讨论了用NOT算符进行测试的这些条件的相反情

形(一文件不存在,两个串不相等,或者一返回状态值小于一特定值)。

NOT 算符改变某一条件的结果。例如,如果某一条件为真,那么用 NOT 算符就使它变成假。反过来,如果条件为假,用 NOT 算符则使它为真。

通过使用两条 IF 语句(其中一条用了 NOT),就可以让批处理文件根据某一单个条件的结果而完成不同的动作。在本章前面有一个 T.BAT 文件,它测试由%1 所说明的文件是否存在,如果是,就显示该文件的内容。可以对此文件进行修改以完成允许批处理文件显示下示信息的第二个测试:

```
filename.ext does not exist
```

修改后的 T.BAT 文件内容如下:

```
@ECHO OFF
IF EXIST %1 TYPE %1
IF NOT EXIST %1 ECHO %1 does not exist
```

如果 EXIST %1 条件的结果为假(意味着该文件不存在),那么 NOT 算符就将此结果改成真,这样 MS-DOS 就执行 ECHO 命令。如果 EXIST %1 条件的结果为真(意味着该文件存在),NOT 算符就将此结果改成假,这样 MS-DOS 就不执行 ECHO 命令。

NOT 算符在 MS-DOS 批处理文件中被广泛使用。下面是另一个例子。批处理文件 COMPSTR.BAT 比较批处理参数%1 和%2 中的两个串。如果它们相同,该文件就显示信息:

```
串 1 AND 串 2 are identical
```

如果这两个串不同,就显示:

```
串 1 AND 串 2 are not the same
```

该批处理文件的命令如下:

```
@ECHO OFF
IF "%1"=="%2" ECHO %1 AND %2 are identical
IF NOT "%1"=="%2" ECHO %1 AND %2 are not the same
```

该批处理文件需要二条 IF 命令来完成其功能。第一条 IF 命令处理串相等的条件。第二条 IF 命令处理串不同的条件(程序员会看出这与一条 IF-THEN-ELSE 语句的相似性)。

NOT 批处理文件算符

功能:

取一 IF 命令中真或假值的相反结果。如果测试的结果为假,NOT 就将它改成真。相反,如果测试的结果为真,NOT 就将它改为假。

格式：

IF NOT 条件 DOS 命令

注释：

NOT 算符可和三个 IF 条件中的任何一个一起使用：

```
IF NOT EXIST filename.ext DOS 命令
IF NOT 串 1==串 2 DOS 命令
IF NOT ERRORLEVEL 值 DOS 命令
```

例子：

下列批处理文件测试\AUTOEXEC.BAT 文件是否在当前盘的根目录中。如果不在，该批处理文件就显示信息以让你创建该文件。

```
@ECHO OFF
IF NOT EXIST \AUTOEXEC.BAT GOTO NO_FILE
GOTO DONE
:NO_FILE
ECHO Your root directory does not contain the batch
ECHO file AUTOEXEC.BAT. This batch file lets you
ECHO specify one or more commands that you want MS-DOS
ECHO to execute each time your system starts. Most
ECHO users place the MS-DOS PRINT, PROMPT, and PATH
ECHO commands in this file.
:DONE
```

如果 AUTOEXEC.BAT 不在当前盘的根目录中，该批处理文件就显示：

```
Your root directory does not contain the batch
file AUTOEXEC.BAT. This batch file lets you
specify one or more commands that you want MS-DOS
to execute each time your system starts. Most
users place the MS-DOS PRINT, PROMPT, and PATH
commands in this file.
```

5.3 用 GOTO 进行分支转移

我们至今所讨论的所有批处理文件都是从批处理文件中的第一条命令开始执行，然后从顶向下一条命令一条命令地执行。IF 命令可以让 MS-DOS 只在一特定条件被满足时才执行某命令。随着具体应用程序在复杂性上的增加，有时会需要让 MS-DOS 在特定情形下完成一组特定的命令，或者在其他情形下，跳过一条或多条命令。GOTO 命令就可让批处理文件从一条命令分支至另一条。其格式如下：

GOTO DOS 批处理标号

让我们来看一个简单的例子。下面的批处理文件 REMOVE.BAT 用%1 参数来显示一文件的内容，然后删除该文件——除非用 CTRL-C 来结束该批处理文件：

```
@ECHO OFF
IF "%1"=="" GOTO NO_FILE
TYPE %1
ECHO About to delete %1
ECHO Press Ctrl-C to stop
PAUSE
DEL %1
GOTO DONE
:NO_FILE
ECHO Need to specify filename
:DONE
```

当运行此批处理文件时，REMOVE.BAT 一开始测试是否有一文件被说明成命令行参数。如果没有，该批处理文件就分支转移至标号 NO_FILE 处：

```
@ECHO OFF
IF "%1"=="" GOTO NO_FILE
TYPE %1
ECHO About to delete %1
ECHO Press Ctrl-C to stop
PAUSE
DEL %1
GOTO DONE
:NO_FILE
ECHO Need to specify filename
:DONE
```

该批处理文件显示下列信息后结束：

```
Need to specify filename
```

如果说明了一个文件，那么该批处理文件就显示其内容。然后显示下列信息：

```
About to delete filename.ext
Press Ctrl-C to stop
Press any key to continue
```

如果按了一个键来删除该文件，该批处理文件就执行 DEL 命令，然后用 GOTO 来转移至批处理文件结尾处的:DONE 标号。如果你不想删除该文件，就可以按 CTRL-C 来结束此批处理文件的运行。

GOTO 命令把分支转移至所说明的标号处。一个批处理文件中的标号必须以一冒号开始，并可含任意数目的字符——虽然 MS-DOS 只识别前八个。该批处理文件用了标号 NO_FILE 和:DONE。注意含标号的行是以冒号开始的，而 GOTO 命令中说明的标号则没有冒号：

```
@ECHO OFF
IF "%1"=="" GOTO NO_FILE ──────────────────────GOTO 命令中的标号
TYPE %1
ECHO About to delete %1
ECHO Press Ctrl-C to stop
PAUSE
DEL %1
GOTO DONE ──────────────────────┐
:NO_FILE ──────────────────────┐
ECHO Need to specify filename  ├──── MS-DOS 的带冒号的标号
:DONE ──────────────────────┘
```

当 MS-DOS 在批处理文件中遇到以冒号开始的一行时，就知道该行含一标号从而不会将该标号作为一命令而执行。不论 ECHO 的状态如何，MS-DOS 在执行一批处理文件时不会显示标号。

批处理文件的应用几乎是无限止的。例如，下面的批处理文件 ADDRBOOK.BAT 提供了一联机的含重要地址与电话的地址簿。为使用此批处理文件，只需说明某人的姓名（用大写字母），该批处理文件就会显示如下所示的该人的电话号码及地址：

```
C:\>ADDRBOOK SMITH
Gary Smith
28000 South 5th Street
Seattle, Washington 98444
(206)555-5555
```

如果说明了在地址簿中不存在的某人的名字，该批处理文件就显示如下信息：

```
C:>ADDRBOOK KAHN
No such entry KAHN
```

下列批处理文件是 ADDRBOOK.BAT 的内容：

```
@ECHO OFF
IF "%1"=="" GOTO NO NAME
IF "%1"=="SMITH" GOTO SMITH
IF "%1"=="JONES" GOTO JONES
IF "%1"=="ALLEN" GOTO ALLEN
ECHO No such entry %1
GOTO DONE

:SMITH
ECHO Gary Smith
ECHO 28000 South 5th Street
ECHO Seattle, Washington 98444
ECHO (206) 555-5555
GOTO DONE

:JONES
ECHO Alex Jones
ECHO 12345 Second Street
ECHO Phoenix, Arizona 85023
ECHO (602) 555-5555
```

```
GOTO DONE

:ALLEN
ECHO Joe Allen
ECHO 1321 Main Street
ECHO Las Vegas, Nevada 89119
ECHO (702) 555-5555
GOTO DONE

:NO_NAME
ECHO No name specified
:DONE
```

该批处理文件只是察看一下第一个参数的内容，并确定你希望知道的电话号码和地址。由于 IF 命令将大小写字母看作是不同的，所以必须将名字都说明成大写的。

如果你的计算机上有一调制解调器，那么你就可以建立一个名为 AUTOCALL.BAT 的批处理文件，它能自动拨出你所要打的电话。假定你的调制解调器与 COM1 相联，该批处理文件的内容可以是：

```
@ECHO OFF
IF "%1"=="" GOTO NO_NAME
IF "%1"=="SMITH" ECHO ATDT 1-206-555-5555 > COM1
IF "%1"=="JONES" ECHO ATDT 1-602-555-5555 > COM1
IF "%1"=="ALLEN" ECHO ATDT 1-702-555-5555 > COM1
PAUSE
ECHO ATH > COM1
:NO_NAME
```

在此例中，批处理文件用 ECHO 来输出为拨电话而需要的调制解调器命令。当电话响时，你就可以拿起电话，开始通话。当通话结束时，可以按任何键以回答该批处理文件的 PAUSE 命令，这样批处理文件就会执行为挂起电话而需要的命令。

GOTO 批处理文件命令

功能：

将批处理文件中的处理从一条命令转至用一标号标识的另一处。

格式：

GOTO DOS 批处理标号

分支转移至的位置用下述标号标识：

```
:DOS 批处理标号
```

注释：

缺省时，MS-DOS 处理批处理文件中语句的方式是从第一条命令开始，从顶向下地依次执行各条命令。GOTO 命令使 MS-DOS 跳至批处理文件中的一个新地方(用一标号标识)，并从该标号后面的命令处继续执行。该分支能力可让你建立重复执行命令以及根据某条件而完成不同动作的批处理文件。

标号的特征是在标号名前面有一冒号(:)。只有标号名的前八个字符被用来标识标号。因此，标号:BATCHFILE1 和:BATCHFILE2 是相同的。在 GOTO 命令中，对标号的引用不带冒号。

GOTO 命令一般总是和 IF 命令一起使用，以根据由 IF 语句测试的条件而完成批处理文件中的不同任务。GOTO 和 IF ERRORLEVEL 命令可以和 CHOICE 命令一起使用以根据用户所按的键而选择任务(参见有关 CHOICE 的讨论)。单独使用时，GOTO 可被用来建立循环，并在完成由 IF 语句所设置的任务后将控制转移至批处理文件的主要部分。

例子：

下列批处理文件使用 IF 条件来测试命令行上所说明的值。该值表示用来确定批处理文件的哪一部分被处理的语言。

```
@ECHO OFF
IF "%1"=="GERMAN" GOTO GERMAN
IF "%1"=="FRENCH" GOTO FRENCH
IF "%1"=="SWEDISH" GOTO SWEDISH
GOTO INVALID_LANGUAGE

:GERMAN
ECHO You picked German
GOTO DONE
:FRENCH
ECHO You picked French
GOTO DONE
:SWEDISH
ECHO You picked Swedish
GOTO DONE
:INVALID_LANGUAGE
ECHO That lauguage is not available
:DONE
```

该批处理文件将%1 的值与所接受的选项(GERMAN，FRENCH 或 SWEDISH)进行比较。如果%1 与这些选项中的一个匹配上了，该批处理文件就将控制转移至相应的标号。如果%1 没有匹配上任何选项，则该文件就将转移至 INVALID_LANGUAGE 标号(MS-DOS 会将此标号看成是 INVALID_，所以不要另用以 INVALID_开始的标号)。在完成

了针对具体语言的任务后，另一条 GOTO 分支就转移至文件尾(跳过其他语言段)。

下面的批处理文件重复显示一菜单，并用 CHOICE 命令来从用户获得响应：

```
@ECHO OFF
:LOOP
ECHO Press N for a directory listing sorted by Name.
ECHO Press E for a directory listing sorted by Extension.
ECHO Press Q to Quit.
CHOICE /N /CNEQ Select an option:
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 DIR /ON
IF ERRORLEVEL 2 IF NOT ERRORLEVLE 3 DIR /OE
IF ERRORLEVEL 3 GOTO DONE
GOTO LOOP
:DONE
```

第一组 ECHO 命令显示菜单选择。CHOICE 命令被用来从用户处获得一响应，它被存作 ERRORLEVEL(我们将在对 CHOICE 命令的讨论中进一步解释这一点)。根据 ERRORLEVEL 的值，任务之一被完成。除非用户选择退出菜单，GOTO LOOP 使得菜单再次被显示，而 CHOICE 等待另一次响应。该过程一直继续下去，直到选项被选择(或者用户用 CTRL-C 结束了批处理文件)。

5.4 得到用户的选择

从 MS-DOS 6 开始，在批处理文件中可以用 CHOICE 命令来给出提示并评价键盘响应。缺省时，CHOICE 命令显示一提示并判断对提问的回答是 Y(表示 YES)还是 N(表示 NO)。例如，如果某个批处理文件想要知道是否打印文件 CONFIG. SYS 的内容，那么可以用下列 CHOICE 命令：

```
CHOICE Do you want to print CONFIG.SYS
```

这将显示下列信息：

```
Do you want to print CONFIG.SYS[Y/N]?
```

如果用户按了 Y 或 N 之外的任一键，CHOICE 就从计算机的内部扬声器中发出一蜂鸣声，并继续等待一正确的回答。在此例中，按 Y 返回值为 1 的返回状态值，按 N 返回值 2。在具体的批处理文件中，可以加入下列 IF ERRORLEVEL 命令以在回答为 Y 时打印文件：

```
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 PRINT \CONFIG.SYS
```

如果想让用户从一组不同的键组合中选择，可用 CHOICE 命令的/C 选项，它可让你说明那组键。例如，下列 CHOICE 命令可让用户从 1、2、3 或 4 这四个键中选择：

```
CHOICE /C:1234 Select an option:
```

跟在 CHOICE 的/C 开关后的那组键说明了 CHOICE 将等待用户按下的键。如果任何其他键被按下了，CHOICE 就发出一蜂鸣声，并继续等待所说明的键。当一所说明的键被按下时，CHOICE 就返回与/C 开关后的那组键中被选中的键的位置相对应的返回状态值。在此例中，CHOICE 显示提示信息 Select an option:[1,2,3,4]?，然后在按了 1 时返回退出状态值 1，在按了 2 时返回 2，以此类推。类似地，下列命令显示提示信息 Select an option：[A,B,C,D]?，然后在按了 A 时返回退出状态值 1，按了 B 时返回 2，C 和 D 时返回 3 和 4。

```
CHOICE /C：ABCD Select an option:
```

如果按了 CTRL-C 键组合来取消 CHOICE 命令，CHOICE 返回退出状态值 0。

接着就可以用 IF ERRORLEVEL 命令来确定按下了哪个键。例如，下面的批处理文件提示你按 A、B、C 或 D，然后在屏幕上显示哪一键被按下了：

```
@ECHO OFF
CHOICE /C:ABCD Press one of the following keys:
IF ERRORLEVEL 0 IF NOT ERRORLEVEL 1 ECHO Ctrl-C
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 ECHO You pressed A
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 ECHO You pressed B
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 ECHO You pressed C
IF ERRORLEVEL 4 IF NOT ERRORLEVEL 5 ECHO You pressed D
```

注意：如果在 TERMINATE BATCH JOB(Y/N)? 提示下按 N，就会只看见 CTRL-C 信息。

CHOICE 命令的最常见用处之一是处理一菜单选项。下面的批处理文件 MENU_CH.BAT 显示这样的一菜单，确定哪个键被按下，并运行由用户选择所说明的程序：

```
@ECHO OFF
:LOOP
CLS
ECHO        Main Menu
ECHO.
ECHO  A     Run MSAV (Virus Protection)
ECHO  B     Run DEFRAG (Disk Defragmentation)
ECHO  C     Run MEM (Memory Use Display)
ECHO  Q     Quit to DOS
ECHO.
CHOICE /C:ABCQ Enter choice:
IF NOT ERRORLEVEL 1 GOTO DONE
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 MSAV
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 DEFRAG
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 MEM /DEBUG /PAGE
IF ERRORLEVEL 4 GOTO DONE
PAUSE
GOTO LOOP
:DONE
```

择 N，并返回退出状态值 2。

5.4.3 区分大小写选择

除非另外说明，CHOICE 将大写与小写字母视为等同。例如，下列 CHOICE 命令等待用户输入 A、B、C 或 D：

```
CHOICE /C：ABCD Select an option·
```

在此例中，CHOICE 不管接受的是 A 或 a，都返回同样的退出状态值。

当想让 CHOICE 区分大小写字母时，可以在 CHOICE 命令中加入/S 开关。例如，下面的批处理文件 SORTMENU.BAT 显示一个菜单，该菜单可以让你用六种方式显示目录清单。该六个选项分别对应字母 N、n、E、e、S 和 s：

```
@ECHO OFF
CLS
ECHO        Sort Menu
ECHO.
ECHO  N     Sort files by name A-Z
ECHO  n     Sort files by name Z-A
ECHO  E     Sort files by extension A-Z
ECHO  e     Sort files by extension Z-A
ECHO  S     Sort files by size smallest-largest
ECHO  s     Sort files by size largest-smallest
ECHO.
CHOICE /C:NnEeSs /S /N Sort Option:
IF NOT ERRORLEVEL 1 GOTO DONE
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 DIR /O:N
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 DIR /O:-N
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 DIR /O:E
IF ERRORLEVEL 4 IF NOT ERRORLEVEL 5 DIR /O:-E
IF ERRORLEVEL 5 IF NOT ERRORLEVEL 6 DIR /O:S
IF ERRORLEVEL 6 IF NOT ERRORLEVEL 7 DIR /O:-S
:DONE
```

5.4.4 CHOICE 和 GOTO 的结合

很多时候用户的一个选择可能会导致几条命令，而不是一条命令的执行。例如，下面的批处理文件可让用户确定是处理收到的帐单、付出的帐单作总帐目，还是作月终处理。该批处理文件用 CHOICE 命令来得到用户选择，并用 GOTO 命令来转移至几条相关命令可被执行的特殊地方：

```
@ECHO OFF
:LOOP
CLS
ECHO            Accounting Main Menu
ECHO.
ECHO     A      Accounts Receivable
ECHO     B      Accounts Payable
ECHO     C      General Ledger
ECHO     D      Month End Processing
ECHO     Q      Quit
```

```
ECHO.
CHOICE /C:ABCDQ Enter choice:
IF NOT ERRORLEVEL 1 GOTO DONE
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 GOTO ACCT_REC
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 GOTO ACCT_PAY
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 GOTO GENERAL_LEDGER
IF ERRORLEVEL 4 IF NOT ERRORLEVEL 5 GOTO MONTH_END
IF ERRORLEVEL 5 GOTO DONE
GOTO LOOP

:ACCT_REC
GET_RECV
PRT_RECV
ENV_RECV
GOTO LOOP

:ACCT_PAY
GET_BILL
PRT_BILL
GOTO LOOP

:GENERAL_LEDGER
GEN_LED
PRINT REPORT.GLL
GOTO LOOP

:MONTH_END
COMBINE
PRINT TOTALS.DAT
MERGE MONTH.DAT YEAR.DAT
GOTO LOOP

:DONE
```

在大多数情况下，显示一选项菜单的批处理文件用 CHOICE 来得到用户选择，用 IF ERRORLEVEL 来确定选择，以及用 GOTO 来转移至相应的批处理文件命令。

CHOICE 批处理文件命令

功能：

显示一信息，然后等待用户按一组预定义键中的某一个。CHOICE 接着就返回一退出状态值，利用该值，批处理文件就能确定所按的键。

格式：

CHOICE [/C[:]Keys] [/N][/S][/T[:]Default,Seconds][Prompt]

注释：

当批处理文件执行一 CHOICE 命令时，CHOICE 等待用户按下预定义的一组键中的某一个。如果用户按的键不在被定义之列，CHOICE 就使计算机的内部扬声器发出一蜂

鸣声，并等待正确键的按下。当用户按了正确的键时，CHOICE 就返回一退出状态值，该批处理文件可以用 IF ERRORLIVEL 来测试该值以确定哪一个键被按下了。返回的退出状态值是什么取决于被按下的键以及跟在 CHOICE 命令的/C 开关后的键的次序。CHOICE 对第一个键返回为 1 的退出状态值，对第 2 键返回值 2。以此类推。

/C 开关说明了用户能按下的一组键。如果没有用/C 开关，CHOICE 就认为你在提示用户输入 Y 或 N。

缺省时，CHOICE 显示正确的那组键以及紧跟在用户提示符后的问号。如果用了/N 开关，CHOICE 就既不显示键，也不显示问号。

同样，缺省时 CHOICE 不区分大写字母与小写字母。如果用了/S 开关，CHOICE 就对大小写作出区分。

/T 开关可用来说明缺省键以及从 0 至 99 的秒数，在此时间之后，CHOICE 就自动地选择该缺省键。

注意：CHOICE 是 MS-DOS 的一外部命令，它必须在当前目录或搜索路径中。

例子：

下面的 CHOICE 命令提示用户说明 MS-DOS 是否要打印 AUTOEXEC.BAT 文件：

```
CHOICE Do you want to print AUTOEXEC.BAT
```

在此例中，由于 CHOICE 命令行没有/C 开关，CHOICE 就提示用户回答 Y 或 N。

下一条 CHOICE 命令显示提示 SELECT AN OPTION：然后等待用户按 A、B、C 或 Q 键。

```
CHOICE /C:ABCQ Select an option:
```

最后一条 CHOICE 命令提示用户说明 MS-DOS 是否要执行 MSAV 反病毒实用工具。如果用户不在 15 秒内响应，CHOICE 就选择 N 作为缺省响应：

```
CHOICE /T:N,15 Do you want to run MSAV
```

5.5 对一组文件重复执行某条 MS-DOS 命令

在本章前面讲到了 IF 命令可让批处理文件只在满足一特定条件时才执行某条 MS-DOS 命令。根据某一条件是真或假而进行的处理被称为条件处理。另一条批处理文件命令 FOR 可让批处理文件对一组文件重复执行某条 MS-DOS 命令。执行至少一次，也可能多次的处理被称为循环处理。和大多数批处理文件所特有的命令(CHOICE 除外)一样，FOR 也是内部命令。MS-DOS 始终将内部命令保存在内存中(而诸如 DISKCOPY 这样的外部命令一直在磁盘上等待被使用)。

FOR 命令的格式如下：

```
FOR %%BatchVar IN (SetOfFiles) DO DOS命令
```

其中 BatchVar 是批处理文件中的变量，SetOfFiles 是一组文件。

命令名 FOR 后跟%%BatchVar 变量。以前我们讨论过%0 至%9 变量，MS-DOS 在批处理文件执行期间将命令行参数值赋给这些变量。MS-DOS 另将一值赋给%%BatchVar 变量。差别在于%0 至%9 变量的含意是预定义了的，而任何单个字母（如%%A，%%F）都可作为%%BatchVar 变量。MS-DOS 将批处理文件中的变量名限制为一个字符。

%%BatchVar 变量后跟 IN，IN 告诉 MS-DOS，FOR 命令将要使用的那组值紧跟在后面，且用括号括起。例如，为说明一组文件，只需键入文件名，且彼此之间用空格或逗号隔开。下面是 FOR 命令可用的正确的文件组：

```
(MAY.PAY JUNE.PAY JULY.PAY)
(MAY.PAY,JUNE.PAY,JULY.PAY)
(*.BAT *.EXE *.COM)
(*.*)
```

如你所见，FOR 支持星号 * 通配符以及问号?。当 FOR 发现一通配符，MS-DOS 就将它扩展成相应的多个文件名。

MS-DOS 在执行 FOR 命令时，将那组文件中的每一文件名都赋给%%BatchVar 变量。在第一个文件名被赋值后，MS-DOS 就执行在命令行中跟在 DO 后面的命令。该命令可以是任何 MS-DOS 命令。当执行完此命令时，MS-DOS 就将第二个文件名赋给%%BatchVar 变量，并重复此过程。当所有文件都经过这样的处理后，FOR 命令就执行完了。

为更好地理解此过程，考虑下列命令：

```
FOR %%A IN (A.BAT B.BAT C.BAT) DO TYPE %%A
```

这里，FOR 首先将 A.BAT 文件赋给%%A 变量，然后 MS-DOS 执行 TYPE %%A 命令。由于 MS-DOS 已经将 A.BAT 文件名赋给了%%BatchVar 变量——在此例中是%%A，所以 MS-DOS 实际上就执行 TYPE A.BAT 命令。

当执行完 TYPE 命令后，FOR 就将下一文件名赋给%%A 变量，并重复 TYPE 命令。在此例中，MS-DOS 显示文件 B.BAT 的内容。FOR 然后再次重复此过程，将 C.BAT 文件赋给该变量。最后，当 TYPE 命令执行完后，FOR 发现不再有文件了，所以也就完成了其执行过程。

5.5.1 显示目录清单

下面的批处理文件 SHOW.BAT 以一种类似的方式来使用 FOR 命令以显示当前目录中的所有批处理文件：

```
FOR %%I IN (*.BAT) DO TYPE %%I
```

该 FOR 命令用%%I 作为%%BatchVar 变量。实际的变量名只要是单个字母，是什么字符都无所谓。

下面是另一个例子。缺省时，DIR 命令显示文件名、扩展名、大小及创建日期和时间：

```
TIMEDATE   BAT        23 04-07 94   1:23a
DISKINFO   BAT        18 04-07-94   11:13a
SHORTDIR   BAT        21 04-07-94   11:15a
VERVOL2    BAT        12 04-08-94   1:23p
DIRAB      BAT        30 04-08-94   1:37p
DIRAB      OUT         0 04-08-94   1:38p
PRINTDIR   BAT        41 04-09 94   2:29p
  .          .          .     .       .
  .          .          .     .       .
  .          .          .     .       .
ERRLEVEL   BAT        73 04-15-94   5:58p
```

如果用带/W 开关的 DIR，MS-DOS 就只在屏幕上以五列显示文件名和扩展名。有时可能想要一个接一个地显示文件名：

```
TIMEDATE. BAT
DISKINFO. BAT
SHORTDIR. BAT
VERVOL2. BAT
DIRAB. BAT
DIRAB. OUT
PRINTDIR. BAT
  .
  .
  .
ERRLEVEL BAT
```

下面的批处理文件 SHORTDIR. BAT 就用了带 ECHO 命令的 FOR 命令来显示按上述形式的文件名：

```
@ECHO OFF
FOR %%I IN (*. *) DO ECHO %%I
```

FOR 命令将当前目录中的每个文件的名字都赋给%%I 变量。MS-DOS 然后就用 ECHO命令来在屏幕上显示文件名(当然，在 MS-DOS 5. 0和6. 0版中，也可以用 DIR /B 来完成此任务)。

通过使用批处理文件参数，就可提高 SHORTDIR. BAT 的灵活性，以将想要显示的文件名说明成第一个命令行参数。然后就可以用下列命令来显示当前目录中带 BAT 扩展名的所有文件。

```
C:\>SHORTDIR *. BAT
```

为显示带 TXT 扩展名的文件，可以敲入：

```
C:\>SHORTDIR *. TXT
```

为显示当前目录中的所有文件，命令行是：

```
C:\>SHORTDIR *.*
```

为提供此灵活性，可对 SHORTDIR.BAT 修改如下：

```
@ECHO OFF
FOR %%1 IN (%1) DO ECHO %%I
```

5.5.2 对显示进行分类

下一个批处理文件 SORTDIR.BAT 和 FOR 命令一起使用了 MS-DOS 重定向命令，来对由第一个命令行参数所说明的文件目录进行分类显示：

```
@ECHO OFF
IF EXIST SORTFILE.DAT DEL SORTFILE.DAT
FOR %%1 IN (%1) DO ECHO %%I >> SORTFILE.DAT
SORT < SORTFILE.DAT
DEL SORTFILE.DAT
```

该批处理文件首先用 IF EXIST 命令来在 SORTFILE.DAT 文件存在的情况下删除它。然后 FOR 命令用 ECHO 命令显示每个文件名(如同上一个批处理文件那样)。但在此例中，批处理文件用了>>重定向符来将 ECHO 命令从屏幕上的输出添加至 SORTFILE.DAT 文件中。在 FOR 建立了由文件名组成的文件后，该批处理文件就用 SORT 命令对该文件的内容进行分类，并在屏幕上显示分好类的文件名。当 SORT 命令执行完后，就删除掉不再需要的 SORTFILE.DAT 文件。在 MS-DOS 5.0及以后版本中，DIR /B /O:N 可达到同样目的。

5.5.3 建立灵活的 TYPE 命令

使用 FOR 命令可以改进在本章开始建立的 T.BAT 批处理文件，以使它支持 MS-DOS 通配符。修改如下：

```
@ECHO OFF
FOR %%I IN (%1) DO TYPE %%I
```

用下面的命令就可显示在当前目录中所有批处理文件的内容：

```
C:\>T *.BAT
```

事实上，你可能想通过加上几个批处理文件参数来将特定文件作为 FOR 的文件组：

```
@ECHO OFF
FOR %%I IN (%1 %2 %3 %4 %5) DO TYPE %%I
```

然后就可以说明显示哪些文件：

```
C:\>T SORTDIR.BAT SHORTDIR.BAT *.BAT
```

该批处理文件首先显示 SORTDIR.BAT 的内容，然后是 SHORTDIR.BAT 的内容。接着，该批处理文件显示当前盘和目录中所有带 BAT 扩展名的文件内容。

FOR 命令使批处理文件的灵活性得到大大提高，在本书后面我们还将常常用到它。

FOR 批处理文件命令

功能：

对一组给定的文件重复一 MS-DOS 命令。

格式：

FOR %%BatchVar IN（一组文件名）DO DOS 命令

注释：

当运行 FOR 命令时，FOR 把在那组文件中说明的第一个文件赋给所说明的变量 BatchVar。该变量名必须是单个字符。对那组文件名的说明可以敲入用空格或逗号隔开的文件名，也可以用 * 号和？号这些通配字符。FOR 命令完全支持 MS-DOS 的通配字符。

如果文件集是用通配符说明的，FOR 就扩展每个通配符以建立文件名集，然后再将每个文件名赋给变量，一次一个。

在将文件名赋给变量后，FOR 就执行所说明的 MS-DOS 命令。当 MS-DOS 命令执行完后，FOR 就将下一个文件名赋给此变量，并重复此过程。

当不再有新的文件名时，FOR 命令就结束了，批处理文件也就从下一条命令开始继续处理。

例子：

下面的批处理文件用 FOR 命令来使用 TYPE 命令并显示文件 A.TXT，B.TXT 和 C.TXT 的内容：

```
FOR %%I IN (A.TXT B.TXT C.TXT) DO TYPE %%I
```

下列的文件以类似的方式使用 TYPE 命令来显示所有与%1参数相应的文件内容：

```
FOR %%V IN (%1) DO TYPE %%V
```

如果用一通配符运行此批处理文件，如 *.BAT 中的 * 号，那么该批处理文件就显示每个相应文件的内容。

第 6 章

批处理文件的高级概念

6.1 使用有名参数

前面我们已经看到，从%0至%9的参数可让单个批处理文件用于若干目的，从而提高了批处理文件的能力。除了这些批处理文件参数外，MS-DOS 3.3或以后版本还支持所谓的有名参数，它们是在两端用百分号标识的批处理文件变量。当 MS-DOS 在批处理文件中遇到一有名参数时，就在其环境中寻找一相应的项。例如，批处理文件 SHOWFILE.BAT 显示了对应于有名参数%SHOW_FILE%的文件：

```
TYPE %SHOW_FILE% | MORE
```

当 MS-DOS 遇到%SHOW_FILE%时，就在其环境中以下列格式寻找一项：

```
SHOW_FILE=
```

SET 命令可用于改变或显示 MS-DOS 环境中的项。例如，可以不带命令行地使用 SET 命令：

```
C:\>SET
```

MS-DOS 然后就显示当前的环境项：

```
COMSPEC=C:\DOS\COMMAND.COM
PATH=C:\DOS
PROMPT=$P$G
```

用 SET 命令可以创建一环境项并对之进行赋值：

```
C:\>SET SHOW_FILE=\AUTOEXEC.BAT
```

然后就能使用 SET 命令来显示新的环境项：

```
C:\>SET
COMSPEC=C:\DOS\COMMAND.COM
PATH=C:\DOS
PROMPT=$P$G
SHOW_FILE=\AUTOEXEC.BAT
```

当运行 SHOWFILE.BAT 时，MS-DOS 就用相应的环境项——在此例中是 AUTOEXEC.BAT ——来代替有名参数 SHOW_FILE，并显示该文件的内容。

为删除一环境项，可以使用如下的 SET 命令：

```
C:\>SET SHOW_FILE=
```

一定要在最后加上等于号。在此例中，MS-DOS 删去了 SHOW_FILE 项，环境的内容现在就变成了：

```
C:\>SET
COMSPEC=C:\DOS\COMMAND.COM
PATH=C:\DOS
PROMPT=$P$G
```

如果再次运行 SHOWFILE.BAT 文件，MS-DOS 就找不到能匹配 SHOW_FILE 的环境项，因为该有名参数的内容是空串。通过使用 IF 命令，就可以让批处理文件对空串作测试，然后进行相应的处理：

```
@ECHO OFF
IF "%SHOW_FILE%"=="" GOTO NO_PARAMETER
TYPE %SHOW_FILE% | MORE
GOTO DONE
:NO_PARAMETER
ECHO The SHOW_FILE named parameter is
ECHO not defined. Use the SET command
ECHO to assign a file name to SHOW_FILE.
:DONE
```

许多人都对需要有名参数的批处理文件感到不太理解，所以我们再来看两个例子。第一，假定有一个会在其执行期间常常建立临时文件的批处理文件，而你又不想让 MS-DOS 在当前目录中建立这些文件。例如，下面的批处理文件建立了一个名为 SORTDIR.TMP 的临时文件：

```
DIR | SORT > SORTDIR.TMP
PRINT SORTDIR.TMP
MORE < SORTDIR.TMP
DEL SORTDIR.TMP
```

如果运行该批处理文件，并在 DEL 命令执行前用 Ctrl-C 结束该文件，那么 SORTDIR.TMP 文件就仍在当前目录中。另一种方法是建立一个 TEMPDIR 目录来保存临时文件，然后在整个批处理文件中引用该目录：

```
DIR | SORT > \TEMPDIR\SORTDIR.TMP
PRINT \TEMPDIR\SORTDIR.TMP
MORE < \TEMPDIR\SORTDIR.TMP
DEL \TEMPDIR\SORTDIR.TMP
```

该方法的问题是它使得所有用户都建立一 TEMPDIR 目录来保存临时文件。

下面的批处理文件 NAMEDTMP.BAT 通过使用一有名参数而解决在哪里存放临

时文件的问题：

```
DIR ¦ SORT > %TEMPDIR%SORTDIR.TMP
PRINT %TEMPDIR%SORTDIR.TMP
MORE < %TEMPDIR%SORTDIR.TMP
DEL %TEMPDIR%SORTDIR.TMP
```

当MS-DOS遇到%TEMPDIR%有名参数时，就在其环境中寻找与之相应的项。如果不存在这样的项，MS-DOS就将一空串赋给%TEMPDIR%，并在当前目录中创建SORTDIR.TMP文件。如果在运行NAMEDTMP.BAT之前创建了下面这样的环境项，MS-DOS就用该有名参数在D盘的当前目录中建立SORTDIR.TMP文件：

```
C:\>SET TEMPDIR=D:
```

为说明D盘上一名为DATA的目录，可以这样：

```
C:\>SET TEMPDIR=D:\DATA\
```

该批处理文件能满足使用它的所有潜在用户的需要。不在乎是否让MS-DOS在当前目录中创建文件的用户可以不用SET命令来定义TEMPDIR就运行此批处理文件；而想让MS-DOS在一说明的目录中（如快速虚拟盘上的一目录）创建该文件的用户只需用SET来为有名参数设一合适的驱动器、子目录或驱动器加子目录。

如先前讨论过的那样，PATH=项定义了MS-DOS据此来搜索外部命令的一组子目录。大多数人都是在AUTOEXEC.BAT中用PATH命令来定义PATH=的。当安装一应用软件时，其安装程序常常将该软件新建立的目录加至AUTOEXEC.BAT的PATH命令中。假定AUTOEXEC.BAT文件含下列命令：

```
@ECHO OFF
PRINT /D:LPT1 /Q:32
PATH C:\DOS
SET PROMPT=$P$G
```

接着假定安装程序想将C:\UTIL目录加到PATH命令中。安装程序不会这样修改PATH：

```
PATH C:\DOS
```

而是将新的目录项加至末尾：

```
SET PATH=%PATH%;C:\UTIL
```

这里利用PATH命令建立了一个MS-DOS环境项，而你可将该项作为有名参数。

为更好地理解该过程是怎样工作的，可建立含下列命令的名为MYPATH.BAT的批处理文件：

```
@ECHO OFF
SET MYPATH=C:\DOS
```

```
ECHO Original entry is %MYPATH%
ECHO Appending C:\UTIL to %MYPATH%
SET MYPATH=%MYPATH%;C:\UTIL
ECHO Complete entry is %MYPATH%
```

当运行该批处理文件时，MS-DOS 显示下列内容：

```
Original entry is C:\DOS
Appending C:\UTIL to C:\DOS
Complete entry is C:\DOS;C:\UTIL
```

可见，该批处理文件的确建立了项 C:\DOS;C:\UTIL。

在本书后面我们将考察的几个批处理文件还将使用 MS-DOS 的有名参数。

6.2 在一个批处理文件中运行另一个批处理文件

迄今为止，我们所建立的批处理文件所执行的程序都是 EXE 或 COM 文件。如果希望一个批处理文件运行另一个批处理文件（然后在第二个批处理文件执行完后继续执行第一个批处理文件），那么就需要使用 CALL 命令或 COMMAND /C 命令，具体情况视 MS-DOS 的版本而定。

为理解为什么一个批处理文件不能像 MS-DOS 命令或程序那样只通过调用第二个批处理文件的名而执行这第二个文件，让我们来建立一个含下列命令的名为 VERVOL.BAT 的批处理文件：

```
VER
VOL
```

然后再建立一个含下面三条命令的批处理文件 PRIMARY.BAT：

```
DATE
VERVOL
TIME
```

PRIMARY.BAT 通过引用批处理文件 VERVOL.BAT 的名字而运行它。当运行 PRIMARY.BAT 时，MS-DOS 首先执行 DATE 命令，并显示下列内容：

```
C:\>PRIMARY

C:\>DATE
Current date is Thu 04-07-1994
Enter new date (mm-dd-yy):
```

如果按了 ENTER 键（即不修改日期），MS-DOS 就运行第二个批处理文件 VERVOL.BAT。当 MS-DOS 执行此文件时，MS-DOS 显示：

```
C:\>PRIMARY
```

```
C:\>DATE
Current date is Thu 04-07-1994
Enter new date (mm-dd-yy):

C:\>VERVOL

C:\>VER

MS-DOS version 6.20

C:\>VOL

  Volume in drive C is DOS 6 DISK
  Volume Serial Number is 3921-18D3
C:\>
```

当 VERVOL.BAT 批处理文件执行完后，MS-DOS 不会执行 PRIMARY.BAT 中的最后一条 TIME 命令，为在 VERVOL.BAT 执行完后返回至 PRIMARY.BAT，必须使用 CALL 或 COMMAND /C 命令来运行第二个批处理文件。

如果用的是 MS-DOS 3.3或以后的版本，那么就可以使用 CALL 命令来执行第二个批处理文件中的命令，然后再返回至第一个批处理文件中的下一条命令。CALL 命令的格式如下：

```
CALL 批处理文件 [参数]
```

参数是 MS-DOS 赋给%1至%9变量的可选值。从 MS-DOS 3.3版开始，可以将 PRIMARY.BAT 改成：

```
DATE
CALL VERVOL
TIME
```

当运行此批处理文件时，MS-DOS 再次执行 DATE 命令。同前所述，按 ENTER 而不修改系统日期，然后 MS-DOS 执行 CALL 命令，该命令又运行 VERVOL.BAT 批处理文件。当 VERVOL.BAT 执行完后，MS-DOS 就执行 PRIMARY.BAT 中的下一条命令 TIME。因此，MS-DOS 显示如下内容：

```
C:\>PRIMARY

C:\>DATE
Current date is Thu 04-07-1994
Enter new date (mm-dd-yy):
```

```
C:\>CALL VERVOL

C:\>VER

MS-DOS version 6.2

C:\>VOL

  Volume in drive C is DOS 6 DISK
  Volume Serial Number is 3921-18D3

C:\>TIME
Current time is 1:54:09.09p
Enter new time:

C:\>
```

由此可见，CALL 命令可让一个批处理文件成功地运行第二个批处理文件，然后再返回第一个文件以执行剩余的命令。

如果用的是 MS-DOS 3.3以前的版本，就必须使用 COMMAND /C 来运行第二个批处理文件。在3.3以后的版本中也可用此命令。COMMAND.COM 文件含 MS-DOS 命令行处理器，该处理器既可执行在命令提示符下被敲入的命令，又可执行批处理文件中的命令。在 MS-DOS 3.3版之前，可修改 PRIMARY.BAT 文件以使其内容变为：

```
DATE
COMMAND /C VERVOL
TIME
```

COMMAND.COM 文件必须在当前目录或由 PATH 所说明的一目录中。

当运行 PRIMARY.BAT 时，MS-DOS 执行 DATE 命令。同样，按 ENTER 键以不修改系统日期。该批处理文件继续处理，用 COMMAND /C 来运行 VERVOL.BAT 批处理文件。当 VERVOL.BAT 执行完后，MS-DOS 就执行在 PRIMARY.BAT 文件中的 TIME 命令。COMMAND /C 可让 MS-DOS 将第二个命令处理器装入内存。/C 告诉 MS-DOS 只在后面的命令尚未被执行完时才将命令处理器放在内存。这样，第二个命令处理器在内存中的时间就正好是 VERVOL.BAT 文件中的命令被执行的时间。当该批处理文件执行完后，MS-DOS 就将第二个命令处理器从内存中删除，并在 PRIMARY.BAT 批处理文件中继续执行其余的命令。

CALL 批处理文件命令

功能：

允许一个批处理文件执行另一个批处理文件，然后再返回至原先的批处理文件。

格式：

CALL 批处理文件 [参数]

注释：

如果不用 CALL 或 COMMAND /C 命令而从一个批处理文件中运行另一个批处理文件，那么 MS-DOS 就只执行到第二个批处理文件执行完时执行的命令。如果运行的第二个批处理文件在第一个批处理文件的中间，那么第一个批处理文件中跟在调用第二个批处理文件的命令后面的其他命令就永远不会被执行。当第二个批处理文件结束时，所有批处理文件命令的执行也就结束了。

如果用的是 MS-DOS 3.3或以后的版本，就可以用 CALL 命令来从一个批处理文件中运行另一个批处理文件，然后再返回至原先的批处理文件。方法只是将第二个批处理文件的名字和任何批处理文件的参数放在 CALL 命令行上。

例子：

下面的批处理文件使用 CALL 命令来运行名为 NESTED.BAT 的第二个批处理文件：

```
VER
CALL NESTED
VOL
```

在此例中，NESTED.BAT 含这些命令：

```
DATE
TIME
```

如果将 CALL 命令从第一个批处理文件中移去就变成：

```
VER
NESTED
VOL
```

MS-DOS 将永远也不会执行 VOL 命令。当 NESTED.BAT 执行完后，MS-DOS 就不再执行批处理文件的命令，而是返回至命令提示符。

COMMAND /C 批处理文件命令

功能：

允许一个批处理文件执行另一个批处理文件，然后再返回至原先的批处理文件。

格式：

COMMAND /C 批处理文件［参数］

注释：

如果不用 CALL 或 COMMAND /C 命令而从一个批处理文件中运行另一个批处理文件，那么 MS-DOS 就只执行到第二个批处理文件执行完时执行的命令。如果运行的第二个批处理文件在第一个批处理文件的中间，那么第一个批处理文件中跟在调用第二个批处理文件的命令后面的其他命令就永远不会被执行。当第二个批处理文件结束时，所有批处理文件命令的执行也就结束了。

如果用的是 MS-DOS 3.2或以前的版本，就可以用 COMMAND /C 命令来从一个批处理文件中运行另一个批处理文件，然后再返回至原先的批处理文件。方法只是将第二个批处理文件的名字和任何批处理文件的参数放在 COMMAND /C 命令行上。

例子：

下面的批处理文件使用 COMMAND /C 命令来运行名为 NESTED.BAT 的第二个批处理文件：

```
VER
COMMAND /C NESTED
VOL
```

在此例中，NESTED.BAT 含这些命令：

```
DATE
TIME
```

如果将 COMMAND /C 命令从第一个批处理文件中移去就变成：

```
VER
NESTED
VOL
```

注意：在 MS-DOS 各种版本中都能不带参数地使用 COMMAND.COM 命令而装入第二个命令处理器。必须用 EXIT 命令来返回至批处理文件的处理。

6.3 使用第二个命令处理器

随着批处理文件复杂性的增加，可能会希望显示一选项菜单，每一选项都引起执行一不同的 MS-DOS 命令。其中的一个选项可以暂时停止批处理文件的执行，从而用户可以从命令提示符下执行命令。MS-DOS 的 COMMAND 命令恰好可让批处理文件具备此功能。

批处理文件 USEDOS.BAT 含下面这些命令：

```
@ECHO OFF
ECHO In batch file, about to access command prompt
COMMAND
ECHO Back in batch file
```

当 MS-DOS 遇到 COMMAND 时，就将第二个命令处理器装入内存，并显示一个命令提示符以使你可以执行 MS-DOS 命令。在第二个命令处理器中对环境项所作的任何修改都只在退出此命令处理器之前才有效。当不再想执行 MS-DOS 命令时，可用 EXIT 命令来退出第二个命令处理器。

如果 COMMAND.COM 文件在当前目录中，或在 PATH 说明的目录中，那么当运行 USEDOS.BAT 批处理文件时，MS-DOS 显示如下：

```
C:\>USEDOS
In batch file, about to access command prompt

MICROSOFT® MS-DOS® Version 6.2
                ©Copyright Microsoft Corp. 1981-1993

C:\>
```

MS-DOS 已装入第二个命令处理器，从而可以在命令提示符下执行命令。在这里，我们可执行 DATE 命令以显示当前系统日期。为将控制返回至批处理文件，可执行 EXIT 命令。屏幕上的显示大致如下：

```
C:\>USEDOS
In batch file, about to access command prompt

MICROSOFT® MS-DOS® Version 6.2
                ©Copyright Microsoft Corp. 1981-1993

C:\>DATE
Current date is Thu 04-07-1994
Enter new date (mm-dd-yy):
```

```
C:\>EXIT
Back in batch file

C:\>
```

当 EXIT 命令执行完后，MS-DOS 就将控制返回至批处理文件以使该文件得以继续执行。

6.4 用 SHIFT 处理多于九个的参数

前面说过，%1至%9的批处理文件参数可允许将九个值传给批处理文件。以这种方式将值传给批处理文件增加了批处理文件的灵活性。比起不支持参数的批处理文件来说，这还使得带参数的批处理文件能支持多得多的应用程序。尽管如此，九个参数的限制在某些时候也会带来问题。这时，可用 SHIFT 命令来解决此问题。

SHIFT 命令的主要用途是让批处理文件可访问多于九个的批处理文件参数。当 MS-DOS 在一批处理文件中遇到 SHIFT 时，就将每个批处理参数值左移一位，即将%1的值赋给%0参数，%2的值赋给%1，等等。如果命令行含多于九个的参数，SHIFT 就会将第十个参数赋给%9；否则，SHIFT 就将空串赋给%9。例如，批处理文件 ONESHIFT.BAT 的内容如下：

```
@ECHO OFF
CLS
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
SHIFT
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
```

可以如下这样运行该文件：

```
C:\>ONESHIFT A B C
```

MS-DOS 将清屏并显示：

```
ONESHIFT A B C
A B C
```

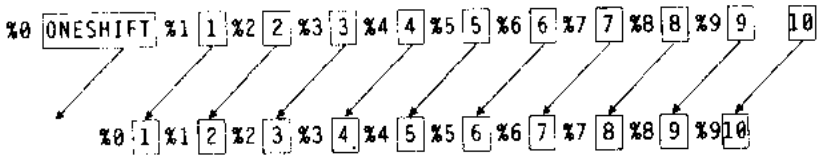
此外，也可以在带多于九个参数的情况下运行此批处理文件：

```
C:\>ONESHIFT 1 2 3 4 5 6 7 8 9 10 11
```

MS-DOS 会显示：

```
ONESHIFT 1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9 10
```

注意 SHIFT 将值10赋给%9参数。下面是 SHIFT 操作的图示：



下一个批处理文件 TWOSHIFT.BAT 使用了两次 SHIFT 命令：

```
@ECHO OFF
CLS
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
SHIFT
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
SHIFT
ECHO %0 %1 %2 %3 %4 %5 %6 %7 %8 %9
```

如果像下面这样运行此文件：

```
C:\>TWOSHIFT A B C
```

那么，MS-DOS 就先清屏，然后显示：

```
TWOSHIFT A B C
A B C
B C
```

同前面一样，可以用多于九个参数来运行此文件：

```
C:\>TWOSHIFT 1 2 3 4 5 6 7 8 9 10 11
```

MS-DOS 先清屏，然后显示：

```
TWOSHIFT 1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9 10
2 3 4 5 6 7 8 9 10 11
```

记住，当 SHIFT 不再有值赋给%9时，就将空串赋给%9。通过测试是否空串，可建立批处理文件 SHIFT_IT.BAT，它用 GOTO 命令循环处理所有的命令行参数，并用 ECHO 命令每行一个地显示各个参数：

```
@ECHO OFF
:REPEAT
ECHO %9
SHIFT
IF NOT "%0"=="" GOTO REPEAT
```

可以如下运行此文件：

```
C:\>SHIFT_IT ONE TWO THREE
```

MS-DOS 显示如下：

```
SHIFT_IT
ONE
TWO
THREE
```

当该批处理文件开始执行时，%0是它的名字。当它执行到 SHIFT 命令时，MS-DOS 将 ONE 赋给%0。由于%0不等于空串，该批处理文件就执行 GOTO 命令，重复此过程。在第四次执行 SHIFT 命令后，%0为空串了，从而就结束了此文件的执行。

用同样的方法，可以对 SHIFT_IT.BAT 稍加修改以建立一个可放进 DIR 命令的批处理文件 D.BAT：

```
@ECHO OFF
:REPEAT
DIR %1
SHIFT
IF NOT "%1"==="" GOTO REPEAT
```

现在可以用下列命令行运行此批处理文件：

```
C:\>D \*.BAT\BAT \UTILS\*.BAT
```

此批处理文件对所说明的文件和目录显示连续的目录清单。在此例中，显示的是根目录中带 BAT 扩展名的所有文件(*.BAT)。然后是 BAT 目录中的所有文件，最后是\UTILS 目录中的所有 BAT 文件。

可以用

```
FOR %%A IN (%1) DO TYPE %%A
```

代替

```
DIR %1
```

然后该批处理文件就使用 TYPE 命令来显示在命令行中说明的每个文件。

SHIFT 批处理命令

功能：

SHIFT 将两个批处理参数值左移一个参数，MS-DOS 将%1的值赋给%0参数，%2的值赋给%1，等等。

格式：

SHIFT

注释：

MS-DOS 支持批处理文件参数%0到%9，如果命令行参数多于九个，就可以用 SHIFT 命令存取多出的参数。例如，可以如下运行批处理文件 SHIFTTST.BAT：

```
C:\SHIFTTST A B C D
```

批处理文件参数值就如下面所示：

%0值为 SHIFTTST
%1值为 A
%2值为 B
%3值为 C
%4值为 D
%5值为""(空串)

如果批处理文件包含一条 SHTFT 命令，MS-DOS 将移动多个参数，结果如下：

%0值为 A
%1值为 B
%2值为 C
%3值为 D
%4值为""(空串)
%5值为""(空串)

如果命令行参数多于九个，SHIFT 就将第十个参数赋给%9，在下一次迭代时将第十一个参数赋给%9，以此类推。当没有参数时，SHIFT 将空串赋给%9。

例子：

下面的批处理文件用 SHIFT 显示在命令行上输入的所有批处理文件参数：

```
:REPEAT
IF "%1"==="" GOTO NO_MORE
ECHO %1
SHIFT
GOTO REPEAT
:NO_MORE
```

该批处理文件只是简单地循环，显示%1的当前值。只要%1不为空串，就一直循环，并且一个新的参数值被移到%1中。

6.5 有关批处理文件的几个问题

本节讨论在运行批处理文件时可能会遇到的几个问题。应当仔细阅读下面的各小节，这会在你试图找出一批处理文件工作不正常的原因时节省你大量的时间和精力。

6.5.1 Bad command or file name 错误信息

如果一个批处理文件显示了下列信息，则该文件中就有一条命令是不正确的：

```
Bad command or file name
```

确定哪条命令导致该信息的最快方法是删去 ECHO OFF，这样在执行批处理文件时，MS-DOS 就会显示每一命令名，在屏幕上错误信息前面的就是不正确的命令。例如，下面的批处理文件 BADCMD.BAT 就含一错误命令：

```
VER
XYZ12345
VOL
```

在运行此文件时，MS-DOS 显示：

```
C:\>VER
MS-DOS version 6.2
C:\>XYZ12345
Bad command or file name

C:\>VOL

 Volume in drive C is DOS 6 DISK
 Volume Serial Number is 3921-18D3

C:\>
```

注意 MS-DOS 在 XYZ12345命令后立即显示错误信息。

当一批处理文件含一非法命令时，MS-DOS 就显示错误信息，并继续执行下一条命令。有时命令虽然是一正确的 MS-DOS 命令但 MS-DOS 在遇到该命令时仍会显示出 Bad command or file name 错误信息。这时，问题就不是出在命令名上，而是出在 MS-DOS 在硬盘上搜索此命令的路径上。

记住：MS-DOS 的命令有内部与外部之分。内部命令（如 TYPE、DATE、CLS 和 VER）在 MS-DOS 启动后始终驻留在计算机内存中。外部命令（如 CHOICE 和 FORMAT）则在磁盘上。为执行一外部命令，MS-DOS 必须先在磁盘上找到该命令，然后将其装入内存中，如果该命令不在当前目录、所说明的目录或搜索路径中，那么 MS-DOS 就会因为找不到该文件而显示 Bad command or file name 错误信息。如前所述，PATH 命令说明了 MS-DOS 在何处对命令进行搜索。如果你知道引起信息 Bad command or file name 的命令是正确的，那么可检查该命令的目录是否在搜索路径中，从而让 MS-DOS 能找到该命令。

注意：不可以从一批处理文件中激活一 DOSKEY 宏（我们下一章将讨论）。否则，MS-

DOS 也会显示 BAD command or file name。

如果你知道批处理文件中的所有命令都是正确的并能被成功装入，但仍然出现了 Bad command or file name 信息，那么你的字处理器可能含有一个 MS-DOS 在此文件中不能理解的怪字符。如果用某个字处理器来建立自己的批处理文件，那么一定要将文件用 ASCII 正文格式存放。

6.5.2 File not found 错误信息

如果批处理文件显示了下列信息，那么文件中的命令是正确的，但 MS-DOS 找不到其中某条命令想要使用的文件：

```
File not found
```

例如，下面的批处理文件 BADFILE.BAT 含这些命令：

```
TYPE 12345678.XYZ
```

当运行此批处理文件时，MS-DOS 显示：

```
File not found-12345678.XYZ

C:\>
```

虽然 MS DOS 成功地执行了 TYPE 命令，但 TYPE 却找不到文件12345678.XYZ。通过使用 IF EXIST 命令，批处理文件就能进行测试以保证在试图使用某文件时该文件存在。

注意：MS-DOS 对命令而不是对文件搜索 PATH 命令中说明的目录。所以或者该文件必须在当前目录，或者必须说明该文件的完整路径。

6.5.3 标号问题

前面说过，GOTO 命令可让批处理文件从文件的一处转移至另一处。当 MS-DOS 遇到一 GOTO 命令时，就在该批处理文件中寻找所说明的标号。如果 MS-DOS 遇到了该标号，就从该标号后面的命令处开始继续执行。如果该标号不存在，MS-DOS 就停止对批处理文件的执行，并显示下列信息：

```
Label not found
```

如果出现该错误，就要检查你的批处理文件，以保证它含正确拼写的该标号，并且要在它前面放上一冒号（记住在 GOTO 命令引用标号时不应使用冒号）。下面的批处理文件 NOLABEL.BAT 说明了 MS-DOS 在运行一批处理文件时，遇到一不正确的标号后是如何立即终止批处理文件的执行的：

```
GOTO DONE
VER
```

如，下面的命令测试 IO_TEST.DAT 文件是否存在，并在其不存在时，将 DIR 命令的输出重定向至 IO_TEST.DAT 文件中：

```
IF NOT EXIST IO_TEST.DAT DIR > IO_TEST.DAT
```

此命令很直观。你可能会认为如果 IO_TEST.DAT 文件不存在，那么该批处理文件就会创建它，将当前目录中的文件清单赋给它，否则，批处理文件就什么也不做。遗憾的是，MS-DOS 实际上并不如此运行。当 IF 命令中说明的命令用了至一文件的重定向时，MS-DOS 就首先打开该文件以准备后面的输出。接着，MS-DOS 进行 IF 命令测试——在此例中是测试 IO_TEST.DAT 文件是否存在。由于 MS-DOS 刚刚创建此文件，所以它确实存在，这样 IF NOT EXIST 命令始终会失败。于是在该命令执行后，IO_TEST.DAT 文件存在但内容为空。

为解决此问题，可取而代之用 IF…GOTO 语句来测试文件是否存在，并在不存在时，转移至正确的地方：

```
IF NOT EXIST IO_TEST.DAT GOTO NO_FILE
REM Other commands here
:NO_FILE
DIR > IO_TEST.DAT
```

第 7 章

DOSKEY 的使用及 DOSKEY 宏的定义

7.1 将命令再现到命令行上

我们已经知道，MS-DOS 批处理文件能够节省时间以及减少击键次数。从 MS-DOS 5.0 版开始，还可以使用功能很强的 DOSKEY 命令以简化具体任务。在执行 DOSKEY 后，MS-DOS 就建立一个表，用以存放从命令行上执行的每条命令的名字。使用键盘上的箭头键就可以很快遍历该表，然后对先前键入的命令重复执行或略作修改后加以执行。

为更好地了解此过程，先像下面这样激活 DOSKEY：

```
C:\>DOSKEY
DOSKEY installed
```

如果没有看见 DOSKEY installed 信息，就说明 DOSKEY 已经被安装了。此时，键入 DOSKEY /REINSTALL 命令以清除所有已被保存的命令。

接着按顺序激活 CLS、VER 和 VOL 命令。随着每个命令的执行，MS-DOS 在命令表中记录各个命令的名字。通过执行 DOSKEY 时带上/HISTORY(或/H)开关就可显示此命令名清单。在本例中，DOSKEY 显示如下的命令 CLS、VER 和 VOL：

```
C:\>DOSKEY /HISTORY
CLS
VER
VOL
DOSKEY /HISTORY

C:\>
```

显示命令历史的一个简单方法是按 F7 键。当用 F7 来显示命令时，DOSKEY 在每个命令名的前面加上表示该命令在表中相对位置的数字：

```
1:CLS
2:VER
3:VOL
4:DOSKEY /DISTORY
```

如果按 F9 键，DOSKEY 就提示你输入想要调出的命令的行号：

```
C:\>Line number:
```

如果敲入 1 并按 Enter，DOSKEY 就在当前命令行上显示 CLS；如果输入 2，

DOSKEY 就显示 VER；等等。

还可以用下列键来将以前的命令调出至命令行上：

键	功　能
上箭头	调出在当前显示的命令紧前面的命令。
下箭头	调出在当前显示的命令紧后面的命令。
PgUp	调出表中最老的命令。
PgDn	调出表中最新的命令。

每次调出一命令时，MS-DOS 都将光标放在命令尾，这样就可按 Enter 键而快速执行该命令。如果想要在执行前改变此命令，MS-DOS 可允许使用下列编辑键来将光标移至合适的位置：

键	功　能
左箭头	不删除光标前的字符，将光标左移一位(注意：如果未安装 DOSKEY，则该字符就被删去了)。
右箭头	将光标右移一位(注意：如果从一空命令行开始，则显示的会是前一条命令中的下一字符)。
Ctrl-左箭头	将光标左移一词。
Ctrl-右箭头	将光标右移一词(注意：如果从一空命令行开始，则显示的会是前一条命令中的下一词)。
Home	将光标移至命令的开始。
End	将光标移至命令的结束。
Esc	从命令行上清除该命令。

可以用这些键来编辑命令行：

键	功　能
Backspace	删除光标左边的字符。
Delete	删除光标处的字符。
Insert	在插入与覆盖方式间切换。

Insert 键的使用需要进行一些解释。在缺省方式下，新字符替换(覆盖)已在命令行上的字符。当按下 INSERT 时，光标就从下划线改成一方杠，DOSKEY 也就转变成插入方式，新字符就被插入至正文中(原先的字符被向右移动)。当按下 ENTER 后，插入方式就被重新设置成覆盖方式。通过在初次运行(或重新安装)DOSKEY 时说明/INSERT 开关可以修改此过程。如果带/INSERT 开关运行 DOSKEY，插入方式就成为缺省的。按 INSERT 仍然会使光标变成一方框，但现在新字符会替换原先的字符(缺省设置等同于用/OVERSTRIKE 开关)。

MS-DOS 始终将前一条命令作为命令模式而存放。通过编辑旧命令而建立一新的命

令行常常比较方便。可以用这些标准命令编辑键来编辑命令模式中存放的命令：

键	功　能
F1	将前面命令的一字符拷贝至当前命令行。
F2	将前面命令中直到(不包括)按了 F2 键后敲入的字符之间的字符拷贝至当前命令行。
F3	将前面命令中从当前字符至末尾的字符拷贝至当前命令行。
F4	从前面命令中删除直到(不包括)按了 F4 键后敲入的字符间的字符。然后可以通过按 F1、F2 或 F3 来显示该命令的其余部分。从一空命令行开始，用前面命令的字符，就可建立新的命令。
F5	不执行当前显示的命令而使之成为命令模式，并清除当前命令行(注意：若没装入 DOSKEY，则 F5 的功能就如同 ESC 键)。

在使用 MS-DOS 时，用 DOSKEY 的命令重现功能来迅速重复先前的命令可以节省大量的时间和击键次数。

7.2　建立 DOSKEY 宏

DOSKEY 不仅可用来迅速调出先前的命令，而且还可以用来建立宏——类似于简单的批处理文件。与批处理文件相似，宏可以含一个或多个 MS-DOS 将其视为宏名的命令。宏与带 BAT 扩展名并存放在磁盘上的批处理文件的不同之处在于宏存放在计算机的 RAM 内存中，在每次关机时就丢失了。

注意：为使最常使用的宏在每次系统启动时都可使用，可以将定义了宏的 DOSKEY 命令放在 AUTOEXEC.BAT 文件中。

为建立一个宏，DOSKEY 命令后应跟上宏名、等号，以及组成该宏的命令。如果一个宏含不止一个命令，那么可以用 $T(或 $t)元字符来隔开它们。

下面的 DOSKEY 命令建立了一名为 CLSVV 的宏，其功能是先清屏，然后执行 VER 和 VOL 命令：

```
C:\>DOSKEY CLSVV=CLS $T VER $T VOL
```

宏名 CLSVV 必须被说明，后面跟着一等号。注意该宏用 $T 来间隔命令。为执行该宏，只需在命令提示符下敲入该宏名，然后按 ENTER：

```
C:\>CLSVV
```

我们知道，%1 至%9 的批处理文件参数大大提高了批处理文件的功能。DOSKEY 宏也支持命令行参数。但是，在宏中必须用符号 $1 至 $9 来访问参数。下面的宏 SHOWEM 就用这些符号来显示其命令行参数：

```
C:\>DOSKEY SHOWEM=ECHO $1 $2 $3 $4 $5 $6 $7 $8 $9
```

如果带字母 A、B 和 C 运行 SHOWEM，屏幕显示为：

```
C:\>SHOWEM A B C
C:\>ECHO A B C
A B C
```

DOSKEY 宏不支持将参数左移一位的 SHIFT 命令。但是，DOSKEY 宏支持 $ * 元字符，MS-DOS 会用宏的命令行减去宏名来替换此元字符。下面的命令建立了一名为 SHOWIT 的宏，用 $ * 来显示它的所有参数：

```
C:\>DOSKEY SHOWIT=ECHO $ *
```

注意：如同 MS-DOS 批处理文件，宏也在命令执行时显示命令名。但是，不能像对批处理文件那样用 ECHO OFF 来阻止对宏的命令名的显示。

许多人都用 MD(或 MKDIR)来建立一目录，然后立即用 CD(或 CHDIR)使该目录成为当前目录。下面的宏 MDCD 就创建了所说明的目录，然后立即进入该目录：

```
C:\>DOSKEY MDCD=MKDIR $1 $T CHDIR $1
```

为用该宏建立一名为 TEST 的目录并使之成为当前目录，可以这样运行该宏：

```
C:\>MDCD TEST
```

如果所建立的宏与一现存的宏同名，DOSKEY 就会覆盖掉现存的宏。例如，下面的命令修改 SHOWEM 宏以在各自行上显示每一参数：

```
C:\>DOSKEY SHOWEM=FOR %I IN ($ *)DO ECHO %I
```

注意我们用的是 FOR 语句的命令行形式(%I)，而不是批处理文件形式(%%I)。DOSKEY 中的命令形式与从命令行上键入的形式一样。DOSKEY 还支持 MS-DOS I/O 重定向符。但是，如果用标准的符号(>，<和|)，则从 DOSKEY 命令上的输出就被重定向。为在宏中表示重定向符，可用下列符号：

符　号	功　能
$G 或 $g	输出重定向符(>)
$L 或 $l	输入重定向符(<)
$B 或 $b	管道重定向符(|)

下面的宏 FINDNOTE 使用了 FIND 命令来搜索 MS-DOS 6.0 版中的 README.TXT 文件。它将来自 FIND 命令的信息管道重定向至 MORE 以便一次读一屏。

```
DOSKEY FINDNOTE=FIND $1 \DOS\README.TXT /I $B MORE
```

当运行 FINDNOTE 时，可以说明要搜索的词(用双引号括起)；例如：

```
FINDNOTE "MEMMAKER"
```

该宏被扩展成：

```
FIND "MEMMAKER" \DOS\README.TXT /I MORE
```

注意 $B 被管道符号所替代(/I 开关被用来配置 FIND 以在搜索时忽略大小写)。

下一例子 FINDNAME 用了两个参数，分别说明串和文件名。FIND 命令的输出被排序，然后被传至打印机。该文件被用于从一未排序的文件中提取含该串的行，并产生一排好序的清单。FINDNAME 的宏定义如下：

```
DOSKEY FINDNAME=FIND "$1" $2 /I $B SORT $G PRN
```

在该例中，引号括起了变量，这样就不必在命令行上再键入引号。为使用该命令，可以输入：

```
FINDNAME JOE BIRTHDAY.DAT
```

这就变成：

```
FIND "JOE" BIRTHDAY.DAT /I | SORT > PRN
```

然后就会打印出一个排好序的在其名字中含"JOE"的名单。

/MACROS(或/M)开关让 DOSKEY 显示其当前的宏的清单。假定已经建立了我们至今所讨论过的各个宏，则 DOSKEY 会显示如下内容：

```
C:\>DOSKEY /M
CLSVV=CLS $T VER $T VOL
SHOWIT=ECHO $*
MDCD=MDDIR $1 $T CHDIR $1
SHOWEM=FOR %I IN ($*) DO ECHO %I
FINDNOTE=FIND $1 \DOS\APPNOTES.TXT /I $B MORE
FINDNAME=FIND "$1" $2 /I $B SORT $G PRN

C:\>
```

DOSKEY 命令

功能：

可允许用键盘上的箭头键来提出及编辑先前执行过的命令。还可以对含一条或多条 MS-DOS 命令的宏进行定义。

格式：

[驱动器:][路径]DOSKEY [/REINSTALL][/BUFSIZE=缓冲区大小]

[/MACROS 模 /M][/HISTORY 模 /H][/INSERT 模 /OVERSTRIKE][宏名=正文]

注释：

DOSKEY 是一内存驻留的软件程序。

/REINSTALL 开关让 DOSKEY 清除缓冲区的所有命令和宏，然后将 DOSKEY 的新版本安装至内存。

DOSKEY 将命令和宏放在内存的缓冲区中。可以用/BUFSIZE=缓冲区大小开关来在第一次执行 DOSKEY 或使用/REINSTALL 开关时说明该缓冲区的大小。缺省时，缓冲区大小为 1024 字节。如果有大量的宏被建立，可以将缓冲区大小增加至 4096 字节。

DOSKEY /MACROS 开关显示当前在内存中的宏。/HISTORY 开关则显示缓冲区中的各个命令。

/INSERT 开关设置插入方式，这样新敲入的字符就被插入在行上。/OVERSTRIKE 开关则用新字符来替换老字符。INSERT 键也可被用来临时切换方式。

为建立一宏，说明其名字，后跟一等号，然后是组成该宏的命令，这些都必须在一行上。命令之间用元字符 $T 或 $t 隔开。

例子：

下面的命令显示当前命令缓冲区的内容以及所有定义了的宏：

```
C:\>DOSKEY /HISTORY /MACROS
```

下面的 DOSKEY 命令建立一名为 CK 的宏，用以简写 MS-DOS 的 CHKDSK 命令：

```
C:\>DOSKEY CK=CHKDSK
```

7.2.1 宏的局限

如前所述，宏类似于但不等同于批处理文件。例如，在宏中不可以用 ECHO OFF 命令来消除命令名的显示。类似地，不可以在宏中用 GOTO 命令来进行转移。MS-DOS 允许从一宏中运行一批处理文件，但却不可以从一批处理文件中运行一宏，也不可以从一宏中运行另一宏。但是，可以用批处理文件来定义一个或多个宏。批处理文件的执行可用 Ctrl-C 来中断。但在宏中，Ctrl-C 只能终止当前命令。因此，必须为宏中的每一条命令都按 Ctrl-C 才能终止该宏的执行。

7.2.2 重定义 MS-DOS 命令

我们知道，不可以将一条 MS-DOS 的内部或外部命令的名字赋给一批处理文件。但是，有时可能想要将一 MS-DOS 命令的名字赋给一宏。当从命令行上执行该命令时，MS-DOS 首先检查该命令是否匹配一 DOSKEY 宏。如果没有，MS-DOS 就检查该命令是否是一内部命令，接着是外部命令或批处理文件。将宏名定义成与一 MS-DOS 命令相同的名字，就可以有效地重新定义或屏蔽该命令。

例如，下面的 DOSKEY 命令建立了一名为 FORMAT 的宏，它总是格式化一盘后，将系统文件拷贝至该盘上：

```
C:\>DOSKEY FORMAT=FORMAT /S $ *
```

每次用此 FORMAT 宏格式化一磁盘时，$ * 就被替换成所说明的选项(诸如待格式化的驱动器)。例如，敲入：

```
FORMAT A:
```

DOEKEY 宏就将它替换成：

```
FORMAT /S A:
```

该宏替换标准的FORMAT 命令并插入必要的开关。它的一个缺点是如果重复了此开关，则 FORMAT 命令会返回一错误。所以，如果敲入 FORMAT /S A:，MS-DOS 就会出错。

DOSKEY 的这种用法可对任何 MS-DOS 命令进行预置(很类似于将 DIR 命令预置成 DIRCMD 环境项)。这一性质还可被用来屏蔽一 MS-DOS 命令。为屏蔽 FORMAT 命令，可用下列宏：

```
DOSKEY FORMAT=CLS $T ECHO That Command Is Not Available
```

这样，每当试图运行 FORMAT 命令时，就会被清屏，然后 MS-DOS 显示：

```
ECHO That Command Is Not Available
That Command Is Not Available
```

在使用宏时，没有办法屏蔽掉 ECHO 命令的显示。

由于 MS-DOS 处理宏的方式，批处理文件中不会识别出宏。所以，如果有一批处理文件含有 FORMAT 命令，则 MS-DOS 会执行通常的命令，而不是 FORMAT 的宏定义。此外，如果在命令行上说明了命令的整个路径(\DOS\FORMAT)，则执行的也是原先的命令而不是宏。最后，在命令名前放一空格，就总是执行 MS-DOS 命令(而不是宏)。

7.2.3 用 DOSKEY 宏来节省磁盘空间

每次 MS-DOS 建立一文件时，都分配一个或多个磁盘存储单元(簇)来存放文件内容。在典型的硬盘上，一簇含 4096 字节。无论文件大小，MS-DOS 对任一文件都至少分配一个簇。例如，假定有一个文件只含 10 个字节，MS-DOS 也分配一个簇来存放该文件。这时，只有 10 个字节被使用，而另外 4086 个字节就被浪费了。很明显，磁盘上小文件的数量越多，被浪费的磁盘空间也就越多。

由于 MS-DOS 将 DOSKEY 宏存放在内存中，而非磁盘上，所以宏不会像小的批处理文件那样浪费磁盘空间。如果经常使用一组 DOSKEY 宏，那么可以将这些宏的定义放进 AUTOEXEC.BAT 文件中。通过将较小的批处理文件转化成宏，就可以从磁盘上删除这些批处理文件，从而释放大量的磁盘空间。

7.2.4 将宏定义保存在文件中

如果建立了若干宏后，想将它们的定义放入一文件，那么可以重定向 DOSKEY/MACROS 命令的输出至一带 BAT 扩展名的文件中：

```
C:\>DOSKEY /MACROS > MACDEFS.BAT
```

DOSKEY 就会在 MACDEFS.BAT 文件中为每一个宏定义建立一行内容。例如，如果已经定义了三个宏(CP、PR 和 CLSVV)，那么，现在它们就会在 MACDEFS.BAT 文件中：

```
CP=COPY $1 $2
PR=PRINT $*
CLSVV=CLS $T VER $T VOL
```

通过编辑该文件，在每行前加上 DOSKEY 命令，就可以很快建立一批处理文件来定义你最常使用的宏：

```
DOSKEY CP=COPY $1 $2
DOSKEY PR=PRINT $*
DOSKEY CLSVV=CLS $T VER $T VOL
```

7.2.5 建立宏库

以与建立 MS-DOS 批处理文件的库相同的方式，可以建立 DOSKEY 宏的库。本节讨论你可能想加至宏库中的几个简单的宏。

文件隐藏属性的修改

从 MS-DOS 5.0 版开始，可以用 ATTRIB 命令来改变一文件的隐藏属性。当该属性被设置时，文件就不再出现在目录显示中，虽然该文件仍然安全地放在磁盘上。当未设置隐藏属性时，该文件就出现在目录显示中。下面的 DOSKEY 宏 HIDE 设置一文件的隐藏属性：

```
C:\>DOSKEY HIDE=ATTRIB +H $1
```

在定义了该宏后，可以如下这样隐藏文件：

```
C:\>HIDE FILENAME.EXT
```

以类似的方式，UNHIDE 宏移去隐藏属性：

```
C:\>DOSKEY UNHIDE=ATTRIB -H $1
```

为移去一文件的隐藏属性，用下面的命令来运行该宏：

```
C:\>UNHIDE FILENAME.EXT
```

有时可能会想要查看盘上隐藏文件的名字。下面的宏 SHOWHIDE 可显示当前盘上所有隐藏文件的目录：

```
C:\>DOSKEY SHOWDIR=DIR \*.* /A:H /S
```

磁盘格式化操作的简化

如果想在1.2MB软驱中格式化360KB的盘，或在1.44MB软驱中格式化720KB的盘，那么可以使用下面的DOSKEY宏MAKE360和MAKE720来使此过程变得非常简单：

```
C:\>DOSKEY MAKE360=FORMAT $1 /F:360
C:\>DOSKEY MAKE720=FORMAT $1 /F:720
```

为在A盘中格式化一360KB的软盘，就可以如下地执行MAKE360宏：

```
C:>MAKE360 A:
```

第 8 章

使用 ANSI.SYS 来改进批处理文件

8.1 装载 ANSI.SYS

本章详细讨论 ANSI.SYS 设备驱动程序，使用该程序可以设置屏幕颜色、清屏以及在屏幕上设定光标位置，这样你的批处理文件就能在特定位置显示其输出，并将命令赋给功能键。以后我们会看到，使用 ANSI.SYS 不仅很有趣，而且还能大大提高批处理文件的功能。

ANSI.SYS 设备驱动程序是存放在 DOS 目录中的。文件扩展名 SYS 说明 ANSI.SYS 是一设备驱动程序。设备驱动程序是一个用来控制系统与某一设备或一组设备通信的程序。在 ANSI.SYS 中，设备是控制台 CON，指键盘(用于输入)和屏幕(用于输出)。

在使用 ANSI.SYS 之前，必须将它装入内存。为装入一设备驱动程序，应在 CONFIG.SYS 文件中加上一条 DEVICE=(或 DEVICEHIGH=)命令。MS-DOS 在每次系统启动时都执行 CONFIG.SYS 中的命令以配置 MS-DOS 环境和 MS-DOS 与设备的通信。对 ANSI.SYS 来说，CONFIG.SYS 中应加入：

```
DEVECE=C:\DOS\ANSI.SYS
```

如果用的是 MS-DOS 5.0 或更新的版本，并且正在使用内存高区，那么可以用 DEVICEHIGH=来装载 ANSI.SYS 设备驱动程序：

```
DEVICEHIGH=C:\DOS\ANSI.SYS
```

在将上述命令加至 CONFIG.SYS 文件后，重新启动系统就可使所作的修改开始起作用。

8.2 ANSI.SYS 转义序列

为从批处理文件中利用 ANSI.SYS 所增强了的有关屏幕和键盘的功能，必须使用 PROMPT 命令或 ECHO 命令来发出为该设备驱动程序所解释的转义序列。一个转义序列是以 ASCII 转义字符(ASCII 27)开始的独一无二的字符组合。ANSI.SYS 设备驱动程序控制 MS-DOS 与屏幕和键盘的通信，并支持可用来设置屏幕颜色、设置光标位置或重定义键盘上的键的若干转义序列。PROMPT 命令提供了对 ANSI 转义序列作试验的最简单的方法。利用 PROMPT 命令，就可以或者直接从命令行上，或者从批处理文件中使用转义序列。

PROMPT 可以方便地在转义序列中加入转义字符。PROMPT 命令使用元字符(以

美元号开始的两个字符的组合)来说明在显示提示符时将被插入的值。例如，$P 元字符表示当前驱动器和目录，$G 元字符表示大于号(>)。这两个字符组合起来就形成了标准的命令提示符。为了在 PROMPT 命令中使用转义序列，可使用 $e 元字符来表示转义字符。当显示提示符时，该转义字符就被替换成 $e。接着，跟在转义字符后面的指令就被 ANSI.SYS 解释而产生所要求的变化。例如，为用 PROMPT 命令来清屏，可输入：

```
C:\>PROMPT $e[2J
```

[2J 是 ANSI 用来清屏的控制序列。字母 J 必须是大写。每次按 Enter 键时，屏幕就被清除，然后光标就被移至左上角。如你所知，每次按 ENTER 时，MS-DOS 都显示提示符。在此例中，显示提示符同时清屏。为返回至缺省的命令提示符，可输入下列命令：

```
C:\>PROMPT $P$G
```

使用 PROMPT 命令来发送转义序列至屏幕的一个缺点是在每次发送时都改变了提示符。幸运的是，有一种方法(我们将在本章后面讨论)可以解决这问题。另一个缺点是为了使 PROMPT 命令产生的字符能被 ANSI.SYS 所解释，必须将 ECHO 打开。

发送转义序列至屏幕的另一种方法是使用 ECHO 命令。由于不能在命令提示符下用 ECHO 命令来说明转义字符，所以只能从一批处理文件中用 ECHO 来发送一转义序列。

转义字符不能在屏幕上被显示。如果用 TYPE 命令来显示含一转义序列的批处理文件，则转义序列将被 ANSI.SYS 所解释，而不是显示在屏幕上。如同前面讨论过的蜂鸣字符，转义字符也是引起某事发生的控制字符。在转义字符中，具体发生什么是由跟在转义字符后面的 ANSI 代码决定的。

对某些控制字符来说(如蜂鸣符 ^G)，有一个该字符在文件中应如何被显示的约定。但是，对转义字符来说则不存在这样的约定。可以在正文编辑器中看见转义序列，但不同的编辑器对它们的显示也各不相同。用来将转义字符插入至批处理文件中的步骤也随正文编辑器的不同而相差很大。本书一般用左箭头(←)来描述转义字符，因为这是 EDIT 所用的符号。

为在 EDIT 中插入一转义字符，可以使用先前所使用的 Ctrl-P 序列来插入响铃字符。只需同时按下 Ctrl 和 P 键即可。然后可以按 ESC 键来将转义字符加至正文中。它显示一左箭头。例如，输入下列命令可打开一个名为 CLRSCRN.BAT 的新文件：

```
C:\>EDIT CLRSCRN.BAT
```

现在敲入词 ECHO，后再跟一空格。接着再按住 CTRL 键，按下 P 键。为插入转义字符，按 ESC 键(通常在键盘的左上角)。屏幕应类似于图 8-1。

输入[2J 并按 ENTER 键就完成了此清屏命令。为使该序列产生作用，J 必须是大写的。最后保存该文件(ALT-F,S)并退出 EDIT 程序(ALT-F,X)。当输入 CLRSCRN 命令时，批处理文件就清屏。

如果更喜欢使用 EDLIN(或用的是 MS-DOS 5.0 以前的版本而不得不使用 EDLIN)，则用来插入转义字符的组合键是 Ctrl-V。为用 EDLIN 建立同样的批处理文件，执行下列命令：

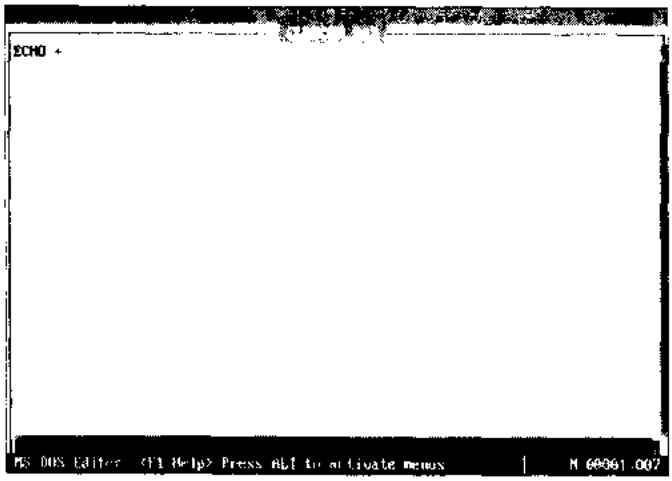


图 8-1 转义字符的屏幕表示(左箭头)

```
C:\>EDLIN CLRSCRN2.BAT
```

在新文件中,必须先敲入 I 后按 ENTER 键以进入 EDLIN 的插入方式。然后,敲入 ECHO,后跟一空格。再按住 CTRL 键,按下 V 键。屏幕上这时应显示:

```
C:\>EDLIN CLRSCRN2.BAT
New file
*I
    ECHO ^V
```

^V 表示转义字符。现在敲入[2J 就输入了 ANSI 用来清屏的控制序列。按 ENTER 键移至下一行后,再按 CTRL-C 退出 EDLIN 的插入方式。最后,敲入 E 并按 Enter 就退出 Edlin 并保存了该文件。每次执行 CLRSCRN2 时,屏幕原先的内容就全部消失,光标被移至左上角。

当然,用 CLS 命令就可做到这一切,但现在你了解了转义序列是怎样用于控制显示的。更重要的是,你知道了如何在批处理文件中输入转义字符。

8.3 设置屏幕颜色

在本节我们讨论如何用 ANSI.SYS 和 PROMPT 命令来设置屏幕颜色。ANSI.SYS 用来设置屏幕颜色的转义序列如下:

```
←[colorm
```

这里,左指示箭(←)是 ASCII 的转义字符,该转义序列以字母 M 结束,color 是下表中定义的一个颜色值:

color 值	颜　色	color 值	颜　色
0	缺省颜色(黑和白)	35	紫色前景
1	粗体正文属性	36	青色前景
2 *	低亮度正文属性	37	白色前景
3 *	斜体正文属性	40	黑色背景
4	对 IBM 单显而言,被打开的下划线,对 VGA 而言,下划线的颜色(蓝色)	41	红色背景
5	闪烁正文属性	42	绿色背景
6 *	迅速闪烁正文属性	43	黄色背景
7	反相视频正文属性	44	蓝色背景
8	取消正文属性	45	紫色背景
30	黑色前景	46	青色背景
31	红色前景	47	白色背景
32	绿色前景	48	下标
33	黄色前景	49 *	上标
34	蓝色前景		

* 在 VGA 监视器上不可使用。

注意：并非所有上面这些值都为所有 ANSI.SYS 的版本所支持。

例如,下面的 ANSI.SYS 转义序列就将屏幕前景颜色设置成红色:

```
←[31M
```

记住,左箭头表示转义字符。下面的序列将屏幕背景色置成绿色:

```
←[42M
```

下面的批处理文件 CYANBG.BAT 用 PROMPT 命令的 $e 元字符(表示转义)来将屏幕的背景色置成青色:

```
PROMPT $e[46m
PROMPT [$p]
CLS
```

$e 元字符是 PROMPT 的 ASCII 转义字符。第一条命令将背景色置成青色,第二条 PROMPT 命令将命令提示符重新设置成当前路径(驱动器和目录),并在显示时夹在两个方括号之间。CLS 命令清除屏幕显示,并将整个屏幕的背景色置成青色(而不只是提示符周围那一部分)。

类似地,下面的批处理文件 BLUEFG.BAT 用 PROMPT 命令来将屏幕的前景色置成蓝色:

```
PROMPT $e[34m
PROMPT [$p]
```

```
CLS
```

如果运行此批处理文件，ANSI.SYS 设备驱动程序就将屏幕的前景置成蓝色，而屏幕背景色则不改变。

8.3.1 使用 SETCOLOR.BAT

从上面可以看出，CYANBG.BAT 与 BLUEFG.BAT 两个文件的内容几乎完全一样——唯一的差别仅在颜色值上。所以，可以建立一个名为 SETCOLOR.BAT 的批处理文件，以根据%1 参数的值而设置屏幕颜色属性。在此例中，该批处理文件的内容如下：

```
IF "%1"=="" GOTO DONE
PROMPT $e[%1m
PROMPT [$p]
CLS
:DONE
```

为设置屏幕颜色，只需将颜色值说明成一个批处理文件的参数。例如，下面的命令行就将屏幕的背景色置成绿色：

```
C:\>SETCOLOR 42
```

MS-DOS 用值 42 来替代%1，从而产生下面的转义序列：

```
←[42m
```

用此批处理文件既可设置前景色，也可设置背景色。事实上，可以先运行它一次来设置前景色，再运行一次来设置背景色，最后一次则设置正文属性。如果输入下列命令，最后就会得到在青色背景上闪烁的紫色正文：

```
SETCOLOR 35
SETCOLOR 46
SETCOLOR 5
```

用 SETCOLOR 0 命令可以将屏幕颜色置成缺省的黑色背景上的白色前景。

8.3.2 恢复提示符

至今所讨论的批处理文件都是用 PROMPT 命令来生成设置屏幕颜色的转义序列的。在这些例子中，批处理文件的最后都是一条把提示符恢复成方括号内的当前驱动器和目录(如[C:\])的 PROMPT 命令。一个更好的方法是让批处理文件恢复成在开始运行此批处理文件时的提示符。下面的批处理文件 SETCLR2.BAT 的目的就是如此：

```
IF "%1"=="" GOTO DONE
SET ORIGINAL_PROMPT=%PROMPT%
PROMPT $e[%1m
PROMPT %ORIGINAL_PROMPT%
```

```
SET ORIGINAL_PROMPT=
CLS
:DONE
```

该批处理文件一开始先将当前提示符的值赋给一个名为 ORIGINAL_PROMPT 的环境变量。为此，批处理文件用了%PROMPT%有名参数。每条 PROMPT 命令都改变%PROMPT%的值。在该文件结束前，它用%ORIGINAL_PROMPT%有名参数和 PROMPT 命令来恢复原先的提示符。以这种方式使用有名参数，批处理文件就能将产生副作用(由于批处理文件的执行而引起的未曾预料的改变)的可能性减至最小。

当然，可以用 ECHO 命令而不是 PROMPT 命令来改变屏幕颜色以完成上述批处理文件的功能。使用 ECHO 方法，批处理文件就可以只有一行——带相应参数的 ECHO 命令。

8.3.3 使用正文属性

如果不清楚何时批处理文件会使用 ANSI.SYS 提供的粗体，闪烁或反相视频属性，那么考虑显示下面信息的一应用程序：

```
About to delete FILENAME.EXT
Press any key to continue . . .
```

在本书前面，我们在用 PAUSE 显示一信息时用了计算机的内部"响铃"引起用户注意。下面的批处理文件 DELETE.BAT 使用闪烁正文属性来引起用户注意：

```
@ECHO OFF
IF "%1"=="" GOTO DONE
REM Set text attribute to blinking.
REM Must turn ECHO on for PROMPT command to
REM change the text attribute.
PROMPT $e[5m
ECHO ON
ECHO OFF
CLS
PROMPT [$p]
REM Displaying warning message
ECHO About to delete %1
PAUSE
DEL %1
PROMPT $e[0m
REM Restore default color. Again, turn ECHO on.
ECHO ON
ECHO OFF
PROMPT [$p]
:DONE
```

该批处理文件显示前面说明的闪烁信息，但在它完成执行时，将屏幕颜色置成缺省设置。如果用 ANSI.SYS 来设置屏幕颜色，那么该颜色就被丢失了。为解决此问题，必须始终用一记录有关值的批处理文件来设置屏幕颜色。下面的批处理文件 SCRCOLOR.BAT 定义

了三个有名参数——TEXTATTR、REGROUND 和 BACKGROUND：

```
@ECHO OFF
:LOOP
IF "%1"=="" GOTO DONE
PROMPT $e[%1m
ECHO ON
ECHO OFF
PROMPT [$p]
CLS
@ECHO OFF
REM Set the named parameters TEXTATTR, FOREGROUND,
REM and BACKGROUND
IF "%1"=="0" SET TEXTATTR=0
IF "%1"=="1" SET TEXTATTR=1
IF "%1"=="2" SET TEXTATTR=2
IF "%1"=="3" SET TEXTATTR=3
IF "%1"=="4" SET TEXTATTR=4
IF "%1"=="5" SET TEXTATTR=5
IF "%1"=="6" SET TEXTATTR=6
IF "%1"=="7" SET TEXTATTR=7
IF "%1"=="8" SET TEXTATTR=8
IF "%1"=="30" SET FOREGROUND=30
IF "%1"=="31" SET FOREGROUND=31
IF "%1"=="32" SET FOREGROUND=32
IF "%1"=="33" SET FOREGROUND=33
IF "%1"=="34" SET FOREGROUND=34
IF "%1"=="35" SET FOREGROUND=35
IF "%1"=="36" SET FOREGROUND=36
IF "%1"=="37" SET FOREGROUND=37
IF "%1"=="40" SET BACKGROUND=40
IF "%1"=="41" SET BACKGROUND=41
IF "%1"=="42" SET BACKGROUND=42
IF "%1"=="43" SET BACKGROUND=43
IF "%1"=="44" SET BACKGROUND=44
IF "%1"=="45" SET BACKGROUND=45
IF "%1"=="46" SET BACKGROUND=46
IF "%1"=="47" SET BACKGROUND=47
IF "%1"=="48" SET BACKGROUND=48
IF "%1"=="49" SET BACKGROUND=49
SHIFT
GOTO LOOP
:DONE
```

由此可见，根据在命令行上作为 SCRCOLOR.BAT 的第一个参数而说明的颜色值，该批处理文件定义三个有名参数中的一个。SHIFT 命令的使用是为了能够在一个命令行上键入所有三个值。

接着，就可以建立改变屏幕属性、测试是否存在有名参数之一，并当其存在时恢复原先的屏幕颜色的批处理文件。为此修改 DELETE.BAT 为如下内容：

```
@ECHO OFF
IF "%1"=="" GOTO DONE
```

```
REM Set text attribute to blinking.
REM Must turn ECHO on for PROMPT command to
REM change the text attribute.
SET ORIGINAL_PROMPT=%PROMPT%
PROMPT $e[5m
ECHO ON
ECHO OFF
CLS
PROMPT [$p]
REM Displaying warning message
ECHO About to delete %1
PAUSE
DEL %1
REM Restore default color. Again, turn ECHO on.
ECHO ON
IF NOT "%TEXTATTR%"==""  PROMPT $e[%TEXTATTR%m
IF NOT "%FOREGROUND%"==""  PROMPT $e[%FOREGROUND%m
IF NOT "%BACKGROUND%"==""  PROMPT $e[%BACKGROUND%m
ECHO OFF
CLS
PROMPT %ORIGINAL_PROMPT%
SET ORIGINAL_PROMPT=
:DONE
```

当然，为使该批处理文件能运行，还必须先用 SCRCOLOR.BAT 来设置原先的屏幕属性和颜色以使它们的值被保存在环境变量中。该批处理文件的唯一不足之处是如果用 CTRL-C 来中止它(而不是用 DEL 操作来继续它)，则屏幕属性就仍被置为闪烁的。一个更好的方法是用 ECHO 命令来设置屏幕属性，该方法还避免了不得不修改原先的提示符。

在许多情况下，可能会想将用户的注意吸引到单个词上，为此可以使用 ECHO 命令而不是 PROMPT 命令。下面的批处理文件 DELETE2.BAT 建立了一个警告信息，文件名是在白色背景上闪烁红色。接着，它重置屏幕属性(假定它们原先是用 SCRCOLOR.BAT 设置的)。在此批处理文件中，用了一条 ECHO 命令来改变几个屏幕属性：

```
@ECHO OFF
IF "%1"=="" GOTO DONE
REM Display a warning message with the
REM filename set to blinking red on white.
CLS
ECHO About to delete ←[5m←[31m←[47m%1
REM Restore default color
ECHO ←[0m
IF NOT "%TEXTATTR%"=="" ECHO ←[%TEXTATTR%m
IF NOT "%FOREGROUND%"=="" ECHO ←[%FOREGROUND%m
IF NOT "%BACKGROUND%"=="" ECHO ←[%BACKGROUND%m
REM Provide the user the option to cancel or continue
ECHO Press Ctrl-C to cancel
PAUSE
REM Delete the file
DEL %1
:DONE
```

由于有可能在批处理文件执行前没有设置任何属性，所以在使用存放在%TEXTATTR%、%FOREGROUND%或%BACKGROUND%中的任何属性值之前，都重新设置了黑白缺省属性。该方法先在批处理文件中关闭属性集，然后使用环境变量来打开在开始执行批处理文件时的属性集。同样，只有在用 SCRCOLOR.BAT 来设置颜色时，此方法才奏效。从这个例子中可以看出，一条 ECHO 或 PROMPT 命令可以设置几个属性。该方法唯一的缺点是用来恢复缺省颜色的四条 ECHO 命令都产生了一空行。

8.4 光标定位

除了清屏及置屏幕的前景和背景色，还可以在批处理文件中用 ANSI.SYS 来控制光标在屏幕上的位置。在用 ANSI.SYS 转义序列定位光标后，下一次输出就出现在屏幕的那个位置处。

下表列出了 ANSI.SYS 的五个光标定位的转义序列：

转义序列	结　果
←[NumRowsA	光标上移 NumRows 行
←[NumRowsB	光标下移 NumRows 行
←[NumRowsC	光标右移 NumRows 列
←[NumRowsD	光标左移 NumRows 列
←[Row;ColH	光标被定位在 Row,Col 处

大多数屏幕都显示 25 行，80 列。屏幕的左上角是 1,1——即第 1 行第 1 列。屏幕的右下角被定义成 25,80。下面的批处理文件 CURPOS.BAT 用 ECHO 命令和 ANSI 转义序列来在屏幕位置(1,1)、(5,5)、(10,10)和(20,20)处显示信息：

```
CLS
ECHO ←[1;1HRow 1,Column 1
ECHO ←[5;5HRow 5, Column 5
ECHO ←[10;10HRow 10,Column 10
ECHO ←[20;20HRow 20,Column 20
```

下面的批处理文件片断说明了如何月 ANSI.SYS 的光标定位转义序列来显示关于 CLS 命令的信息。它在屏幕中央显示粗体的命令名以及不同颜色的信息：

```
@ECHO OFF
CLS
ECHO ←[0M
ECHO ←[1M
ECHO ←[1;39HCLS
ECHO ←[34M
ECHO ←[3;10HCOMMAND TYPE:
ECHO ←[3;25HINTERNAL
ECHO ←[5;10HFUNCTION:
ECHO ←[5;25HErases the screen display, placing the cursor
ECHO ←[6;30Hat the upper-left, or home, position.
```

```
ECHO ←[8;10HEXAMPLE:
ECHO ←[8;25HCLS
```

在此例中，首先屏蔽所有当前设置的属性以防止可能出现的颜色不一致。然后该批处理文件用 ANSI.SYS 将正文属性设置成粗体，并在屏幕中央显示 CLS 命令名。接着，该文件选择前景色并显示信息。可以看出，该批处理文件用了下列形式的转义序列来在屏幕上设定光标位置：

```
←[row;colH
```

其中，COMMAND TYPE:、FUNCTION:和 EXAMPLE:左边都空出 10 列，其他的则缩进 25 列或 30 列。还可以插入转义序列以改变各显示部分的颜色。例如可以使标题成为白色的，而正文成为蓝色的。在批处理文件显示 CLS 信息后它就像前面的例子所示的那样重新设置属性(假定属性已用 SCRCOLOR.BAT 设置了)。

8.5 键的重定义

ANSI.SYS 设备驱动程序既可改进屏幕，又可改进键盘功能。我们已经将 ANSI.SYS 用于清屏，设置前景和背景颜色，以及定位光标。为改进键盘性能，ANSI.SYS 可允许重新定义键。

键盘上的每个键都是用唯一的某个值来标识的。这个所谓扫描码的值可和 ANSI.SYS 一起使用以将一正文串赋给键盘上的任一键。这样，就可以利用此性质来将某命令或正文赋给一特定的键，从而不必重复敲入该正文。一般来说，这样的赋值是针对功能键，而不是标准键盘字符的。

由于 F1 至 F4 被 MS-DOS 用来编辑命令行，而 F1 至 F9 以及与 Alt 组合的某些功能键为 DOSKEY 所用，所以最好用 Shift 或 Ctrl 组合的功能键来作为自己键赋值的对像。下面的扫描码是和 Shift 与 Ctrl 组合的十个标准功能键：

功能键	+Shift 的扫描码	+Ctrl 的扫描码
F1	84	94
F2	85	95
F3	86	96
F4	87	97
F5	88	98
F6	89	99
F7	90	100
F8	91	101
F9	92	102
F10	93	103

标准键盘字符的扫描码表在本章后面给出。

为将一字符串赋给某个功能键，可用下列格式的 ANSI.SYS 转义序列：

```
←[0;扫描码;"串"P
```

例如，为将 VER 命令赋给 Shift-F7 键组合，转义序列是：

```
←[0;90;"VER"P
```

类似地，为将 CLS 命令赋给 Shift-F10 键组合，转义序列是：

```
←[0;93;"CLS"P
```

注意：由于这些例子中的扫描码说明了被称为扩充键的特殊键，所以转义序列的第一个参数是 0，然后跟一分号和扫描码的值。扩充键包括功能键(F1 至 F12)、箭头键、编辑键以及这些键与 Shift、Ctrl 或 Alt 的任意组合。

如果想让任何其他键与 Alt 键组合，那么转义序列的第一个参数也必须为 0。对不是扩充键的扫描码来说，则可省去 0 及其分号。

下面的批处理文件 DEFKEY.BAT 使用了 ANSI.SYS 的键盘重新赋值转义序列来定义键盘上的一个功能键：

```
@ECHO OFF
IF "%1"=="" GOTO DONE
IF "%2"=="" GOTO DONE
ECHO ←[0;%1;"%2"P
:DONE
```

该批处理文件用%1 参数来决定定义哪个功能键(或键组合)，用%2 参数来确定将赋给该键的值。例如，为将 CLS 命令赋给 SHIFT-F10 键组合，命令行为：

```
C:\>DEFKEY 93 CLS
```

该批处理文件用值 93 和 CLS 来建立下面的转义序列：

```
ECHO ←[0;93;"CLS"P
```

还可以将一个含空格的正文串赋给某键：

```
C:\>DEFKEY 93 DIR *.* /P
```

为此，只需修改 DEFKEY.BAT 文件以使用%1 至%9 的参数：

```
IF "%1"=="" GOTO DONE
IF "%2"=="" GOTO DONE
ECHO ←[0;%1;"%2 %3 %4 %5 %6 %7 %8 %9"P
:DONE
```

在建立了 DEFKEY.BAT 文件后，可以从 AUTOEXEC.BAT 中用 CALL 或 COMMAND /C 命令来运行它。这样，你的键盘定义就会在每次系统启动时起作用。

上面我们讨论了给键盘重新赋值的用途，以及如何进行重新赋值。下表是你可能想要

重新定义的扩充键和键组合的扫描码：

键/键组合	扫描码
SHIFT-TAB	15
Alt-q,w,e,r,t,y,u,i,o,p	16-25
Alt-a,s,d,f,g,h,j,k,l	30-38
Alt-z,x,c,v,b,n,m	44-50
F1-F10	59-68
Home	71
Up(上箭头)	72
PgUp	73
Left(左箭头)	75
Right(右箭头)	77
End	79
Down(下箭头)	80
PgDn	81
Insert	82
Delete	83
Shift-F1-Shift-F10	84-93
Ctrl-F1-Ctrl-F10	94-103
Alt-F1-Alt-F10	104-113

8.6 使用 IBM 的扩充字符表

计算机用从 0 到 255 的值来表示所有数字、字母和符号。前 128 个值(从 0 到 127)表示通常使用的数字，字母和标点符号。这些值被称作 ASCII 字符集。后 128 个值(从 128 到 255)提供了为 IBM PC 及其兼容机画图的符号等杂类字符。这些字符被称作 IBM 扩充字符集，见下表：

字符	值	字符	值	字符	值	字符	值
Ç	128	á	160	└	192	α	224
ü	129	í	161	┴	193	β	225
é	130	ó	162	┬	194	Γ	226
â	131	ú	163	├	195	π	227
ä	132	ñ	164	─	196	Σ	228
à	133	Ñ	165	┼	197	σ	229
å	134	ª	166	╞	198	μ	230
ç	135	º	167	╟	199	τ	231
ê	136	¿	168	╚	200	Φ	232
ë	137	⌐	169	╔	201	Θ	233
è	138	¬	170	╩	202	Ω	234
ï	139	½	171	╦	203	δ	235
î	140	¼	172	╠	204	∞	236

ì	141	¡	173	═	205	φ	237
Ä	142	«	174	╬	206	ε	238
Å	143	»	175	╧	207	∩	239
É	144		176	╨	208	≡	240
æ	145	█	177	╤	209	±	241
Æ	146	█	178	╥	210	≥	242
ô	147	│	179	╙	211	≤	243
ö	148	┤	180	╘	212	⌠	244
ò	149	╡	181	╒	213	⌡	245
û	150	╢	182	╓	214	÷	246
ù	151	╖	183	╫	215	≈	247
ÿ	152	╕	184	╪	216	°	248
Ö	153	╣	185	┘	217	•	249
Ü	154	║	186	┌	218	·	250
¢	155	╗	187	█	219	√	251
£	156	╝	188	▄	220	ⁿ	252
¥	157	╜	189	▌	221	²	253
₧	158	╛	190	▐	222	■	254
ƒ	159	┐	191	▀	223		255

随着你建立的批处理文件越来越复杂，你可能想通过设置菜单或方框中的其他信息来增强批处理文件所生成的屏幕显示。例如，可以建立名为 BOXMENU.BAT 的显示下列菜单的批处理文件：

```
┌──────────────────────────────────┐
│ 1 - Display directory listing    │
│                                  │
│ 2 - Display MS-DOS version number│
│                                  │
│ 3 - Display disk volume label    │
│                                  │
│ 4 - Quit                         │
└──────────────────────────────────┘
Enter your choice (1, 2, 3, or 4):
```

该批处理文件用 IBM 扩充字符值 179、191、192、196、217 和 218 来画一方框：

```
┌                  ─                  ┐
 218              196              191

│179                               179

 192              196              217
└                  ─                  ┘
```

该批处理文件可用 EDIT（或必要时用 EDLIN）来建立。开始时，用下列命令建立一名为 BOXMENU.BAT 的新文件：

C：\>EDIT BOXMENU.BAT

然后键入@ECHO OFF 和 CLS 命令，还有标号:LOOP。

接着，在一新行上敲入 ECHO，但不要按 ENTER。按住 ALT 键，再按数字小键盘上

的 218(如果在 Edlin 中,可以用同样的方法来插入字符——Alt 加上 ASCII 代码)。当放下 Alt 键时,屏幕就显示方框的左上角:

```
@ECHO OFF
CLS
:LOOP
ECHO ┌
```

现在按住 Alt 键,再敲入数字小键盘上的 196。当放下 Alt 键时,MS-DOS 就显示从方框顶部延伸下去的一水平线字符:

```
@ECHO OFF
CLS
:LOOP
ECHO ─
```

再敲入 34 个这样的字符:

```
@ECHO OFF
CLS
:LOOP
ECHOZ ┌──────────────────────────────────
```

按住 Alt 键,敲入 191 就生成了右上角,这样方框顶部就完成了:

```
@ECHO OFF
CLS
:LOOP
ECHO ┌──────────────────────────────────┐
```

用 IBM 扩充字符值 179 来生成垂直条字符,下面七行的内容成为:

```
@ECHO OFF
CLS
:LOOP
ECHO ┌──────────────────────────────────┐
ECHO │  1 - Display directory listing   │
ECHO │                                  │
ECHO │  2 - Display MS-DOS version number │
ECHO │                                  │
ECHO │  3 - Display disk volume label   │
ECHO │                                  │
ECHO │  4 - Quit                        │
```

该方框的底部很类似于顶部,差别仅在于必须用 IBM 扩充字符值 192 和 217 来生成方框的左下角和右下角。

现在,就完成了菜单显示的建立,下面还需要提供完成这些操作的命令。在 MS-DOS

6及以后版本中,最方便的方法是使用CHOICE命令。为完成此批处理文件,键入下列内容:

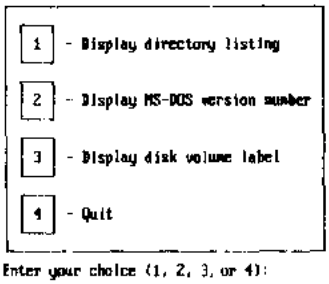
```
REM Evaluate menu choices
CHOICE Enter your choice (1, 2, 3, or 4): /c1234 /n
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 GOTO DO_DIR
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 GOTO DO_VER
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 GOTO DO_VOL
IF ERRORLEVEL 4 IF NOT ERRORLEVEL 5 GOTO EXIT
GOTO LOOP

REM Perform action based on menu choice
:DO_DIR
DIR
PAUSE
GOTO LOOP

:DO_VER
VER
PAUSE
GOTO LOOP

:DO_VOL
VOL
PAUSE
GOTO LOOP

:EXIT
```

完整的批处理文件含这些命令:

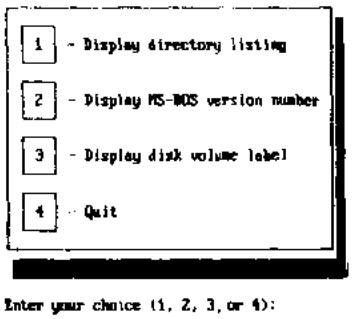
```
ECHO ┌──────────────────────────────────┐
ECHO │  1 - Display directory listing   │
ECHO │                                  │
ECHO │  2 - Display MS-DOS version number
ECHO │                                  │
ECHO │  3 - Display disk volume label   │
ECHO │                                  │
ECHO │  4 - Quit                        │
ECHO └──────────────────────────────────┘

REM Evaluate menu choices
CHOICE Enter your choices (1, 2, 3, or 4): /c1234 /n
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 GOTO DO_DIR
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 GOTO DO_VER
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 GOTO DO_VOL
IF ERRORLEVEL 4 IF NOT ERRORLEVEL 5 GOTO EXIT
GOTO LOOP
```

```
REM Perform action based on menu choice
:DO_DIR
DIR
PAUSE
GOTO LOOP

:DO_VER
VER
PAUSE
GOTO LOOP
:DO_VOL
VOL
PAUSE
GOTO LOOP

:EXIT
```

IBM 扩充字符集可大大增强屏幕显示功能。例如，在上面的批处理文件中再加上几条 ECHO 命令就可以建立如下的菜单：



通过用 IBM 扩充字符值 219 来增加一阴影框就可进一步改善菜单显示：



显然，阴影框的使用使菜单看上去很棒。现在，你已有了一组功能很强的批处理工具。可以用本章的例子来开发专业水准的屏幕显示。

第 9 章

用 DEBUG 程序改进批处理文件

9.1 用 DEBUG 建立程序

标准的批处理文件命令提供了一组功能很强的基本工具，但是，如果你想建立一个需要使用你没有的工具的批处理文件时该怎么办？答案是用 DEBUG 来建立自己的工具。

DEBUG 是用来帮助程序员发现其程序中的错误的实用工具。它还可用于用汇编语言来建立小程序。汇编语言是一组能被计算机直接执行，而无需首先被 MS-DOS 处理的命令。

MS-DOS 成千上万的用户中很少是程序员，而了解怎样用汇编语言编程的就更少了。幸运的是，为建立本章中的实用工具，你不必知道如何编程。对批处理文件中所使用的大多数 DEBUG 程序来说，已有人开发出了实际指令。你只需以合适的次序输入它们即可。每个例子都解释了它的各个步骤，只要仔细研究，就会发现 DEBUG 非常易于使用。

大多数经常开发批处理文件的人都有一组用 DEBUG 建立的实用工具，但这些人中只有极少数人是自己建立实用程序的。大多数人的大多数 DEBUG 程序都是从杂志上和其他出版物上复制过来的。对这种类型的程序不必担心其版权问题，因为它们是已被发表出来的，所以你可以在自己的批处理文件中使用它们。本章我们将建立下列程序：

程　序	目　的
SCRPRINT	从批处理文件中打印屏幕。
REBOOT	从批处理文件中使系统重新启动。
GETYORN	代替 CHOICE 命令来从用户处得到 YES 或 NO。
F1TOF10	将 F1 至 F10 键用作批处理文件的输入。
GETARROW	在菜单及其他批处理文件中使用上下箭。
BELL	发出系统响铃声。
SCANCODE	确定键盘上按下了什么键。
WRITE	替代 ECHO 命令。

此外，DEBUG 还在第 11 章中被用来建立一组确定当前时间和日期的实用程序。

使用 DEBUG 建立程序的方式有两种，我们将在本章中逐一讨论。第一种方法是运行 DEBUG 后，一行一行地输入程序。我们先讨论这种方法，这样就可大致了解 DEBUG 实际上是如何工作的。第二种方法是建立所谓 DEBUG 的含必要指令的文件——不是在 DEBUG 中敲入一条条指令，而是将这些指令敲入一正文文件，然后将该文件重定向至 DEBUG 命令。第二种方法是杂志或其他地方刊登 DEBUG 程序时常常用到的方法。如果你研究了本章中的所有例子，就会知道怎样建立在其他地方发现的任何 DEBUG 程序，而

不论这些程序的作者用的是什么方法。

在和 DEBUG 程序打交道之前，还有一点知识需要知道：十六进制表示。关于这一问题，并不需要了解太多，但至少应该知道 DEBUG 程序用的是与你习惯的计数系统不同的另一系统。十六进制表示不是使用十个数字，而是用十六个：0,1,2,3,4,5,6,7,8,9,A,B,C,D,E 和 F。这意味着值 10H(H 表示十六进制数)等同于 16，因为从 0 数到最右列的 15(或 F)，然后将值 16 表示成左边下一列的 1(十进制中也类似：从 0 数到最右列的 9，然后将值 10 表示成左边下一列的 1)。因此，在十六进制中，A1H 就等于 161(1 是最右列)，A * 16 就等于 10 * 16(A 在左边下一列)。

如果对此有些迷惑，不用着急。只要仔细跟随有关指令，就不会在使用 DEBUG 时有任何麻烦。

我们先来建立一个名为 SCRPRINT.COM 的程序，它打印屏幕显示的当前内容。假定你的批处理文件在屏幕上显示重要的数据，而你想确保用户已保存了这些信息，则在显示了该数据后，可以在你的批处理文件中运行 SCRPRINT 以打印出屏幕内容。

当运行 DEBUG 时，可以在命令行上说明 DEBUG 将要建立的文件名：

```
C:\>DEBUG FILENAME.EXT
```

9.1.1 建立一新文件

首先，运行 DEBUG，并将 SCRPRINT.COM 文件名作为其参数：

```
C:>DEBUG SCRPRINT.COM
```

DEBUG 会显示下列信息：

```
C:\>DEBUG SCRPRINT.COM
File not found

-
```

File not found 表示 SCRPRINT.COM 在磁盘上尚不存在。连字符(-)是 DEBUG 的提示符。

9.1.2 输入打印屏幕程序

敲入命令 A 100，然后按 ENTER 键：

```
C:\>DEBUG SCRPRINT.COM
File not found
-A 100
5AFF:0100
```

A 100 命令告诉 DEBUG 想要在由十六进制地址 100 所指定的特殊位置上输入程序命令。每次 MS-DOS 运行一程序时，MS-DOS 都是从该处开始执行的，所以我们也是在这里放置程序指令。用 DEBUG 创建的所有程序都将从 A 100 开始。

DEBUG 对 A 100 命令的反应是显示一行号，例如是 5AFF：0100。最右边的数 0100 是你在前一行中说明的地址 100(在下面的指令中，程序中表示的行号都是十六进制的。例如，如果每次增加 2，那位前八个行号就是 0100，0102，0104，0106，0108，010A，010C 和 010E)。实际屏幕上 0100 左边显示的数可能与这里的有所不同，因为具体是多少取决于 MS-DOS 是在内存中何处启动 DEBUG 的、所使用的是 MS-DOS 的什么版本以及安装的是什么程序。

为输入第一条命令，敲入 INT 5 命令并按 Enter：

```
C：\>DEBUG SCRPRINT.COM
File not found

-A 100
5AFF：0010 INT 5
5AFF：0102
```

INT 5 命令让计算机打印其屏幕内容。

就这么简单！下面还必须再加上两条指令以告诉 MS-DOS 结束程序。先敲入 MOV AH，4C 并按 Enter。再输入 INT 21 并按 Enter。这样程序就完整了。

9.1.3 保存文件

现在应该告诉 DEBUG 将程序内容保存至磁盘上了。一开始，先按 Enter 键，DEBUG 会如下显示其连字号提示符：

```
C：\>DEBUG SCRPRINT.COM
File not found

-A 100
5AFF：0100 INT 5
5AFF：0102 MOV AH,4C
5AFF：0104 INT 21
5AFF：0106
-
```

接着敲入 R CX 命令，DEBUG 会显示：

```
-R CX
CX 0000
:
```

CX 是 DEBUG 用来存放程序大小的内部寄存器，或叫存储位置。在 DEBUG 将程序写至磁盘前，必须告诉 DEBUG 该程序有多大。程序大小可通过将程序中最后一行上的地址值减去第一行上的地址值而得到：

```
C：\>DEBUG SCRPRINT.COM
```

```
File not found

-A 100
5AFF:0100 INT 5 ←——————— 程序的第一行行号
5AFF:0102 MOV AH,4C
5AFF:0104 INT 21
5AFF:0106 ←——————————— 程序的最后一行行号
-R CX
CX 0000
:
```

在本例中,0106 减去 0100 等于 6,所以程序就是 6 个字节长。在冒号指示符下敲入 6 就将该值放入了 CX 寄存器。当按 Enter 时,DEBUG 就显示如下的命令提示符:

```
C:\>DEBUG SCRPRINT.COM
File not found

-A 100
5AFF:0100 INT 5
5AFF:0102 MOV AH,4C
5AFF:0104 INT 21
5AFF:0106
-R CX
CX 0000
:6
-
```

为了将文件保存在磁盘上,执行 DEBUG 的写文件命令 W:

```
C:\>DEBUG SCRPRINT.COM
File not found

-A 100
5AFF:0100 INT 5
5AFF:0102 MOV AH,4C
5AFF:0104 INT 21
5AFF:0106
-R CX
CX 0000
:6
-W
Writing 00006 bytes
-
```

9.1.4 退出 DEBUG

在 DEBUG 将文件保存至盘上后，可用 DEBUG 的退出命令 Q 来返回至命令提示符：

```
C:\>DEBUG SCRPRINT.COM
File not found

-A 100
5AFF:0100 INT 5
5AFF:0102 MOV AH,4C
5AFF:0104 INT 21
5AFF:0106
-R CX
CX 0000
:6
-W
Writing 00006 bytes
-Q

C:\>
```

9.1.5 使用新程序

如果想要看一下 SCRPRINT.COM 文件的目录清单，MS-DOS 会显示如下：

```
C:\>DIR SCRPRINT.COM
  Volume in drive C is DOS 6 DISK
  Volume Serial Number is 3A2F-18E9
  Directory of C:\BATCH

SCRPRINT COM          6 04-11 93 12:11p
    1 File(s)           6 bytes
                    21155840 bytes free
```

如你所见，文件大小为 6 个字节。

现在就可以从命令提示符上运行 SCRPRINT 程序以打印当前屏幕内容。

9.2 自动进行热启动

如果你的批处理文件修改了系统配置，那么就需要重新启动系统以使这些修改产生作用。要做到这一点，可以按 Ctrl-Alt-Del 键组合，但是建立批处理文件的根本目的就是自动进行各项处理。下面建立的 DEBUG 程序 REBOOT.COM 就如同按了 Ctrl-Alt-Del

那样重新启动系统：

```
C:\> DEBUG REBOOT.COM
File not found

-A 100
584B:0100 MOV AX, 40
584B:0103 MOV DS, AX
584B:0105 MOV AX, 1234
584B:0108 MOV [72], AX
584B:010B JMP FFFF:0
584B:0110
-R CX
CX 0000
:10
-W
Writing 00010 bytes
-Q

C:\>
```

REBOOT.COM 将值 1234 放在内存中，然后用 JMP 命令来跳转到打开计算机时自举计算机的自举程序处。自举程序检查内存地址 4072H 的内容是否是 1234。如果是，程序就完成热启动。否则，就完成如同关上机器后又刚打开那样的冷启动。下面的程序含有解释 REBOOT.COM 处理的注释：

```
; 0100 MOV AX,40          ;First portion of the hexadecimal address
; 0103 MOV DS,AX
; 0105 MOV AX,1234        ;Prepare the value to be inserted
; 0108 MOV [72],AX        ;Move value into the memory address
; 010B JBP FFFF:0         ;Jump to bootstrap program
```

由于地址中的前四位数字可能会因具体系统而不同，所以在这里就未加以显示。程序的输入可以从任何内存地址处进行，但第一条指令必须从 A 100 开始。为输入程序中的注释，需在其前面加上一分号(;)。这样，当从命令提示符下运行 REBOOT.COM 时，计算机就会重新启动。

9.2.1 用 REBOOT 来控制虚拟盘

让我们考虑一个可能会需要让批处理文件重新启动计算机的应用程序。假定你正在将一段内存用作虚拟盘以模拟快速磁盘驱动器。在建立了虚拟盘后，可以用一个盘符(驱动器字母)加上冒号来访问它——如访问硬盘或软盘那样。由于虚盘处于计算机的内存中，所以就不存在硬盘或软盘具有的机械限制；因此，利用虚盘就能更快地访问文件。但是，虚盘只是临时的，当关闭计算机时，其中的内容就丢失了。

根据 MS-DOS 的具体版本，可以用 RAMDRIVE.SYS 或 VDISK.SYS 来创建一虚盘。从 MS-DOS 3.2 开始可以用 CONFIG.SYS 中 DEVICE=来将 RAMDRIVE.SYS 设备驱动程序装入内存，从而建立虚拟盘。例如，下列语句建立了 64KB(缺省大小)的虚盘，

这占常规内存的十分之一：

DEVICE=RAMDRIVE.SYS

为建立一 256KB 的虚盘(占常规内存的五分之二)，可用该句：

DEVICE=RAMDRIVE.SYS 256

注意：为了让 CONFIG.SYS 起作用，必须先重新启动系统。

如果为日常的计算机操作安装了一虚盘，则在运行需要大量内存的应用程序时，还必须移去虚拟盘。为此，就必须编辑 CONFIG. SYS 文件，删去 DEVICE=RAMDRIVE.SYS 语句，然后重新启动计算机。如果想要重新安装虚拟盘，必须再次编辑 CONFIG.SYS，然后再重新启动。

注意：如果用的是 MS-DOS 6 或以后版本，就可以利用多重配置。下列的批处理文件不能和多重配置 CONFIG.SYS 文件一起使用，除非 RAMDRIVE 命令在决定性的块中。

不必不停地重复此编辑—重启动循环，可以用下面的批处理文件 RAMDRIVE.BAT 来自动完成此过程：

```
@ECHO OFF
IF "%1"=="INSTALL" GOTO INSTALL_DISK
IF "%1"=="UNLOAD" GOTO UNLOAD_DISK
GOTO DONE
:INSTALL_DISK
REM Install the RAM drive by adding a DEVICE=RAMDRIVE.SYS
REM entry to CONFIG.SYS. Use %2 to determine the
REM size of the RAM drive. After CONFIG.SYS is
REM updated, reboot.
IF NOT EXIST \CONFIG.SYS GOTO ADD_ENTRY
REM
REM Remove all lines in CONFIG.SYS containing RAMDRIVE.
REM
TYPE \CONFIG.SYS | FIND /V /I "RAMDRIVE" > \CONFIG.NEW
DEL \CONFIG.SYS
REM
REM Append the RAMDRIVE entry to CONFIG.SYS.
REM
:ADD_ENTRY
ECHO DEVICE=RAMDRIVE.SYS %2 >> \CONFIG.NEW
REN \CONFIG.NEW CONFIG.SYS
REBOOT
GOTO DONE
:UNLOAD_DISK
REM Remove the RAM drive by removing the RAMDRIVE entry
REM from CONFIG.SYS and rebooting.
IF NOT EXIST \CONFIG.SYS GOTO DONE
REM
REM Remove all lines in CONFIG.SYS containing RAMDRIVE.
REM
```

```
TYPE \CONFIG.SYS | FIND /V /I "RAMDRIVE" > \CONFIG.NEW
DEL \CONFIG.SYS
REN \CONFIG.NEW CONFIG.SYS
REBOOT
:DONE
```

为安装一个 128KB 的虚盘，可如下运行此批处理文件：

```
C:\>RAMDRIVE INSTALL 128
```

在此例中，该批处理文件安装了一个能存放 128KB 的虚盘。为去掉此虚盘，可如下运行此文件：

```
C:\>RAMDRIVE UNLOAD
```

无论哪种情况，该批处理文件都先编辑 CONFIG. SYS，然后重新启动计算机以使所做的修改发挥作用。

该批处理文件需要调用上一节建立的 REBOOT 程序(或者在当前目录中，或者在搜索路径中)。类似地，ADD_ENTRY 符号后的 ECHO 命令假定 RAMDRIVE. SYS 文件能在根目录中被找到。如果不是这样，就必须在文件名前加上合适的路径——例如，C:\DOS\RAMDRIVE. SYS。

在此批处理文件中以 ADD_ENTRY 标号开始的段中，跟在 REBOOT 命令后的 GOTO DONE 命令一般不应该被执行，因为系统应该重新启动。但是，如果 REBOOT 不存在或未被发现，则 GOTO DONE 就保证此批处理文件的其余部分不会被执行。

9.3 得到 YES 或 NO 回答

在 MS-DOS 6 或以后版本中，可以在批处理文件中使用 CHOICE 命令来从用户那里得到 YES 或 NO 回答。如果用的不是 MS-DOS 6 或以后版本，就可以建立下面的程序 GETYORN. COM 来得到 YES 或 NO 回答：

```
C:\> DEBUG GETYORN.COM
File not found

-A 100
5B10:0100 MOV AH, 08 ; Prepare character service
5B10:0102 INT 21     ; Call the service
5B10:0104 CMP AL, 59 ; Did user press Y?
5B10:0106 JZ 010E    ; If so, jump to line 010E
5B10:0108 CMP AL, 4E ; Did user press N?
5B10:010A JZ 010E    ; If so, jump to line 010E
5B10:010C JMP 0100   ; Loop to get valid character
5B10:010E MOV AH, 4C ; Prepare program termination service
5B10:0110 INT 21     ; Call the service
5B10:0112
-R CX
CX 0000
:12
-W
```

```
REM The COPYIT.BAT batch file is no longer needed.
DEL COPYIT.BAT
```

实际情况是 COPYYN.BAT 创建了一个临时批处理文件用于一次一个地拷贝文件。当不再需要该临时文件 COPYIT.BAT 时，COPYYN.BAT 就删除它。

我们在看一下该批处理文件中的下面三行：

```
ECHO ECHO Do you want to copy %%1? > COPYIT.BAT
ECHO GETYORN >> COPYIT.BAT
ECHO IF ERRORLEVEL 89 COPY %%1 A: >> COPYIT.BAT
```

该批处理文件用 MS-DOS 的输出重定向符(>)来建立名为 COPYIT.BAT 的批处理文件。接着的两条命令用 MS-DOS 的添加重定向符(>>)来将命令加至批处理文件中，注意在%%1 中用了两个百分号。我们说过，每次 MS-DOS 在批处理文件中发现了%1，就用该文件的第一个参数的值来替换它。但是，在本例中，我们却不想让 MS-DOS 来替换某参数；我们只是想让 MS-DOS 将字符%1 写至批处理文件 COPYIT.BAT 中，用两个百分号就可达到此目的，从而允许 COPYIT.BAT 支持批处理文件参数。

COPYYN.BAT 的主要部分是 FOR 循环，该循环将各个文件传给 COPYIT.BAT 批处理文件。每次 FOR 循环调用 COPYIT.BAT 时，该批处理文件就询问是否应该拷贝所说明的文件。接着，该批处理文件就用我们前面建立的 GETYORN.COM 程序来决定回答是什么。如果 Y 被按下了，则该批处理文件就拷贝该文件。如果 N 被按下，则 COPYIT.BAT 就完成执行，跳过 COPY 命令，然后重复 FOR 循环。

可以用下列命令行来运行 COPYYN.BAT：

```
C:\>COPYYN *.*
```

该批处理文件接着就可有选择性地将当前目录中的文件拷贝至 A 盘。

9.4 返回功能键的扫描码

下面的程序 F1TOF10.COM 以一种类似于 GETYORN.COM 程序的方式，返回对应于 F1 至 F10 功能键的键盘扫描码值 59 至 68。在下面的程序清单中，地址的前四位被省略了：

```
C:\> DEBUG F1TOF10.COM
File not found

 A 100
:0100 MOV AH, 08 ; Prepare character service
:0102 INT 21     ; Call the service
:0104 CMP AL, 0  ; Is first portion 0 (is it function key)
:0106 JNZ 0100   ; If it wasn't 0, return to 0100 and try again
:0108 MOV AH, 08 ; Next, prepare service to get next portion
:010A INT 21     ; Call the service
:010C CMP AL, 3B   Compare scan code to 3B (F1)
:010E JL 0100    . If less than 3B, jump to 0100 and try again
:0110 CMP AL, 44 ; Compare scan code to 44 (F10)
```

```
:0112 JG 0100     ; If greater than 44, jump to 0100 and try again
:0114 MOV AH, 4C  ; Prepare program termination service
:0116 INT 21      ; Call the service
:0118
-R CX
CX 0000
:18
-W
Writing 00018 bytes
-Q

C:\>
```

GETYORN 程序对字符作比较，并在用户按了 Y 或 N 时跳至结尾处。F1TOF10 程序则检查字符值是否在功能键的范围内，如果不是，该程序就返回以等待另一字符。一个功能键返回的扫描码值由两部分组成：0 以及跟在它后面标识所按的键的数字。0104 行上的语句检查第一个值是否为 0。如果不是，该程序就立即返回以得到另一字符。只有在第一个值为 0 时，第二个值才测试。下面的行拒绝小于 3B(十进制 59)或大于 44(十进制 68)的值，并使程序尝试下一字符。

可以在批处理文件中使用 F1TOF10，例如显示如下菜单的 DOSMENU.BAT 批处理文件：

F1 - Display directory listing

F2 - Display MS-DOS version number

F3 - Display disk volume label

F4 - Quit

如果按了 F1 键，则该批处理文件显示一目录清单。如果按了 F2 键，则显示当前 MS-DOS 的版本号。如果按了 F3 键，则该批处理文件显示文卷号。该文件重复此过程，直至按了 F4 退出。下面就是 DOSMENU.BAT 的命令：

```
@ECHO OFF
CLS
:LOOP
ECHO F1 - Display directory listing
ECHO F2 - Display MS-DOS version number
ECHO F3 - Display disk volume label
ECHO F4 - Quit
REM Get user response
:GET_KEY
F1TOF10
IF ERRORLEVEL 63 GOTO GET_KEY
IF ERRORLEVEL 62 GOTO DONE
IF ERRORLEVEL 61 IF NOT ERRORLEVEL 62 VOL
IF ERRORLEVEL 60 IF NOT ERRORLEVEL 61 VER
IF ERRORLEVEL 59 IF NOT ERRORLEVEL 60 DIR
GOTO LOOP
:DONE
```

该批处理文件首先显示上述菜单，然后运行 F1TOF10 程序来得到一功能键应答。F1TOF10 要求用户按一个 F1 至 F10 的功能键。该批处理文件对能被按下的键作出限制。如果按下的是 F5 至 F10 的功能键，则该批处理文件循环以得到一正确的键。如果按

下的是F4键(扫描码62),则该批处理文件就结束。注意我们用了嵌套的IF命令来测试一特定的退出状态值。以这种方式使用两条IF命令,该批处理文件就能测试退出状态值是否为61、60或59,而不是仅测试该退出状态值是否大于或等于其中的一个值。

9.5 接受上下箭头或ENTER键

本章中我们将讨论的最后一个程序GETARROW.COM等待用户按上箭头键、下箭头键或ENTER键。该程序对上箭头键返回退出状态值72,对下箭头键返回值80。如果按了ENTER键,该批处理文件返回退出状态值13。该程序在这三个键中的一个被按下之前不会退出。下面说明了如何建立GETARROW.COM程序:

```
C:\> DEBUG GETARROW.COM
File not found

-A 100
:0100 MOV AH, 8
:0102 INT 21
:0104 CMP AL, D
:0106 JZ 11A
:0108 CMP AL, 0
:010A JNZ 100
:010C MOV AH, 8
:010E INT 21
:0110 CMP AL, 48
:0112 JZ 11A
:0114 CMP AL, 50
:0116 JZ 11A
:0118 JMP 100
:011A MOV AH, 4C
:011C INT 21
:011E
-R CX
CX 0000
:1E
-W
Writing 0001E bytes
-Q

C:\>
```

该程序依靠MS-DOS字符服务程序来得到所按的最后一个键。同GETYORN.COM一样,该程序检查所按键是否与特定扫描码相匹配。如果所按键的扫描码与三个键中的任何一个相匹配,则该程序就结束并返回扫描码至ERRORLEVEL。如果没有匹配扫描码,该程序就继续以等待另一字符(0118行)。第一个比较是对ENTER键(扫描码为十进制的13或十六进制的D),这在行0114上(CMP AL,D)。如果该键是ENTER键,则程序就跳至结尾(行0106)。由于箭头键的扫描码有两部分(如同前面程序中的功能键),所以接着的测试就检查第一部分是否为0(行0108)。如果该值不为0,则程序就跳回行0100(行010A);如果该值为0,该程序接着继续检查该扫描码的第二部分是否表示上箭头(扫描码

值 72 或 48H；行 0110)或表示下箭头(扫描码值 80 或 50H；0114)。如果是两种情况之一，该程序就跳至结尾(行 0112 和 0116)。否则，该程序返回至 0100 行。

注意：这是第一个其字节大小是明显十六进制数(1E)的程序。实际上，字节大小始终是以十六进制表示的。当保存 REBOOT.COM 程序时，会显示它正在将 10 个字节写至盘上，如果对该文件显示一下目录清单，则会看见它实际上有 16 个字节。十六进制的 10 等于十进制的 16。GETARROW.COM 程序在目录显示时实际上是 31 个字节。

该程序可用在像 ARROW.BAT 这样的批处理文件中，该文件显示一个类似于上一个批处理文件的菜单：

```
Display directory
Display MS-DOS version
Display volume label
Quit
```

在此例中，该批处理文件用 ANSI.SYS 设备驱动程序来加亮显示当前的菜单选择。随着上下箭头键被按下，当前选择就被改变。当按了 ENTER 键，该批处理文件就执行对应于当前选择的命令：

```
@ECHO OFF
SET CURRENT=DIR
:LOOP
CLS
IF %CURRENT%==DIR ECHO ←[1mDisplay directory
IF NOT %CURRENT%==DIR ECHO ←[0mDisplay directory
IF %CURRENT%==VER ECHO ←[1mDisplay MS-DOS version
IF NOT %CURRENT%==VER ECHO ←[0mDisplay MS-DOS version
IF %CURRENT%==VOL ECHO ←[1mDisplay volume label
IF NOT %CURRENT%==VOL ECHO ←[0mDisplay volume label
IF %CURRENT%==QUIT ECHO [1mQuit
IF NOT %CURRENT%==QUIT ECHO ←[0mQuit
GETARROW
IF ERRORLEVEL 80 GOTO DOWN_ARROW
IF ERRORLEVEL 72 GOTO UP_ARROW
IF %CURRENT%==DIR DIR
IF %CURRENT%=VER VER
IF %CURRENT%==VOL VOL
IF %CURRENT%==QUIT GOTO DONE
PAUSE
GOTO LOOP

:UP_ARROW
IF NOT %CURRENT%==DIR GOTO UP_VER
SET CURRENT=QUIT
GOTO LOOP
:UP_VER
IF NOT %CURRENT%==VER GOTO UP_VOL
SET CURRENT=DIR
GOTO LOOP
:UP_VOL
IF NOT %CURRENT%==VOL GOTO UP_QUIT
```

```
SET CURRENT=VER
GOTO LOOP
:UP_QUIT
SET CURRENT=VOL
GOTO LOOP

:DOWN_ARROW
IF NOT %CURRENT%==DIR GOTO DOWN_VER
SET CURRENT=VER
GOTO LOOP
:DOWN_VER
IF NOT %CURRENT%==VER GOTO DOWN_VOL
SET CURRENT=VOL
GOTO LOOP
:DOWN_VOL
IF NOT %CURRENT%==VOL GOTO DOWN_QUIT
SET CURRENT=QUIT
GOTO LOOP
:DOWN_QUIT
SET CURRENT=DIR
GOTO LOOP

:DONE
ECHO ←[0m
SET CURRENT=
```

在此批处理文件中说明的菜单操作功能通常仅限于为用 PASCAL 或 C 语言编写的程序所有。该批处理文件用%CURRENT%有名参数来记录当前选择。当该批处理文件开始执行时，它用 SET 命令来将 DIR 的值赋给有名参数。在 ARROW.BAT 显示的菜单中，DIR 是第一个选择。接着，该批处理文件用 IF 命令来确定当前选择，并用 ANSI.SYS 转义序列←[1m 来以粗体显示当前选择。该批处理文件用转义序列←[0m 来以通常属性显示所有其他选项。有关转义序列的更多信息可参见第八章。

接着，该处理文件用 GETARROW.COM 程序来得到上、下箭头键或 ENTER 键。如果按了 ENTER 键，则该批处理文件应用%CURRENT%有名参数来确定执行哪一条命令。如果按下了上箭头键或下箭头键，则该批处理文件就确定新的当前选择并重新显示该菜单。

当 Quit 选项被选择时，该批处理文件重新将属性设置成通常属性，并从 MS-DOS 环境中移去%CURRENT%有名参数。

必须承认，该批处理文件的菜单选项相当简单；但是，ARROW.BAT 说明了用 MS-DOS 批处理文件能够完成的编程量。

注意：可以看出，ARROW.BAT 重复地调用 GETARROW.COM 来确定哪个键被按下了。通过将 GETARROW.COM 存在一虚盘上，然后从虚盘上调用该文件，就可以使该批处理文件的执行速度大大加快。

9.6 用 DEBUG 命令文件来建立程序

至今，我们都是交互式地在 DEBUG 中通过敲入指令而建立程序的。许多人（以及计算机杂志上的例子）都是用 DEBUG 命令文件来建立程序的。一般来说，一个 DEBUG 原本文件只是这样一个文件，它包含在 DEBUG 中交互式地建立程序时所需敲入的内容。通常，DEBUG 命令文件的扩展名是 SCR。

可以用 EDIT 或 EDLIN 来建立 DEBUG 命令文件。例如，下面的 DEBUG 命令文件 BELL.SCR 含有用于建立一个使计算机内部扬声器发声的程序 BELL.COM 的指令：

```
N BELL.COM
A 100
MOV DL,7
MOV AH,2
INT 21
MOV AH,4C
INT 21
R CX
A
W
Q
```

在建立了一个含这些命令的名为 BELL.SCR 的正文文件后，就可将其内容重定向至 DEBUG 命令：

```
C:\>DEBUG < BELL.SCR
```

DEBUG 又进一步创建 BELL.COM 程序，该程序可在必要时被一批处理文件激活以引起用户的注意。命令文件中的每一行都执行 DEBUG 中的一条命令。在 BELL.COM 被建立时，屏幕上显示实际发生的步骤。现在可以在批处理文件中调用 BELL.COM：

```
@ECHO OFF
BELL
BELL
ECHO Do you want to delete %1?
REM Remainder of the batch file commands here.
```

9.7 替代 CHOICE 命令

在 MS-DOS 6 或以后版本中，CHOICE 命令可以方便地给出提示以等待键盘响应，然后再确定按下了哪个键。如果用的不是 MS-DOS 6 或以后版本，那么就可以用 DEBUG 的命令文件 SCANCODE.SCR 来建立 SCANCODE.COM 程序以返回所按键的扫描码。

```
N SCANCODE.COM
A 100
XOR AH, AH
INT 16
CMP AH, 0
JE 100
MOV AL, AH
MOV AH, 4C
INT 21

R CX
10
W
Q
```

创建该命令文件，然后将其内容重定向至 DEBUG：

```
C:\>DEBUG < SCANCODE.SCR
```

接着就可以在任何批处理文件中调用 SCANCODE.COM，该程序将返回对应于所按的键的退出状态值。图 9-1 说明了各个可能的扫描码值。

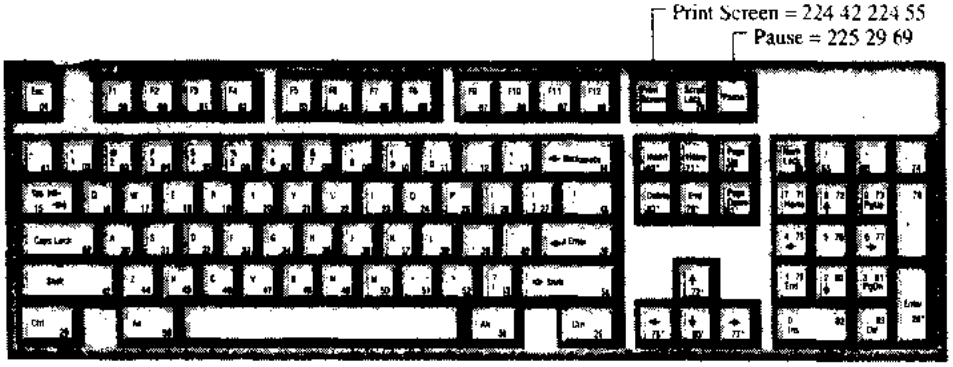


图 9-1　说明所有可能的扫描码的键盘

* = 扫描码前跟 224

9.8　用 WRITE 来代替 ECHO

随着批处理文件复杂性的增加，有时可能会想让批处理文件不必将光标移至下一行（像 ECHO 那样）就可在屏幕上写信息。下面的 DEBUG 命令文件 WRITE.SCR 建立一名为 WRITE.COM 的文件，用以不带回车换行字符地显示一行输出。

```
N WRITE.COM
A 100
MOV BX, 80
INC BX
```

```
CMP BYTE [BX], D
JNE 103
MOV BYTE [BX], 24
MOV AH, 9
MOV DX, 82
CMP DX, BX
JG 117
INT 21
MOV AH, 4C
INT 21

R CX
1B
W
Q
```

同样，如下使用 DEBUG 就可建立程序文件：

```
C:\>DEBUG < WRITE.SCR
```

这样，就可以在批处理文件中使用 WRITE.COM 和 SCANCODE.COM 来显示一提示符并等待键被按下：

```
WRITE Enter choice:
SCANCODE
```

当该批处理文件执行时，屏幕上显示提示符，但并不将光标移至下一行：

```
Enter choice:_
```

在第 10 章中，我们将再次使用 WRITE.COM 程序来建立临时批处理文件。

第 10 章

建立批处理文件库

10.1 设置屏幕颜色

我们已经讨论了几个简单的可用于日常工作的批处理文件。在本章中，我们将再建立六个通用的批处理文件。在建立这些文件时，你可以任意修改以满足自己的需要。

第一个批处理文件 COLORS.BAT 可允许使用键盘的箭头键来循环显示为 ANSI.SYS 设备驱动程序所支持的背景屏幕颜色（为了成功地执行此批处理文件，CONFIG.SYS 必须将 ANSI.SYS 设备驱动程序装入内存）。每次按上、下箭头时，该批处理文件就改变屏幕颜色。当按 Enter 键时，所选择的颜色就成为屏幕的颜色而一直保留下去。

```
@ECHO OFF
REM Allow the user to cycle through the background
REM screen colors that are available, until the user
REM presses Enter to select a color.
SET CURRENT=BLACK
:LOOP
IF %CURRENT%==BLACK ECHO ←[40m
IF %CURRENT%==RED ECHO ←[41m
IF %CURRENT%==GREEN ECHO ←[42m
IF %CURRENT%==YELLOW ECHO ←[43m
IF %CURRENT%==BLUE ECHO ←[44m
IF %CURRENT%==MAGENTA ECHO ←[45m
IF %CURRENT%==CYAN ECHO ←[46m
IF %CURRENT%==WHITE ECHO ←[47m
CLS
ECHO Press Enter to select the current color
ECHO Use Up arrow and Down arrow to change colors
GETARROW

IF ERRORLEVEL 80 GOTO DOWN_ARROW
IF ERRORLEVEL 72 GOTO UP_ARROW
IF ERRORLEVEL 13 GOTO DONE

:DOWN_ARROW
IF NOT %CURRENT%==BLACK GOTO DN_RED
SET CURRENT=RED
GOTO LOOP
:DN_RED
IF NOT %CURRENT%==RED GOTO DN_GREEN
SET CURRENT=GREEN
GOTO LOOP
:DN_GREEN
IF NOT %CURRENT%==GREEN GOTO DN_YELLOW
SET CURRENT=YELLOW
GOTO LOOP
:DN_YELLOW
```

SYS 转义序列回显在屏幕上。该文件使用了 GETARROW.COM 程序(在第九章用 DEBUG 建立的)。首先,%CURRENT%的值被置为黑色。然后,该批处理文件用%CURRENT%的值来设置背景颜色。接着,就可选择是按上箭头、下箭头还是 ENTER。

如果按的是一个箭头键,该批处理文件就确定它是上箭头还是下箭头。如果是上箭头,该文件就跳至 UP_ARROW 标号处开始寻找当前屏幕颜色。当该批处理文件找到当前颜色时,序列中下一颜色的名字(白、青、紫、蓝、绿、红或黑)就被赋给%CURRENT%。然后该批处理文件返回至文件顶部的 LOOP 标号处,并根据%CURRENT%的新值而重新设置背景色。

注意 IF 语句是以与颜色的相反次序而组织排列的,并且如果当前颜色不匹配,就跳至下一条测试。下箭头键被以 DOWN_ARROW 标号开始的那组语句以类似的方式加以处理。当按 Enter 键时,该批处理文件的执行就结束了。

10.2 防止硬盘被格式化

许多硬盘都因为初学者用户执行 FORMAT 命令而被破坏。为防止意外格式化,不少人将 FORMAT.COM 文件从硬盘删除。但这并不能消除意外格式化硬盘的可能性,而且还使一条有用的 MS-DOS 命令被删去了。因为大多数人都是将其工作备份在软盘上的,所以很可能需要用 FORMAT 命令来格式化软盘。

下面可能是一种解决方案。将 FORMAT.COM 改名成 FMAT.COM,然后建立一个含下列命令的名为 FORMAT.BAT 的批处理文件:

```
@ECHO OFF
ECHO To prevent inadvertent formatting of the hard
ECHO disk, this batch file lets you format only
ECHO floppy disks in drive A or B. Specify the disk
ECHO drive to use in your command line.
ECHO Example: FORMAT A:
CHOICE Do you want to continue
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 GOTO CONTINUE_FORMAT
IF ERRORLEVEL 2 GOTO DONE

:CONTINUE_FORMAT
IF "%1"=="" GOTO NO_PARAMS
IF "%1"=="A:" GOTO VALID_DRIVE
IF "%1"=="a:" GOTO VALID_DRIVE
IF "%1"=="B:" GOTO VALID_DRIVE
IF "%1"=="b:" GOTO VALID_DRIVE
GOTO INVALID_DRIVE

:NO_PARAMS
ECHO You must specify the disk drive to format
ECHO in your command line.
GOTO DONE

:VALID_DRIVE
FMAT %1 %2 %3 %4 %5 %6 %7 %8 %9
GOTO DONE
```

```
:INVALID_DRIVE
ECHO This batch file formats only disks in
ECHO drive A or B. The batch file considers
ECHO the disk drive %1 invalid.

:DONE
```

该批处理文件只允许对 A 或 B 驱动器中的软盘进行格式化。当运行文件时，MS-DOS 显示下列信息：

```
To prevent inadvertent formatting of the hard
disk,this batch file lets you format only
floppy disks in drive A or B. Specify the disk
drive to use in your command line.
Example: FORMAT A:
Do you want to continue [Y,N]?
```

如果按 Y，则该批处理文件首先检查你是否肯定说明了一个驱动器。如果没有，就显示出错信息：

```
You must specify the disk drive to format
in your command line.
```

该批处理文件接着就停下来。如果说明了一个驱动器，它就检查是否是一正确的驱动器（有必要对大小写都进行检查以使该批处理文件对用户友好）。如果%1 参数含一正确的驱动器字母，则该批处理文件跳至 VALID-DRIVE 标号，并用 FMAT.COM 来格式化磁盘；否则，该批处理文件就转移至 INVALID-DRIVE 标号，并显示一出错信息。

注意：在某些情况下，FORMAT 命令行要求有诸如/S 或/4 这样的参数。通过使用%2 至%9 的参数，该批处理文件能够支持这些新的命令行参数。

10.3 得到字符串输入

在第九章使用 DEBUG 建立了若干程序可让户按 Y 或 N 键以说明 Yes 或 No 的应答，或者按一特定的功能键，或者按上、下箭头键。但是，有时候批处理文件可能需要得到若干字符而不是单个键，如一文件名或口令。

下面的批处理文件 GETPASS.BAT 提示输入一口令，然后将此口令与预定义的口令 MANAGEMENT 相比较。如果完全相同，则该批处理文件就继续执行；否则，就显示一出错信息并停止处理。当然，对许多批处理文件应用程序来说，可能并不需要一口令，但是 GETPASS.BAT 说明了如何在批处理文件中得到用户输入。例如，可以使用同样的方法来将一文件名从用户那里传给另一命令。

```
@ECHO OFF
IF NOT EXIST PASSWORD.SET GOTO NO_FILE
ECHO Enter your password, press F6, and then
ECHO press Enter.
COPY CON PASSWORD.DAT > NUL
COPY PASSWORD.SET+PASSWORD.DAT PASSWORD.BAT > NUL
CALL PASSWORD
IF NOT "%PASSWORD%"=="MANAGEMENT" GOTO BAD_PASSWORD
ECHO Valid password; ready to process payroll.
REM Insert protected commands here
GOTO DONE

:NO_FILE
ECHO This batch file requires the file PASSWORD.SET,
ECHO which contains SET PASSWORD=^Z
GOTO DONE

:BAD_PASSWORD
ECHO Invalid password

:DONE
SET PASSWORD=
DEL PASSWORD.BAT > NUL
DEL PASSWORD.DAT > NUL
```

该批处理文件依赖于一个可从命令提示符下建立的名为PASSWORD.SET的文件。批处理文件首先检查该文件是否存在。如果不存在,就停止处理。为建立PASSWORD.SET,执行下列命令:

```
C:\>COPY CON PASSWORD.SET
```

然后敲入SET PASSWORD=,但不要敲Enter。在等号后立即按功能键F6,然后再按Enter键。

这样,MS-DOS就建立了文件PASSWORD.SET。建立PASSWORD.SET的步骤产生了下列屏幕显示:

```
C:\>COPY CON PASSWORD.SET
SET PASSWORD=^Z
        1 File(s) copied

C:\>
```

当运行GETPASS.BAT文件时,会出现要求输入口令的提示:

```
Enter your password,press F6,and then
press Enter.
```

该批处理文件建立一个名为PASSWORD.BAT的文件,其内容由键盘输入。此处理由下面的命令完成:

```
COPY CON PASSWORD.DAT > NUL
```

该命令行让MS-DOS将输入从键盘拷贝至PASSWORD.DAT文件中,直至按F6

说明文件结束。大于符号(>)将该信息重定向至 NUL 设备：

```
1 File(s) copied
```

该信息就不会在屏幕上出现。

接着，该命令通过将 PASSWORD. BAT 的内容添加至 PASSWORD. SET 文件而建立批处理文件 PASSWORD. BAT：

```
COPY PASSWORD. SET+PASSWORD. DAT PASSWORD. BAT > NUL
```

假定你敲入了命令 MANAGEMENT，那么文件 PASSWORD. BAT 的内容就是：

```
SET PASSWORD=MANAGEMENT
```

GETPASS. BAT 接着就运行新建立的 PASSWORD. BAT 文件以在 MS-DOS 环境中建立%PASSWORD%有名参数。GETPASS. BAT 批处理文件然后就能用 IF 命令将该口令与 MANAGEMENT 相比较。如果不同，该批处理文件就显示出错信息并停止处理。如果口令相同，该批处理文件就执行其被保护的命令。注意，在完成执行前，该批处理文件将%PASSWORD%有名参数从环境中删去，同样被删去的还有 PASSWORD. BAT 和 PASSWORD. DAT 文件。为保护批处理文件中出现的口令，可用 ATTRIB 命令来将 GETPASS. BAT 隐藏起来。

10. 3. 1 用 WRITE. COM 来改进 PASSWORD. BAT

GETPASS. BAT 的一个不足之处是需要你建立含下列内容的 PASSWORD. SET 文件：

```
SET PASSWORD=
```

理想的情况是该批处理文件不应要求这样的文件的存在，而是应该自己建立该文件，并在不再需要它时删除该文件。由于该文件只是在使用批处理文件时才存在，所以并不占用不必要的磁盘空间。为了不必先建立 PASSWORD. SET 就建立 PASSWORD. BAT 文件，可以使用第九章建立的 WRITE. COM 来修改 GETPASS 批处理文件：

```
@ECHO OFF
WRITE SET PASSWORD=> PASSWORD.BAT
ECHO Enter your password, press F6, and then
ECHO press Enter.
COPY PASSWORD.BAT+CON PASSWORD.BAT > NUL
CALL PASSWORD
IF NOT "%PASSWORD%"=="MANAGEMENT" GOTO BAD_PASSWORD
ECHO Valid password; ready to process payroll.
REM Insert protected commands here
GOTO DONE

:BAD_PASSWORD
ECHO Invalid password

:DONE
SET PASSWORD=
DEL PASSWORD.BAT
```

WRITE.COM 程序将正文 SET PASSWORD=写至 PASSWORD.BAT 文件中。接着 COPY 命令你敲入的字符添加至该文件中。例如，如果你敲入 MANAGEMENT 并按 F6，则该文件含命令 SETPASSWORD=MANAGEMENT。当用 CALL 命令执行该批处理文件时，MS-DOS 就建立所需要的环境值。

10.4 简化系统备份操作

MS-DOS 6 及以后版本提供了 Microsoft 的 Backup 程序，这是一个功能很强的菜单驱动的实用程序。可以在命令提示符下敲入 MSBACKUP 命令来运行该程序。如果用的不是 MS-DOS 6 及以后版本，那么可以建立批处理文件来帮助进行日常或每月工作的自动备份。这些批处理文件使用 5.0 以前版本的 BACKUP 命令。我们将讨论的下一个批处理文件 FILEBU.BAT 就可帮助进行磁盘备份操作。当运行此批处理文件时，MS-DOS 显示：

```
F1 - Monthly Backup
F2 - Daily/Incremental Backup
F3  Specific Files Backup
F4  Quit
```

如果按 F1 键，该批处理文件就进行全面备份——备份盘上的每个文件。取决于具体文件数目的多少，整月备份可能会很费时间。

如果按 F2 键，该批处理文件就进行增值备份——只备份自上次全面备份后才被创建或修改了的文件。增值备份比全面备份会快得多，但是，必须保存从一次全面备份至另一次全面备份之间的所有增值备份盘，以保证对每个文件都有备份。

如果按 F3 键，该批处理文件就提示你说明要备份哪些文件，包括目标盘驱动器。例如，为将 DOS 目录中的所有文件备份至 A 驱动器的软盘中，可输入：

```
C:\DOS A:
```

接着按 F6 键，后跟 Enter。该批处理文件就开始工作，只备份所说明的文件。

如果按 F4 键，该批处理文件就停止处理。

为了方便理解，我们将此批处理文件分作四个不同的批处理文件来讨论：

FILEBU.BAT、MONTHLY.BAT、DAILY.BAT 和 SPECIFIC.BAT。FILEBU.BAT 文件含下列命令：

```
@ECHO OFF
REM Allow the user to perform a monthly, daily, or
REM specific file backup.
CLS
REM Loop until the user presses the F4 key to quit.
:LOOP
ECHO F1 - Monthly Backup
ECHO F2   Daily/Incremental Backup
ECHO F3 - Specific Files Backup
ECHO F4 - Quit
```

```
REM Get user response.
:GET_KEY
F1TOF10
REM If user presses a key other than F1 through F4, get
REM another key.
IF ERRORLEVEL 63 GOTO GET_KEY
IF ERRORLEVEL 62 GOTO DONE
IF ERRORLEVEL 61 IF NOT ERRORLEVEL 62 CALL SPECIFIC
IF ERRORLEVEL 60 IF NOT ERRORLEVEL 61 CALL DAILY
IF ERRORLEVEL 59 IF NOT ERRORLEVEL 60 CALL MONTHLY
GOTO LOOP
:DONE
```

可以看出，该批处理文件显示主菜单，然后用我们在第九章中建立的 F1TOF10 程序来从用户处得到一功能键。取决于所按的功能键，该批处理文件或调用 MONTHLY.BAT 用于全面备份，或调用 DAYLI.BAT 用于增值备份，或调用 SPECIFIC.BAT 用于对特定文件的备份。该批处理文件在按 F4 键退出之前继续循环并重复此过程。

MONTHLY.BAT 批处理文件完成一次全面备份。当选择此选项时，MS-DOS 显示：

```
The monthly backup copies all the files on
your hard disk to a floppy disk.

Depending on the number of files on your disk,
the monthly backup can be quite time-consuming.
Do you wish to continue [Y,N]?
```

该信息只是解释按月备份的过程，并让你继续或取消此操作。如果回答 Y，则 BACKUP 命令就将整个盘备份至 A 驱动器的软盘中。MONTHLY.BAT 含下列命令：

```
REM Perform a complete backup of the hard disk.
ECHO The monthly backup copies all the files on
ECHO your hard disk to a floppy disk.
ECHO.
ECHO Depending on the number of files on your disk,
ECHO the monthly backup can be quite time-consuming.
CHOICE Do you wish to continue
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 GOTO BACKUP
IF ERRORLEVEL 2 GOTO RETURN
:BACKUP
ECHO Be sure that you label each disk with the
ECHO current date, the words MONTHLY BACKUP, and
ECHO your initials.
BACKUP C:\*.* A: /S
:RETURN
CLS
```

注意：该批处理文件使用了 CHOICE 命令。如果用的是 MS-DOS 5.0 或更早的版本，则需要使用 ECHO 命令、GETYORN.COM 程序（第 9 章中建立），以及 IF 语句中的合适值。

DAILY.BAT 批处理文件非常类似于 MONTHLY.BAT 批处理文件：

```
REM Perform a backup of each file on the disk changed
REM or created since the last full backup.

ECHO The daily backup copies all the files on
ECHO your hard disk that have been changed or created
ECHO since the last complete backup to floppy disk.
ECHO.
CHOICE Do you wish to continue
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 GOTO BACKUP
IF ERRORLEVEL 2 GOTO RETURN
:BACKUP
ECHO Be sure that you label each disk with the
ECHO current date, the words DAILY BACKUP, and
ECHO your initials. Use the disk from the previous
ECHO daily backup until the floppy fills.
BACKUP C:\*.* A: /S /A /M
:RETURN
CLS
```

可以看出，如果用户回答 Y 来继续此批处理文件，则 BACKUP 命令就对硬盘所有改变了的文件进行增值备份。

在这四个备份批处理文件中，SPECIFIC.BAT 最为有趣。该文件可让你说明要备份的文件名。下面的清单说明了 SPECIFIC.BAT 是如何依赖于可在命令提示符下建立的名为 BACKUP.FMT 的文件的：

```
REM Back up one or more user-specified files to disk.
REM Prompt the user to enter the filename to back up.
REM Copy the names to the BACKUP.DAT file. Using the
REM BACKUP.FMT file, create a BACKUP command in the
REM BACKIT.BAT batch file.
IF NOT EXIST BACKUP.FMT GOTO NO_FORMAT
ECHO The specific-file option allows you to back up
ECHO one or more files.
CHOICE Do you want to continue
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 GOTO BACKUP
IF ERRORLEVEL 2 GOTO RETURN
NO_FORMAT:
ECHO The specific-file backup requires the file
ECHO BACKUP.FMT, which contains BACKUP ^Z
PAUSE
GOTO RETURN
:BACKUP
ECHO Type the drive, path, and name of the file to back up.
ECHO identify the target drive, press the F6 key,
ECHO and then press Enter.
ECHO Example: *.DAT A: ^Z [Enter]
COPY CON BACKUP.DAT > NUL
COPY BACKUP.FMT+BACKUP.DAT BACKIT.BAT
CALL BACKIT
DEL BACKUP.DAT
DEL BACKIT.BAT
:RETURN
CLS
```

可以用 COPY CON BACKUP.FMT 命令来开始进行特定文件的备份：

```
C:\>COPY CON BACKUP.FMT
```

敲入 BACKUP，后跟一空格，再按 F6 以标识文件尾：

```
C:\>COPY CON BACKUP.FMT
BACKUP ^Z
```

当按 Enter 时，MS-DOS 就建立了此文件：

```
C:\>COPY CON BACKUP.FMT
BACKUP ^Z
    1 File(s) copied

C:\>
```

如果 BACKUP.FMT 不存在，该批处理文件就显示出错信息并返回至主菜单。如果 BAKCUP.FMT 确实存在，则当按 Y 以继续时，该批处理文件提示你敲入要备份的文件名和目标盘驱动器：

```
Type the drive,path and name of the file to back up,
identify the target drive,press the F6 key
and then press Enter
Example: *.DAT A:^Z [Enter]
```

该批处理文件保存在 BACKUP.DAT 文件中敲入的信息。下面的命令将其输出重定向至 NUL 设备：

```
COPY CON BACKUP.DAT>NUL
```

这样就消除了信息 1 File(s) copied 的显示。

接着的命令将 BAKCUP.DAT 文件的内容添加至 BAKCUP.FMT 文件中(前者含想要备份的文件名，后者含 BACKUP 命令)：

```
COPY BACKUP.FMT+BACKUP.DAT BACKIT.BAT
```

所形成的文件就是名为 BACKIT.BAT 的批处理文件，这是接着将被运行的文件。假定敲入了 C:\DOS 和 A:，BACKIT.BAT 文件就含这条命令：

```
BAKCUP C:\DOS A:
```

该命令只备份说明的目录。当 BACKUP 命令执行完后，该批处理文件就删除 BACKUP.DAT 和 BACKIT.BAT 文件。

如本章前面所示，可以使用第九章建立的 WRITE 命令，通过在批处理文件中将 BACKUP 命令加至 BACKIT.BAT 中来取消对 BACKUP.FMT 文件的需要：

```
WRITE BACKUP >> BACKIT.BAT
```

```
ECHO Type the drive,path and name of the file to back up
ECHO press the F6 key,and then press Enter
ECHO Example: *.DAT A:^Z [Enter]
COPY BACKIT.BAT+CON BACKIT.BAT > NUL
CALL BACKIT
DEL BACKIT.BAT
```

这组批处理文件说明了并发有用的实用工具是多么容易。为改进这些批处理文件，你可能会考虑使用 ANSI.SYS 设备驱动程序来增加颜色（见第 8 章）。从 MS-DOS 3.3 版开始，还可以在 BACKUP 命令中加上/L 开关以命令 BACKUP 建立一记录文件。此外，还可以考虑加上使用 IF ERRORLEVEL 的退出状态检查，以在 BACKUP 失败时显示出错信息。

10.5 建立临时批处理文件

有时一个批处理文件会需要激活第二个批处理文件以完成某些简单的操作。例如，下面的批处理文件 ASK_DEL.BAT 就激活一个名为 ASK_EM.BAT 的用来确定你是否想要删除特定文件的批处理文件：

```
@ECHO OFF
FOR %%I IN (%1) DO CALL ASK_EM %%I
```

批处理文件 ASK_EM.BAT 含下列命令：

```
CHOICE Do you want to delete %1
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 DEL %1
```

利用此批处理文件，可以用通配符说明一组文件，然后该批处理文件在删除每一个文件之前给出提示。这是带/P 开关的 DEL 命令的批处理文件版本。在此例中，一个相当简单的任务要求两个批处理文件。在第九章中说过，小批处理文件会占去大量的磁盘空间。不必将这两个批处理文件始终存在磁盘上，而是可以让 ASK_DEL.BAT 批处理文件在需要时再创建 ASK_EM.BAT 文件：

```
@ECHO OFF
ECHO CHOICE Do you want to delete %%1 > ASK_EM.BAT
ECHO IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 DEL %%1 >> ASK_EM.BAT
FOR %%1 IN (%1) DO CALL ASK_EM %%I
DEL ASK_EM.BAT
```

ASK_DEL.BAT 文件通过重定向两个 ECHO 命令的输出而建立了有两行内容的 ASK_EM.BAT 批处理文件。在 FOR 命令执行完后，该批处理文件就删除 ASK_EM.BAT。这样，ASK_EM.BAT 批处理文件只是在需要时才在磁盘上存在。通过建立这种临时批处理文件，就节省了磁盘空间并简化了批处理，因为你只需维护较少的文件。

10.6 一个快速日历

本书中一直在用 ECHO 命令来为用户显示屏幕上的信息。实际上仅仅用 ECHO 就能建立各种批处理文件。例如，下面的批处理文件 JUNE_94.BAT 使用 ECHO 来建立了一个显示 1994 年 6 月的日历的批处理文件：

```
@ECHO OFF
ECHO  S   M   T   W   T   F   S
ECHO                  1   2   3   4
ECHO  5   6   7   8   9  10  11
ECHO 12  13  14  15  16  17  18
ECHO 19  20  21  22  23  24  25
ECHO 26  27  28  29  30
```

当运行此批处理文件时，屏幕上会显示：

```
C:\>JUNE_94
 S   M   T   W   T   F   S
                 1   2   3   4
 5   6   7   8   9  10  11
12  13  14  15  16  17  18
19  20  21  22  23  24  25
26  27  28  29  30
```

第 11 章将说明如何用不同的日期和时间命令来使批处理文件功能更加强大。

第 11 章

使批处理文件知道日期和时间

11.1 确定是星期几

我们已经讨论了所有的 MS-DOS 批处理文件命令，知道了怎样用 DEBUG 建立程序，并已建立了几个复杂的批处理文件。本章中我们讨论几个有趣的可以从 AUTOEXEC.BAT 中运行的批处理文件。所有这些文件都用到了可用 DEBUG 建立的简单程序。

第一个批处理文件 GREETING.BAT 用了 WEEKDAY.COM 程序来确定当前日子是星期几，然后就显示对应于该天的信息。下面就是此文件的内容：

```
@ECHO OFF
REM Determine the day of the week usirg the program
REM WEEKDAY.COM. The program returns a status value
REM that you can check by using the IF ERRORLEVEL
REM command. Sunday has the value 0, Monday has the
REM value 1, and so on. Saturday has the value 6.
REM After the batch file determines the day, it
REM displays an appropriate message.
WEEKDAY
IF ERRORLEVEL 0 IF NOT ERRORLEVEL 1 GOTO SUNDAY
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 GOTO MONDAY
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 GOTO TUESDAY
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 GOTO WEDNESDAY
IF ERRORLEVEL 4 IF NOT ERRORLEVEL 5 GOTO THURSDAY
IF ERRORLEVEL 5 IF NOT ERRORLEVEL 6 GOTO FRIDAY
IF ERRORLEVEL 6 GOTO SATURDAY

:SUNDAY
ECHO It's Sunday - Relax and watch football.
GOTO DONE

:MONDAY
ECHO It's Monday - Get ready for a new week.
GOTO DONE

:TUESDAY
ECHO It's Tuesday - Week is just beginning.
GOTO DONE

:WEDNESDAY
ECHO It's Wednesday - Hump day!
GOTO DONE

:THURSDAY
ECHO It's Thursday - Almost there!
GOTO DONE
```

```
:FRIDAY
ECHO It's finally Friday - Happy Hour!
GOTO DONE

:SATURDAY
ECHO It's Saturday - You should be in bed.

:DONE
```

如果你有一个重要的约会或会议，那么就可以修改 ECHO 语句以在那天或前一天提醒你。

GREETING.BAT 要调用一个名为 WEEKDAY.COM 的程序，该程序用 MS-DOS 的 Get Date(获取日期)服务程序来确定是星期几。它对星期天返回值为 0 的退出状态值，星期一则是 1，以此类推。可以用 DEBUG 命令文件 WEEKDAY.SCR 来建立 WEEKDAY.COM：

```
N WEEKDAY.COM
A 100
MOV AH,2A
INT 21
MOV AH,4C
INT 21

R CX
8
W
Q
```

将以上内容存入一名为 WEEKDAY.SCR 的正文文件中。然后将该文件内容的输出重定向至 DEBUG：

```
C:\>DEBUG < WEEKDAY.SCR
```

11.2 确定月份

以类似的方式，下面的批处理文件 SCHEDULE.BAT 使用了名为 GETMONTH.COM 的程序来确定当前月份。在该批处理文件确定了月份之后，就转移至特定于该月的命令，并显示有关约会和会议的一张表：

```
@ECHO OFF
REM Determine the current month using the program
REM GETMONTH.COM. The program returns an exit-status value
REM from 1 through 12 that represents the current month.
REM (January is 1; December is 12.) After the month
REM is determined, branch to the specified month and
REM display important dates for the month.
```

```
GETMONTH
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 GOTO JANUARY
IF ERRORLEVEL 2 IF NOT ERRORLEVEL 3 GOTO FEBRUARY
IF ERRORLEVEL 3 IF NOT ERRORLEVEL 4 GOTO MARCH
IF ERRORLEVEL 4 IF NOT ERRORLEVEL 5 GOTO APRIL
IF ERRORLEVEL 5 IF NOT ERRORLEVEL 6 GOTO MAY
IF ERRORLEVEL 6 IF NOT ERRORLEVEL 7 GOTO JUNE
IF ERRORLEVEL 7 IF NOT ERRORLEVEL 8 GOTO JULY
IF ERRORLEVEL 8 IF NOT ERRORLEVEL 9 GOTO AUGUST
IF ERRORLEVEL 9 IF NOT ERRORLEVEL 10 GOTO SEPTEMBER
IF ERRORLEVEL 10 IF NOT ERRORLEVEL 11 GOTO OCTOBER
IF ERRORLEVEL 11 IF NOT ERRORLEVEL 12 GOTO NOVEMBER
IF ERRORLEVEL 12 GOTO DECEMBER

:JANUARY
ECHO January  1 - New Year's Day
ECHO January 15 - Martin Luther King's Birthday
GOTO DONE

:FEBRUARY
ECHO February 12 - Lincoln's Birthday
ECHO February 14 - Valentine's Day
ECHO February 22 - Washington's Birthday
GOTO DONE

:MARCH
ECHO March 17 - St. Patrick's Day
GOTO DONE

:APRIL
ECHO April 1 - April Fools' Day
GOTO DONE

:MAY
ECHO May 1 - May Day
ECHO Last Monday in May - Memorial Day
GOTO DONE

:JUNE
GOTO DONE

:JULY
ECHO July 4 - Independence Day
GOTO DONE

:AUGUST
GOTO DONE

:SEPTEMBER
ECHO First Monday in September - Labor Day
GOTO DONE

:OCTOBER
ECHO Second Monday in October - Columbus Day
ECHO October 31 - Halloween
GOTO DONE
```

```
:NOVEMBER
ECHO November 11 - Veterans' Day
ECHO Fourth Thursday - Thanksgiving Day
GOTO DONE

:DECEMBER
ECHO Hanukkah begins
ECHO December 25 - Christmas

:DONE
```

在此例中，批处理文件列出了美国几个主要的节日。用 ECHO 命令可以方便地加入生日、纪念日或其他重要日期。

可以用下列 GETMONTH.SCR 命令文件来建立 GETMONTH.COM 程序：

```
N GETMONTH.COM
A 100
MOV AH,2A
INT 21
MOV AL,DH
MOV AH,4C
INT 21

R CX
A
W
Q
```

为建立 GETMONTH.COM 程序，可用下列命令：

```
C:\>DEBUG <GETMONTH.SCR
```

GETMONTH 实用程序可用在任何批处理文件中以根据月份进行决策。由于 GETMONTH 将月份作为 ERRORLEVEL 的值而返回，所以必须用 SCHEDLILE.BAT 中所示的 GOTO 结构来控制处理。

11.3 确定日期

有时可能想在一个月的某一特定日（如 15 号）在批处理文件中运行某些命令。下面的文件 GETDAY.SCR 建立了名为 GETDAY.COM 的程序，它返回的退回状态值说明了是月中的第几天：

```
N GETDAY.COM
A 100
MOV AH,2A
INT 21
MOV AL,DL
```

```
MOV AH,4C
INT 21

R CX
A
W
Q
```

必须用 DEBUG 来从该命令文件中建立实际的程序：

```
C:\>DEBUG < GETDAY.SCR
```

接着就可在批处理文件中用 GETDAY.COM 来测试程序所返回的 ERRORLEVEL 值，并接着作相应处理(我们将在本章末尾给出该方法的例子)。

11.4 确定小时

下面的批处理文件 GETTIME.BAT 用了名为 GETHOUR.COM 的程序来确定当前小时，然后该文件据此而显示一信息：

```
@ECHO OFF
REM Determine the current hour of the day using the
REM GETHOUR.COM program. After the hour is
REM determined, display a message to the user.

GETHOUR

IF ERRORLEVEL  0 IF NOT ERRORLEVEL 6 GOTO TOO_LATE
IF ERRORLEVEL  6 IF NOT ERRORLEVEL 12 GOTO MORNING
IF ERRORLEVEL 12 IF NOT ERRORLEVEL 18 GOTO AFTERNOON
IF ERRORLEVEL 18 GOTO EVENING

:TOO_LATE
ECHO You're up either too late or too early.
GOTO DONE

:MORNING
ECHO Good morning.
GOTO DONE

:AFTERNCON
ECHO Good afternoon.
GOTO DONE

:EVENING
ECHO Good evening. It's getting late.

:DONE
```

GETHOUR.COM 使用下面的命令文件 GETHOUR.SCR：

```
N GETHOUR.COM
```

```
A 100
MOV AH,2C
INT 21
MOV AL,CH
MOV AH,4C
INT 21

R CX
A
W
Q
```

同样，为建立 GETHOUR.COM，必须用 DEBUG：

```
C:\>DEBUG <GETHOUR.SCR
```

所生成的程序可用于任何批处理文件中。在 ERRORLEVEL 中返回的值表示以 24 小时计的当前小时，值 0 表示午夜。

11.5 确定分钟

这组 DEBUG 实用程序如果没有一个返回当前分钟的也就不完整了。下面的命令文件 GETMIN.SCR 建立了一个程序来将分钟作为从 0 至 59 的一个退出状态值而返回：

```
N GETMIN.COM
A 100
MOV AH,2C
INT 21
MOV AL,CL
MOV AH,4C
INT 21

R CX
A
W
Q
```

同前面的例子一样，为建立实际的程序，用下列命令：

```
C:\>DEBUG < GETMIN.SCR
```

现在就有了一组工具可在批处理文件执行时用来确定月份、日期、小时和分钟了。

11.6 使用日期程序

在建立更复杂的批处理文件时，你会发现这些时间和日期实用程序是非常有用的。例

如，可以在 AUTOEXEC.BAT 文件中使用这些工具以在某一给定的日子里执行其他程序。在每日的十五号可能想要自动执行 MSAV 命令以进行病毒检查或者是 DEFRAG 命令以消除硬盘上的碎片。下面的命令说明了可以怎样来完成该处理。

```
GETDAY
IF ERRORLEVEL 15 IF NOT ERRORLEVEL 16 DEFRAG
IF ERRORLEVEL 15 IF NOT ERRORLEVEL 16 MSAV
```

有了这些工具，就能建立一些功能相当强的批处理文件。可对这些程序做些修改以了解可以如何自动完成其他例程任务。

第 12 章

AUTOEXEC.BAT 和 CONFIG.SYS 的预置

12.1 合理使用 AUTOEXEC.BAT

在硬盘上所有的批处理文件中，最重要的是 AUTOEXEC.BAT。与其他批处理文件不同，AUTOEXEC.BAT 在系统每次启动时都要被执行。为保证该文件所执行的命令适于系统的当前配置，应当定期地检查 AUTOEXEC.BAT 文件的内容。本章讨论若干可能想从 AUTOEXEC.BAT 中完成的操作。

12.1.1 修改 CHOICE 命令

在第五章中讨论过，MS-DOS 6 及以后版本中提供了 CHOICE 命令可让批处理文件给出一提示，等待响应，然后用 IF ERRORLEVEL 命令来处理得到的响应。在 AUTOEXEC.BAT 文件中可使用定时 CHOICE 以便在系统启动时可说明是否想要运行 MSAV（MS-DOS 6 及以后版本的病毒检测工具）、DEFRAG（磁盘碎片回收程序），或安装删除跟踪软件 UNDELETE。下面的命令说明了如何在 AUTOEXEC.BAT 中于 15 秒的时间限制内使用 CHOICE：

```
@ECHO OFF
CHOICE Do you want to run MSAV /T:N,15
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 MSAV
CHOICE Do you want to run DEFRAG /T:N,15
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 DEFRAG
CHOICE Do you want to install UNDELETE's file-tracking /T:Y,15
IF ERRORLEVEL 1 IF NOT ERRORLEVEL 2 UNDELETE /LOAD
```

如果在打开 PC 后走开了，系统就会以缺省选择作为响应，即不运行 MSAV 和 DEFRAG 程序，但安装删除跟踪软件。如果没有缺省值，就必须由用户来对每条 CHOICE 提示符作出响应。

12.1.2 使用日期程序

在第 11 章中，我们建立了 GETMONTH 和 GETDAY 程序。有时可能想在特定日期里，从 AUTOEXEC.BAT 中执行特定的命令。例如，在每个月的十五号，可能想要自动执行 MSAV 命令以完成病毒检查，以及自动执行 DEFRAG 命令来消除硬盘上的碎片。下面的命令就说明了可以怎样完成此处理：

```
GETDAY
IF ERROELEVEL 15 IF NOT ERROELEVEL 16 GOTO 15th
:15th
DEFRAG
MSAV
```

12.1.3 预置命令提示符

许多用户都通过修改命令提示符来使他们使用的系统很有个人特点。缺省时(没有用PROMPT 命令),MS-DOS 显示的是当前盘驱动器字母,后跟大于号(>);例如:C>。但是,MS-DOS 5.0 及以后版本的安装程序将 PROMPT 命令放进了 AUTOEXEC.BAT 文件以让 MS-DOS 将当前目录显示作提示符;例如:C:\>。使用 PROMPT 命令和元字符(由美元号 $ 后跟另一字符组成)的组合就可以建立自己的命令提示符。PROMPT 支持下列元字符:

元字符	对应的正文
$$	字符$
$b	字符|
$d	当前系统日期
$e	ASCII 转义字符
$g	字符>
$h	ASCII 退格字符
$l	字符<
$n	当前磁盘驱动器
$p	当前磁盘驱动器和目录
$q	字符=
$t	当前系统时间
$v	当前 MS-DOS 版本
$_	回车/换行

注意:在第八章中,我们讨论了如何用 $e 元字符来生成转义序列以让 ANSI.SYS 设备驱动程序设置屏幕颜色、将命令赋给功能键或定位光标以使批处理文件可在特定位置处放置菜单或其他输出。

在确定了提示符的形状后,可以考虑将相应的 PROMPT 命令放至 AUTOEXEC.BAT 文件中。例如,下列命令在一行上显示当前系统日期,然后在隔两行下面显示用方括号括起的当前盘和目录:

```
PROMPT $d$_$_[$p]
```

假定在 AUTOEXEC.BAT 中有此命令,则提示行会类似于:

```
Thu 04-07-1994
```

[C:\DOS]

可以看出,MS-DOS 显示了当前日期,后跟用方括号括起的当前目录(注意方括号以及所有前面没加 $ 字符的其他字符显示的与键入的完全一样)。通过在 AUTOEXEC.BAT 中设置所希望的提示符,就不必记住 PROMPT 命令的内容,并在每次启动系统后从头输入了。

12.1.4 预置 PRINT 命令

当使用 MS-DOS 的批处理文件(或其他 ASCII 文件)时,MS-DOS 的 PRINT 命令提供了一种方便的方法来打印一文件的内容。第一次执行 PRINT 命令时,MS-DOS 将此软件安装至内存以管理打印机和要打印的文件。为控制此安装,可以在命令行上加入几条开关。这些开关说明了各种选择,例如打印机选择(PRN 一般表示并行打印机,COM1 表示串行打印机);在打印队列中文件数目的选择(从 4 至 32,缺省为 10);计算机每次花多少时间来打印文件。下表简单地描述了 PRINT 各项开关的功能:

开关	功　能
/B	说明打印缓冲区的大小
/C	删除打印队列中的一个或多个文件
/D	说明目标输出设备
/M	说明时间滴答的次数,在这段时间内,PRINT 可以得到对打印机的控制以开始打印
/Q	说明打印队列能存放的文件数
/S	说明打印时间片,这可用于控制 PRINT 每次得到对打印机控制的时间长短
/T	删除打印队列中的所有文件
/U	说明 PRINT 能够等待打印机就绪的时间滴答次数
/P	打印命令行上命名的文件

注意:能带参数的开关只能在第一次执行 PRINT 时被说明。

大多数人都使用 PRINT 的缺省设置,只是因为他们记不住可用的开关。但是,在许多情况下,在 PRINT 命令中加上几个简单的开关可以让打印机和计算机的性能都得以提高。AUTOEXEC.BAT 提供了一种简单的方法来以最优设置运行 PRINT。对大多数用途来说,下面的命令行可以得到很好的性能:

```
PRINT /D:PRN /Q:32 /M:64 /U:16 /B:4096
```

如果用的是与第一个串行口相联的串行打印机,则可将/D 开关改成指向 COM1:

```
PRINT /D:COM1 /Q:32 /M:64 /U:16 /B:4096
```

在此例中,/D 开关告诉 PRINT 打印输出至哪一设备。使用该开关就不再会出现 PRINT 通常在第一次被执行时会出现的提示(Name of list device [PRN:])。/Q 开关

安装一个大的足以存放 32 个文件的打印队列。/M 开关告诉 PRINT 在每次开始打印后有 64 个时钟滴答可拥有对计算机的控制。/U 开关告诉 PRINT 在等待打印机就绪时(如果当时打印机正忙着)保持 16 个时钟滴答的控制。最后,/B 开关设置了一个有 4096 字节的打印缓冲区。PRINT 每次从磁盘上读文件时就用该文件中的字符填入此缓冲区中。由于该缓冲区相当大(缺省是 512 个字节),所以 PRINT 为打印一文件而必须完成的较慢的磁盘输入操作的次数就减少了。降低磁盘输入操作的次数就改进了整个系统的性能。

12.1.5 用 SMARTDRV.EXE 进行磁盘高速缓存

比起计算机上电子元件的速度,磁盘是很慢的。改进性能的一种方法是减少 MS-DOS 必须完成的磁盘读写操作的次数。如果计算机有扩展内存,则可以通过安装 SMARTDRV 的磁盘高速缓存来改进系统性能。一般来说,SMARTDRV 使用计算机扩展内存的一部分来作为所谓高速缓存的缓冲区,这样,就减少了计算机必须完成的低速磁盘读写操作。根据计算机具体有多少扩展内存,AUTOEXEC.BAT 中的 SMARTDRV 命令可能会有所不同。但是,下面的命令应该能提高一个具有 2 至 4MB 扩展内存的系统的性能。

```
SMARTDRV C+ 1024
```

这里,SMARTDRV 使用了 1MB 的磁盘高速缓存,并对 C 盘完成读写的高速缓存。

一定要保证 CONFIG.SYS 文件安装了一设备驱动程序以管理扩展内存使之能为 SMARTDRV 所用。如果用的 MS-DOS 5.0 或更新的版本,则可用下列的 CONFIG.SYS 命令安装 HIMEM.SYS 驱动程序:

```
DEVICE=C:\DOS\HIMEM.SYS
```

记住:DEVICE=是 CONFIG.SYS 中的命令,而不是 AUTOEXEC.BAT 中的命令。

12.1.6 用 FASTOPEN 来改进目录搜索操作

如果你的系统没有扩展内存,那么可以使用 FASTOPEN 命令而不是 SMARTDRV。FASTOPEN 命令减少了 MS-DOS 花在搜索经常使用的磁盘上文件的时间。每次打开、打印或拷贝文件时,MS-DOS 一般都要读盘以寻找该文件的目录项。当 MS-DOS 找到该目录项时,就确定文件的大小以及该文件在盘上存储的位置。FASTOPEN 命令安装了能记录所用的文件的内存驻留软件,并将该文件目录项的备份放在内存的一缓冲区中。当打开一文件时,MS-DOS 首先检查 FASTOPEN 的存放文件目录项的缓冲区。如果该缓冲区含该文件的目录项,就不再在低速磁盘上搜索此目录项。如果缓冲区中没有目录项,MS-DOS 就从磁盘上读取该项,并将其备份放在 FASTOPEN 的缓冲区中,这样下次使用该文件时,MS-DOS 就能在缓冲区中找到该项。如果该缓冲区满了,则最先的项就被替换掉。

FASTOPEN 命令并不是对所用的每个文件都改进系统性能,而只是对经常重复使用的文件。下面的 FASTOPEN 命令让 MS-DOS 记录 C 盘上的多至 75 个文件:

```
FASTOPEN C:=75
```

12.1.7 使用 LOADHIGH 命令

如果使用的是 386 或更高级的微处理器上的 MS-DOS 5.0 或更新的版本，则可以利用在 640KB 至 1MB 之间的 384KB 的内存(被称为保留或高内存)。为使用高内存，需要在 CONFIG.SYS 文件中使用下列命令：

```
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE NOEMS
DOS=UMB
```

这些语句使 MS-DOS 能访问并管理高内存。具体的细节也许会有些不同，如 NOEMS 参数可能没有。在安装了支持高内存的程序后，就可以将设备驱动程序和内存驻留程序装入高内存，从而释放了系统的常规内存以便为 MS-DOS 及用户自己的 MS-DOS 程序所用。为将设备驱动程序装入高内存，可在 CONFIG.SYS 文件中使用 DEVICEHIGH=命令。例如，下面的命令将 ANSI.SYS 设备驱动程序装入高内存：

```
DEVICEHIGH=C:\DOS\ANSI.SYS
```

为将内存驻留程序(以及内存驻留 DOS 命令)装入高内存，可以使用 LOADHIGH 命令。例如，下面的 LOADHIGH 命令将 PRINT 和 SMARTDRV 程序放入高内存：

```
LOADHIGH C:\DOS\PRINT /D:PRN /Q:32
LOADHIGH C:\DOS\SMARTDRV C+ 1024
```

12.1.8 在磁盘上查找计算机病毒

MS-DOS 6 及以后版本提供了 MSAV 命令来在磁盘上查找并删除计算机病毒。使用 MSAV 的方式有两种：第一，在 MSAV 的图形用户界面中选择命令和目标驱动器及操作；第二，使用命令行开关，这与 MS-DOS 命令的使用很相似。如果你经常与其他用户交换软盘，就应该常常检查盘上是否被病毒感染了。下面的命令用第二种方法运行 MSAV 来扫描并删除 C 盘上的病毒：

```
MSAV C: /C
```

可以将此命令行放在 AUTOEXEC.BAT 文件中。但是，根据盘上文件数目的多少，MSAV 所需的处理时间各不相同，所以不一定会在每次系统启动时都想要运行 MSAV。通过使用 CHOICE 命令，可以让 AUTOEXEC.BAT 询问用户是否想要运行 MSAV。如果回答 Y，则 AUTOEXEC.BAT 就调用 MSAV；否则，就不调用。

12.1.9 允许删除跟踪

MS-DOS 从 5.0 版开始，提供了 UNDELETE 命令来恢复被意外删除的文件。为提高 UNDELETE 恢复文件的可能性，大多数 MS-DOS 5.0 版的用户都通过放置与下面类

似的一条 MIRROR 命令至 AUTOEXEC.BAT 文件中来安装删除跟踪支持程序：

```
MIRROR C: /TC
```

MS-DOS 6 及以后版本不支持 MIRROR 命令。为在这些版本中使用删除跟踪，必须装入 UNDELETE 文件跟踪软件：

```
UNDELETE /LOAD
```

文件跟踪的类型是由 UNDELETE.INI 文件中的设置决定的。可以使用命令行开关（或 UNDELETE 实用工具）来配置保护级别。如果有大量的自由磁盘空间，那么可以用最大安全性的 Delete Sentry 保护来执行 UNDELDTE，这在 MS-DOS 6 及以后版本中也提供了。

12.1.10 提高键盘的反应速度

MODE 命令可用来控制键盘的反应速度。MODE 可以用两种不同的键盘设置来执行：击键速度和击键延迟。击键速度说明每秒钟键重复的次数。击键延迟说明在重复击键前必须按住一个键的时间长短。

PC 支持下面 32 种击键速度：

击键速度	每秒钟的字符数	击键速度	每秒钟的字符数
1	2.0	17	8.4
2	2.1	18	8.6
3	2.3	19	9.2
4	2.5	20	10.0
5	2.7	21	10.9
6	3.0	22	12.0
7	3.3	23	13.3
8	3.7	24	15.5
9	4.0	25	16.0
10	4.3	26	17.1
11	4.6	27	18.5
12	5.0	28	20.0
13	5.5	29	21.8
14	6.0	30	24.0
15	6.7	31	26.7
16	7.5	32	30.0

下表列出了 PC 支持的 4 种击键延迟：

击键延迟	以秒计的延迟	击键延迟	以秒计的延迟
1	1/4	3	3/4
2	1/2	4	1

Enter a choice: 1

但是,同时也能使用 F5 和 F8 键来控制 MS-DOS 如何处理 CONFIG.SYS。

每次系统启动时,MS-DOS 都首先处理 CONFIG.SYS 命令以在内存中配置自己。接着,MS-DOS 执行 AUTOEXEC.BAT 文件中的命令。但是,有时可能并不想让 MS-DOS 处理 CONFIG.SYS 命令或执行AUTOEXEC.BAT中的命令。在 MS-DOS 6.0 及以后版本中,每次系统启动时都会看见此信息:

Starting MS-DOS ...

如果在此信息出现后立即按 F5 键,MS-DOS 就不处理 CONFIG.SYS 或执行 AUTOEXEC.BAT 中的命令。

注意:如果按 F5 键,MS-DOS 就绕过 CONFIG.SYS 和 AUTOEXEC.BAT,系统将以最小配置来启动。例如,搜索路径将只含 C:\DOS,而不是 AUTOEXEC.BAT 中定义的路径。

如果想让 MS-DOS 跳过其他命令而处理一些 CONFIG.SYS 和 AUTOEXEC.BAT 命令,则可在 MS-DOS 显示其启动信息后按 F8 键。MS-DOS 接着就提示按 Y 或 N 以说明是否想处理或跳过各条命令。

例如,假定 CONFIG.SYS 文件的头两行有下列内容:

```
DEVICE=C:\DOS\SETVER.EXE
DEVICE=C:\DOS\HIMEM.SYS
```

如果在 MS-DOS 启动后立即按 F8 键,就可以通过对[Y,N]提示符回答 Y 或 N 而有选择性地让 MS-DOS 处理命令:

```
DEVICE=C:\DOS\SETVER.EXE [Y,N]? N
DEVICE=C:\DOS\HIMEM.SYS [Y.N]? Y
```

在本例中,MS-DOS 将不处理 SETVER.EXE 命令项而处理 HIMEM.SYS 命令项。

在对 CONFIG.SYS 中的每条命令提示作完回答后,MOS-DOS 询问是否想要执行 AUTOEXEC.BAT 文件中的命令:

Process AUTOEXEC.BAT [Y,N]?

如果按了 N,MS-DOS 就不处理任何 AUTOEXEC.BAT 命令,而是立即显示命令提示符。如果按 Y,MS-DOS 就执行 AUTOEXEC.BAT 文件中的命令。MS-DOS 6.2 还允许有选择地处理或不处理每条命令,如果让 MS-DOS 执行命令就按 Y,按 N 则跳过命令。

注意:F8 键主要是在 CONFIG.SYS 或 AUTOEXEC.BAT 中的命令使启动不能成功进行时用于使系统启动。

绕过 CONFIG.SYS 和 AUTOEXEC.BAT

有时可能想让 MS-DOS 绕过对 CONFIG.SYS 和 AUTOEXEC.BAT 的处理。例如，假定其中的一个文件被修改后使系统无法启动。通过重新启动系统并按 F5 键，就可以让系统得以启动，然后就可以编辑该文件以作必要的修改。

逐条处理 CONFIG.SYS 和 AUTOEXEC.BAT 命令

有时可能想让 MS-DOS 有选择性地处理 CONFIG.SYS 和 AUTOEXEC.BAT 命令。当 MS-DOS 显示其启动信息时，可以按 F8 键来让 MS-DOS 提示你按 Y 以处理各条命令，或按 N 以跳过命令。在为各条 CONFIG.SYS 命令说明了是否希望执行后，MS-DOS 就提示你确定是否想执行 AUTOEXEC.BAT 文件中的命令。

12.2.2 创建系统配置菜单

在本章前面，我们讨论了如何用 CHOICE 命令来控制 AUTOEXEC.BAT 文件中命令的执行。此外，我们还知道，在系统启动时按 F8 键就可以为 CONFIG.SYS 中的命令生成 Y/N 提示。虽然这种逐条提示可对让 MS-DOS 处理哪些命令加以控制，但是回答这些提示在 CONFIG.SYS 文件含许多命令时会很费时的。

取决于具体所用的应用程序，可能会希望经常使用不止一个 CONFIG.SYS 文件。例如，如果在使用网络，那么就可能会用 CONFIG.SYS 文件来安装各种网络设备驱动程序。但是，如果没有使用网络，那么就可能想用不同的 CONFIG.SYS 文件来释放常规内存以便能运行诸如数据库这样的大型程序。还可能对第三种配置会有第三个 CONFIG.SYS 文件。过去，改变配置的最简单的方法是建立名字类似于 CONFIG.NOR(用于能访问最大数量内存的普通操作)、CONFIG.NET(用于网络)、CONFIG.DBS(用于数据库)的文件。接着，当需要一特定的设置时，就可以将特定的文件内容拷贝至 CONFIG.SYS，然后重新启动计算机。例如，下面的 COPY 命令可将用于网络配置的命令拷贝至 CONFIG.SYS：

```
C:\>COPY CONFIG.NET CONFIG.SYS
```

利用 MS-DOS 6 及以后版本，就可不必管理多个 CONFIG.SYS 文件。MS-DOS 提供了允许建立一选项菜单以供选择所期望的配置的 CONFIG.SYS 命令。

下面是 MS-DOS 6 及以后版本 CONFIG.SYS 中的预置命令：

命　令	目　的
MENUCOLOR	说明 CONFIG.SYS 启动菜单的前景和背景色。
MENUITEM	说明出现在 CONFIG.SYS 启动菜单中的一选项。
MENUDEFAULT	说明 CONFIG.SYS 启动菜单的缺省选项。
INLUCDE	将在一个配置块中说明的启动命令放入另一块中。
SUBMENU	说明启动选项的一子菜单。

使用这些命令来预置系统配置时，可将 CONFIG.SYS 文件分成五个命令块。例如，可将为网络配置所需要的命令放在一块中，而将为数据库应用所需要的命令放在另一块中。

在 CONFIG.SYS 中所建立的每一块都有一个用方括号括起的唯一名字，如[High Memory]，[Network]和[Database]。块名可以长达 70 个字符。

三个配置(HighMemory，即通常配置；Network；以及 Database)中的每一个都需有自己的一组命令。如果它们被组织成各个块，则 CONFIG.SYS 文件可含下列内容：

```
[HighMemory]
FILES=50
BUFFERS=30
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE NOEMS
DOS=HIGH,UMB

[Network]
FILES=50
BUFFERS=30
DEVICE=C:\NETDRIVE.EXE
DEVICE=C:\NETPRINT.EXE

[Database]
FILES=50
BUFFERS=30
DEVICE=RAMDRIVE.SYS 512
DEVICE=DBCACHE.SYS
```

通常会有几条命令为几个块所共有，如 BUFFERS=、FILES=以及经常使用的 DEVICE=命令。

为避免这些命令在各块中的重复，可以定义一个名为[COMMON]的特殊块以存放这样的公共命令。[COMMON]块可以有不只一个，而且 CONFIG.SYS 文件常常头尾都是该块。安装程序将会把新的命令加入到最终的[COMMON]块中。MS-DOS 不管选择的配置选项是什么都会执行在[COMMON]块中的命令。例如，下面的 CONFIG.SYS 文件有四个块，一个含其他三个块共有的命令，一个是通常的高内存配置，一个是网络配置，还有一个是数据库操作：

```
[COMMON]
FILES=50
```

```
BUFFERS=30

[HighMemory]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE NOEMS
DOS=HIGH,UMB

[Network]
DEVICE=C:\NETDRIVE.EXE
DEVICE=C:\NETPRINT.EXE

[Database]
DEVICE=RAMDRIVE.SYS 512
DEVICE=DBCACHE.SYS
```

在定义了各种配置后，就需要一种方法能在系统启动时选择其中的一种配置。为此，可以用 CONFIG.SYS 文件中的 MENUITEM 命令来建立一启动菜单。每条 MENUITEM 命令都说明了在启动菜单中出现的一个选项，并标识了在选择了该选项时 MS-DOS 要处理的命令块。在每条 MENUITEM 命令中，都要说明块名以及将在启动菜单中出现的正文，如下所示：

MENUITEM=块名[,菜单正文]

例如，下面的命令建立了用于高内存、网络和数据库配置的选项：

```
[MENU]
MENUITEM=HighMemory,Configure for high-memory use
MENUITEM=Network,Configure for network
MENUITEM=Database,Configure for database
```

可以看出，MENUITEM 命令被放在一名为[MENU]的特殊块中。将这些命令与前面的配置块组合在一起，CONFIG.SYS 就成为：

```
[MENU]
MENUITEM=HighMemory,Configure for high-memory use
MENUITEM=Network,Configure for network
MENUITEM=Database,Configure for database

[COMMON]
FILES=50
BUFFERS=30

[HighMemory]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE NOEMS
DOS=HIGH,UMB

[Network]
DEVICE=C:\NETDRIVE.EXE
DEVICE=C:\NETPRINT.EXE

[Database]
DEVICE=RAMDRIVE.SYS 512
DEVICE=DBCACHE.SYS
```

当启动一个含此 CONFIG.SYS 文件的系统时，MS-DOS 就会在屏幕上显示下列菜单：

```
MS-DOS 6.2 Startup Menu
=======================

    1. Configure for high-memory use
    2. Configure for network
    3. Configure for database

Enter a choice: 1
```

要选择一菜单选项，只需敲入该选项对应的数字或用箭头键加亮显示该选项，然后按 Enter。

在 MENUITEM 命令中使用描述性的菜单正文是可选的。如果没有说明任何正文，MS-DOS 就在启动菜单中显示块名。例如，下面的菜单命令中就没有任何描述性的正文：

```
[MENU]
MENUITEM=HighMemory
MENUITEM=Network
MENUITEM=Database
```

当系统启动时，MS-DOS 显示下列信息：

```
MS-DOS 6.2 Startup Menu
=======================

    1. HighMemory
    2. Network
    3. Database

Enter a choice: 1
```

控制启动菜单的颜色

缺省时，MS-DOS 以黑白色显示启动菜单，为帮助进一步预置系统启动，CONFIG.SYS 的 MENUCOLOR 命令可用来说明启动菜单的颜色。该命令的格式如下：

```
MENUCOLOR=前景色[,背景色]
```

前景色和背景色是用从 0 至 15 的颜色值加以说明的，如下表所示：

颜色值	颜色	颜色值	颜色
0	黑	8	灰
1	蓝	9	深蓝
2	绿	10	深绿
3	青	11	深青
4	红	12	深红
5	紫	13	深紫
6	棕	14	黄
7	白	15	亮白

例如，下面的命令让 MS-DOS 在蓝色背景上显示深红色菜单：

```
[MENU]
MENUCOLOR=12,1
MENUITEM=HighMemory,configure for high memory use
MENUITEM=Network,Configure for network
MENUITEM=Database,Configure for database
```

选择一缺省菜单选项

我们知道，在 AUTOEXEC.BAT 文件中加入 CHOICE 命令时，计算机就等待键盘响应，而不是立即完成其启动。类似地，当在 CONFIG.SYS 文件中加入 MENUITEM 命令时，系统就不会启动，除非在键盘上选择了所期望的选项。使用 CHOICE 命令，可以定义缺省选项以及响应时间限制；MENUITEM 命令也一样。

你可能已经注意到了前面显示的启动菜单将第一个菜单选项显示为缺省选择。根据具体的配置设置，可以选择不同的选项来作为缺省选项。此外，还可以说明从 0 至 90 秒的时间限制，在此之后 MS-DOS 将自动地选择缺省选项。通过使用时间限制，就可以保证系统总能启动，即使你不在机器边。

CONFIG.SYS 的 MENUDEFAULT 命令可用来说明缺省选项以及时间限制。MENUDEFAULT 命令的格式如下：

```
MENUDEFAULT=块名[,时间限制]
```

例如，下列的 MENUDEFAULT 命令让 MS-DOS 在 45 秒内仍无选项被选择时就选择 Network 配置：

```
[MENU]
MENUDEFAULT=Network,45
MENUITEM=HighMemory,Configure for high-memory use
MENUITEM=Network,Configure for network
MENUITEM=Database,Configure for database
```

当说明了缺省选项及时间限制时，MS-DOS 显示剩余秒数的倒计时，直到有一选项被选中：

```
MS-DOS 6.2 Startup Menu
=======================
    1. Configure for high-memory use
    2. Select network configuration
    3. Select database configuration
Enter a choice: 2          Time remaining: 25
```

时间限制为 0 时该菜单就会被跳过。

在一个块中包含另一块的命令

为用启动菜单命令配置 CONFIG.SYS 文件，可以将该文件分成几个块。根据具体对命令的组织，一个块可能需要含有同时为另一块所含的命令。不必重复这些命令，而是可以用 INCLUDE 命令来包含第二个块的命令。INCLUDE 的格式如下：

```
INCLUDE=块名
```

例如，假定想要修改[Network]块以利用高内存，则可以加入如下所示的命令：

```
[Network]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE NOEMS
DOS=HIGH,UMB
DEVICE=C:\NETDRIVE.EXE
DEVICE=C:\NETPRINT.EXE
```

可以看出，[Network]块中的前三条命令与[HighMemory]块中定义的相同。不用重复这些命令，可以在[Network]中包含[HighMemory]命令：

```
[MENU]
MENUITEM=HighMemory,Configure for high-memory use
MENUITEM=Network,Configure for network
MENUITEM=Database,Configure for database

[COMMON]
FILES=50
BUFFERS=30

[HighMemory]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE NOEMS
DOS=HIGH,UMB

[Network]
INCLUDE=HighMemory
DEVICEHIGH=C:\NETDRIVE.EXE
DEVICEHIGH=C:\NETPRINT.EXE

[Database]
DEVICE=RAMDRIVE.SYS 512
DEVICE=DBCACHE.SYS
```

建立若干菜单层次

随着系统配置变得较为复杂，有时只用一个菜单来体现不同的配置选项可能会有困难。所以，可以使用SUBMENU命令来说明另一层次的菜单。SUBMENU命令的格式是：

SUBMENU=菜单块[,菜单正文]

菜单块说明了含另一层MENUITEM命令的一个CONFIG.SYS块的名字。当系统启动时，MS-DOS首先找到[MENU]块，并显示它所含的命令。如果选择了一个对应某一子菜单的选项，MS-DOS就显示第二个菜单，后面还可以跟着第三或第四个菜单。

例如，假定如果选择了数据库选项，并且还想让MS-DOS询问是否也想装入网络软件，则可使用下面CONFIG.SYS文件中的SUBMENU命令：

```
[MENU]
MENUITEM=HighMemory,Configure for high-memory use
MENUITEM=Network,Configure for network
SUBMENU=AskNetwork,Configure for database

[COMMON]
FILES=50
BUFFERS=30

[HighMemory]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE NOEMS
DOS=HIGH,UMB

[Network]
DEVICE=C:\NETDRIVE.EXE
DEVICE=C:\NETPRINT.EXE

[Database]
DEVICE=RAMDRIVE.SYS 512
DEVICE=DBCACHE.SYS

[AskNetwork]
MENUITEM=DBNoNet,Do not load network
MENUITEM=DBNet,Configure for network

[DBNoNet]
INCLUDE=Database

[DBNet]
INCLUDE=Network
INCLUDE=Database
```

可以看出，[MENU]块含一条可选择第二层菜单选项的SUBMENU命令。当系统启动时，MS-DOS显示下列信息：

```
MS-DOS 6.2 Startup Menu
```

```
1. Configure for high-memory use
2. Configure for network
3. Configure for database

Enter a choice: 1
```

如果选择了选项 3，MS-DOS 就显示第二个选项菜单，如下所示：

```
MS-DOS 6.2 Startup Menu
=======================

1. Do not load network
2. Configure for network

Enter a choice: 1
```

如果选择了选项 1，则 MS-DOS 转移至[DBNoNet]块，该块用现存的[Database]块来配置系统。如果选择选项 2，则 MS-DOS 转移至[DBNet]块并使用[Network]和[Database]块。在此例中，子菜单命令依赖于起始菜单中的已存在的块，但是，并非必须如此。一条 SUBMENU 命令引用了含 MENUITEM 命令(以及可选的 MENUDEFAULT 和 MENUCOLOR 命令)的一个块。这些 MENUITEM 引用块的方式与它们在主菜单中引用块的方式相同。这些块可以含任何正确的 CONFIG.SYS 命令，包括其他的 SUBMENU 和 INCLUDE 命令。

12.3 CONFIG.SYS 和 AUTOEXEC.BAT 之间的交互

有时可能想让 AUTOEXEC.BAT 文件中的命令随系统启动时所选择的配置而改变。缺省时，MS-DOS 将从启动菜单中选择的配置块名字赋给 CONFIG 环境命令。使用前面示例的 CONFIG.SYS 时，CONFIG 环境变量的值可以是 HighMemory、Network、DBNoNet 和 DBNet。在 AUTOEXEC.BAT 文件中，可以使用%CONFIG%有名参数来确定 CONFIG 环境变量的当前设置：

```
IF %CONFIG%==HighMemory GOTO HIGH_MEMORY_COMMANDS
IF %CONFIG%==Database GOTO DATABASE_COMMANDS
IF %CONFIG%==Network GOTO NETWORK_COMMANDS
```

12.3.1 在 CONFIG.SYS 中使用 SET

在第六章中说过，可以用 SET 命令来为 MS-DOS 环境变量赋值。从 MS-DOS 6 及以后版本开始，可以在 CONFIG.SYS 文件中使用 SET。根据在系统启动时所选择的配置，可以为 CONFIG.SYS 中的环境变量赋值，而对这些变量，可以从 AUTOEXEC.BAT 中进行测试。例如，取决于计算机的内存配置，可以建立一个名为 SMART_DRIVE 的含

INSTALLED 或 NOT_INSTALLED 值的环境变量：

```
[HighMemory]
DEVICE=C:\DOS\HIMEM.SYS
DEVICE=C:\DOS\EMM386.EXE NOEMS
DOS=HIGH,UMB
DEVICE=C:\DOS\SMARTDRV.EXE
SET SMART_DRIVE=INSTALLED
```

随后，在 AUTOEXEC.BAT 中，可以测试该变量的值：

```
IF "%SMART.DRIVE"=="INSTALLED" SomeCommand
```

第 13 章

批处理文件和 Microsoft Windows

13.1 AUTOEXEC.BAT 和 Windows

虽然大多数人在运行 Microsoft Windows 时，不会想起 MS-DOS 的批处理文件，但 MS-DOS 确实提供了一些值得考虑的批处理文件的功能。本章就来简单地讨论其中的几个功能。

有关 Windows 需记住的最重要的事情之一，是它在 MS-DOS 之上形成了一个操作环境。这意味着配置 MS-DOS 的方式会影响 Windows 执行的方式，例如，设备驱动程序和其他配置命令(如 SmartDrive)；除此之外，可在 AUTOEXEC.BAT 中设置的两个环境参数对 Windows 来说是特别重要的。

第一个是确定 Windows 用来存放临时文件的驱动器的 TEMP 参数。虽然并不一定要说明一个 TEMP 驱动器，但一般来说应该加以说明。下面的命令让 Windows 将临时文件保存在 D 驱动器上：

```
SET TEMP=D:
```

将一虚拟盘赋给 TEMP 一般总能改善系统的性能。

Windows 3.1 版的用户可利用另一个重要的环境参数 WINPMT，该参数为在 Windows 中运行的 MS-DOS 设置命令提示符。下面的命令将 WINPMT 环境参数置成信息 Temporary MS-DOS，后跟方括号中的当前目录名：

```
SET WINPMT=Temporary MS-DOS [$P]
```

当从 Windows 中启动一次 MS-DOS 会话时，显示的提示符就包括该信息以及路径信息。信息 Temporary MS-DOS 提示你正在从 Windows 中运行一次 MS-DOS 会话，必须使用 EXIT 命令来返回至 Windows。使用该方法可为单独的 MS-DOS 用 PROMPT 命令说明一提示符，而为 Windows 中的 MS-DOS 会话用 WINPMT 环境参数说明另一提示符。

13.2 使用 WINSTART.BAT

每次 MS-DOS 启动时，都执行 AUTOEXEC.BAT。类似地，当以 386 增强模式开始启动 Windows 时，Windows 都会执行一个名为 WINSTART.BAT 的特殊批处理文件。虽然使用 WINSTART.BAT 的方式与使用任何 MS-DOS 批处理文件的方式一样，但是，该批处理文件的主要目的是为基于 Windows 的应用程序配置环境。所以，可以有为基于

Windows 的应用程序而用的和为 MS-DOS 会话而用的两个环境。当从 Windows 中启动 MS-DOS 会话时，Windows 在执行 WINSTART.BAT 之前会先复制当前的 MS-DOS 环境。

正如可以修改 AUTOEXEC.BAT 以为 MS-DOS 会话配置环境一样，也可以修改 WINSTART.BAT 以配置 Windows 环境。例如，取决于所用的基于 Windows 的应用程序，可能会想要修改搜索路径以删除路径中的某些目录（与 MS-DOS 特定相关的那些程序）或加入某些目录。类似地，在执行 Windows 之前，可能想要释放掉基于 MS-DOS 的内存驻留程序（TSR），并加入基于 Windows 的 TSR。不论需要完成的操作是什么，都可以将必要的命令放在 WINSTART.BAT 中。

13.2.1 为 Windows 修改搜索路径

我们在第二章中简单讨论过，当将目录加至搜索路径时，应该保证这些目录中有常常要使用的命令。搜索路径中的目录越多，MS-DOS 就会花越多的时间来搜索目录内容。我们知道，WINDOWS 目录和存放基于 Windows 的应用程序的目录含有大量的文件。所以，只有在准备使用 Windows 时，才应将这些目录加至搜索路径中。

将这些目录加至搜索路径的方法之一是在 WINSTART.BAT 文件中重定义路径。下面的命令提供了为 Windows 所使用的一种典型路径：

```
PATH C:\WINDOWS;C:\DOS;C:\EXCEL;C:\WINWORD
```

该命令假定文件是在 C 盘上，并被安装至缺省目录中。记住路径只为 Windows 环境而改变。对 MS-DOS 会话来说，在输入启动 Windows 的命令之前定义的路径仍然有效。

另一种方法是建立一个启动 Windows 的批处理文件。这样，在启动 Windows 时就能控制环境，而在推出 Windows 时重置环境。当用自己的批处理文件启动 Windows 时，生成的环境是全局的——它既影响 Windows 环境，又影响 MS-DOS 环境；而用 WINSTART.BAT 所作的修改却不影响 MS-DOS 会话。

下面的批处理文件 WINDOWS.BAT 将当前搜索路径保存在 OLD_PATH 环境参数中。接着，该批处理文件将三个目录加至该路径中并激活 Windows。在退出 Windows 后，该批处理文件又恢复原先的搜索路径。

```
@ECHO OFF
SET OLD_PATH=%PATH%
PATH C:\WINDOWS;C:\EXCEL;C:\WINWORD;%PATH%
WIN
PAHT %OLD_PATH%
SET OLD_PATH=
```

将此批处理文件放在搜索路径上的目录中。当想要使用 Windows 时，可如下这样激活 WINDOWS.BAT：

```
C:\>WINDOWS
```

13.3 在 Windows 中使用批处理文件

在 Windows 中，一个程序项就是一个程序管理器(Program Manager)中对应特定程序的图符。使用程序管理器的 File 菜单，便可以为自己的批处理文件创建图符。为此，可遵循下列步骤：

1. 保证想要将新图符加至进去的组(group)是活动的。

2. 从 Program Manager 的 File 菜单中选择 New。

3. 当 Windows 显示 New Program Object 对话框时，选择 Program Item 并按 Enter 键。

4. 当 Windows 显示 Program Item Properties 对话框(如图 13-1 所示)时，敲入想要出现在该图符下的名字，然后按一次 Tab 键。在 Command Line 正文行中，敲入该批处理文件的完整路径以及要传给该文件的任何参数。

5. 选择 OK。新建立的程序项的图符应出现在当前的窗口组中。如使用鼠标，这时就可以对该图符按两次开关而激活此批处理文件了。

图 13-1 Program Item Properties 对话框

从 Windows 中使用批处理文件的优点起初可能不太明显，但它们确可用于各种有用的目的。最常用的两个目的是为一特定程序而控制环境，以及为一次 MS-DOS 会话而预置环境。当然，用图符来访问批处理文件只是重要的第一步。

13.3.1 使用批处理文件来启动程序

关于在 Windows 中使用批处理文件要记住的最重要的一点是该批处理文件对环境所进行的任何修改都只是在特定的 MS-DOS 会话期间才起作用。如果有多个 MS-DOS 会话在运行，则每个会话都有其自己的环境。当一个 MS-DOS 会话结束时，相应的改变也就消失了。当然，在此会话期间所完成的操作的结果(如删除一文件)是永久的。

通过建立批处理文件来从 Windows 中启动 MS-DOS 应用程序，就可以利用这种环境局限性的优点。对环境所作的一些修改包括修改路径、进入一启动目录或装入一 TSR 以和某特定程序一起使用(记住，该 TSR 只有在 MS-DOS 会话期间才能使用。如果想要在所有 MS-DOS 会话期间访问一 TSR，就必须在启动 Windows 之前装入它)。下面的批处理文件说明了如何用此方法来启动 Microsoft Word：

```
@ECHO OFF
```

```
PATH=C:\\WORD;%PATH%
CD \LETTERS
CALCIT
WORD
```

如果只从 Windows 中调用 Microsoft Word，则就不必将其目录加至搜索路径中。在此例中，CALCIT 是一假想的 TSR，它是在启动 Word 程序之前被装入的。由于这里的修改对一次 Word 会话是唯一的，所以不必担心在退出 Word 时要重新设置它们。退出 Word 时，也就退出了 MS-DOS 会话，所以此环境也就消失了。

注意：一般来说，当退出 MS-DOS 应用程序时，该 MS-DOS 会话就结束了，并返回至 Windows。但是，有些 TSR 使 Windows 不能判别 MS-DOS 会话已经结束，这种情况就必须使用 EXIT 命令来返回至 Windows。

13.3.2 建立定制的 MS-DOS 会话

在 MS-DOS 6 及以后版本中，COMMAND 命令的/K 开关可允许在每次装入一新的 COMMAND.COM 命令处理器时说明要被执行的一个批处理文件或程序。该性能最为有用的一个用途是在 Windows 中预置 MS-DOS 会话。

第一步是建立含环境配置命令的一个批处理文件。下面的批处理文件 WINEXEC.BAT 含一些典型的命令：

```
PROMPT CUSTOM MS-DOS ($P)
PATH C:\DOS;C:\BATCH;C:\PROGRAM
SET TEMP=D:
```

该批处理文件用 PROMPT 和 PATH 命令来配置基本环境，然后将 D 驱动器赋给 TEMP 目录。

使用/K 开关就能在 Program Manager 中建立一图符，在此图符上按两次鼠标开关就可以启动一个带预置环境的 MS-DOS 会话。假定你的批处理文件在 C 盘 BATCH 目录中，Program Item Properties 对话框中的 Command Line 正文框的内容如下：

```
C:\DOS\COMMAND.COM /K C:\BATCH\WINEXEC.BAT
```

当按两次该新图符时，WINEXEC.BAT 就配置为新 COMMAND.COM 所用的环境，这类似于用 AUTOEXEC.BAT 配置最初的 MS-DOS 环境。

虽然还可用其他方法，但这是预置一个 MS-DOS 会话的最有效的方法。在此批处理文件中可以有任何类型的命令，但最常见的是配置命令和 TSR。

在 MS-DOS 6 以前的版本中，最好的方法是启动一普通的 MS-DOS 会话，然后再运行一个对环境作出所有必要修改的批处理文件。这样得到的环境就与在 MS-DOS 6 及以后版本中得到的环境完全一样，唯一的差别在于不能自动运行此批处理文件。

13.4 高级 Windows 批处理文件管理

建立一个对应一批处理文件的程序项是将一图符赋给一批处理文件的最简单的方法，但这么做却没有提供对批处理文件的高度控制。为了更好地控制批处理文件，需要从 Windows 中开发程序信息文件(PIF)。PIF 是定义 Windows 和 MS-DOS 会话(以及运行在此会话中的应用程序)之间关系的特殊文件。Windows 中的标准 MS-DOS 会话是由一个名为 DOSPRMPT.PIF 的 PIF 控制的。修改此 PIF 就为通过按两次 MS-DOS Prompt 图符而建立的 MS-DOS 会话改变了与 Windows 的关系。

为建立 PIF，可按两次 Windows 的 PIF Editor 程序图符(缺省时，该图符在 Main group 中)。如果是以增强方式运行 Windows，就会看见图 13-2 所示的对话框。如果是在标准方式下，则对话框会有些不同，但这里讨论的设置是相同的。

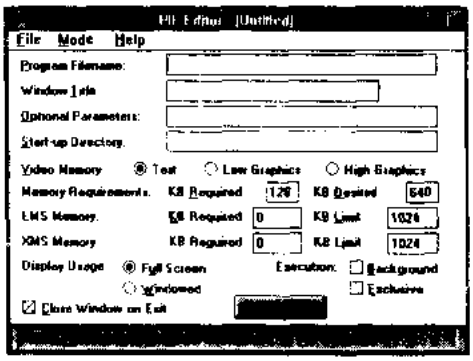


图 13-2 PIF Editor 对话框

必须说明的唯一内容是 Program Filename 正文框中的批处理文件名。对所有其他的设置都可以使用缺省设置，但建立 PIF 的优点是通过预置这些设置而达到更大的控制(后面我们将考查这两种选项：传递可选参数以及控制输出)。

在建立了 PIF 之后，必须保存它并接着为它建立一个程序项，这样它就被表示成 Program Manager 中的一图符。遵循前面在为一批处理文件创建一程序项时描述的步骤，但在 CommandLine 正文框中应输入 PIF 文件名而不是批处理文件名(缺省时，PIF 被存放在 WINDOWS 目录中)。

13.4.1 传递参数给批处理文件

我们知道，参数的说明使用户可以建立更为灵活的批处理文件。在 Windows 中使用批处理文件时，有两种方法可用来说明参数值。

当为一批处理文件建立 PIF 时，可以在 Optional Parameters 正文框中输入各参数所用的值。建立几个指示同一批处理文件的 PIF(每个带有一组不同的参数值)，通常是最方便的。例如，假定已编写了 CLEANUP.BAT 批处理文件，功能是在硬盘上搜索有特定扩展名的文件，然后删去这些文件。接着可以为每一类要删除的文件建立单独的 PIF。在此

例中，为带 TMP 扩展名的所有文件建立一个 PIF，为带 BAK 扩展名的建立另一个 PIF。假定该批处理文件要求将扩展名说明成%1 参数，则此例中的两个 PIF 都有同样的 Command Line 项，但一个 PIF 在 Optional Parameters 正文框中说明了 TMP 扩展名，而另一个 PIF 说明了 BAK 扩展名。如果用此方法，一定要为每个 PIF 起不同的名字，并给每个 PIF 的图符起个描述其用途的名字。

注意：在使用 Program Item Properties 对话框来将图符赋给一批处理文件时，也能在 Commnand Line 项中说明一特定的参数值。使用该方法，不必开发一个 PIF 就可为每一任务建立不同的图符。

第二种方法是在 PIF Editor 的 Optional Parameters 正文框中输入一问号。接着，当在 Program Manager 中按下两次该批处理文件的图符，一个类似于图 13-3 的对话框就会出现。

图 13-3　当 Optional Parameters 项
为一问号时显示的对话框

在对话框的 Parameters 正文框中输入的第一个值被作为%1 参数而传递给此批处理文件。如果输入了不只一个值，则其余的值就被作为其他参数(%2，%3，等等)而传递给该批处理文件。

13.4.2　控制批处理文件的输出

缺省时，当 MS-DOS 执行完一批处理文件时，就会立即返回至 Windows。如果批处理文件产生了输出，则很可能你想看到这些信息。显示输出的最简单的方法是在批处理文件中产生信息的命令后面加上一条 PAUSE 命令。另一种方法是解除对该批处理文件的 PIF 中 Close Window On Exit 选项的选择，这样就必须使用 EXIT 命令来返回至 Windows。

用 PIF 管理 MS-DOS 会话超出了本书范围，但本章介绍了怎样使用批处理文件来改进在 Windows 中对 MS-DOS 会话的控制。

附录 A

MS-DOS 退出状态值表

若干 MS-DOS 命令提供了可用 IF ERRORLEVEL 来加以测试的退出状态值。下表列出了这些值：

命　令	退出状态值	含　义
BACKUP *	0	成功备份
	1	没有文件供备份
	2	文件共享矛盾；备份未完成
	3	用户按了 Ctrl-C；备份未完成
	4	致命错；备份未完成
CHKDSK	0	无错误
	255	发现错误
CHOICE * *	0	用户按了 Ctrl-C
	255	发现错误
DEFRAG	0	成功去除碎片
	1	处理中内部错误
	2	无自由簇；DEFRAG 至少需有一个簇
	3	用户按了 Ctrl-C；去除碎片未完成
	4	处理中一般错误
	5	磁盘读错误
	6	磁盘写错误
	7	簇分配错误；用 CHKDSK /F
	8	内存分配错误
	9	内存不够
DELTREE	0	目录及内容成功删除
DISKCOMP	0	磁盘完全相同
	1	磁盘不相同
	2	用户使用 Ctrl-C；磁盘比较未完成
	3	不可恢复的读或写错
	4	内存不够，驱动器不正确，或语法错误
DISKCOPY	0	成功拷贝
	1	非致命读或写错
	2	用户按 Ctrl-C；磁盘拷贝未完成
	3	不能读源盘或格式化目标盘
	4	内存不够，驱动器不正确，或语法错误
FIND	0	成功搜索（找到了匹配）
	1	搜索完成（没有找到匹配）

续表

命　令	退出状态值	含　义
	2	搜索时出错
FORMAT	0	成功格式化
	3	用户按 Ctrl-C；格式化未完成
	4	致命错误；格式化未完成
	5	在继续提示符处用户按了 N
GRAFTABL *	0	成功地装入表
	1	先前装入的表被替代
	2	没有新表被装入
	3	不正确的命令行参数
	4	不正确的 MS-DOS 版本
KEYB	0	成功的键盘加载
	1	不正确的命令行
	2	不正确的键盘定义表
	3 *	不能创建表
	4	CON 设备出错
	5	代码页没有准备
	6 *	没有翻译表
	7 *	不正确的 MS-DOS 版本
MOVE	0	文件被成功移动(或目录被成功换名)
	1	移动文件或更名目录出错
MSAV	86	发现病毒
REPLACE	0	成功替换
	2	源文件没有找到
	3	源路径或目标路径没有找到
	5	目标文件为只读
	8	内存不够
	11	不正确的命令行
	15 *	不正确的磁盘驱动器
RESTORE	0	成功恢复
	1	没有找到文件
	3	用户按了 Ctrl-C；恢复未完成
	4	致命错误，恢复未完成
SETVER	0	成功修改版本表
	1	不正确的命令行开关
	2	不正确的文件名
	3	内存不够
	4	不正确的版本号
	5	所说明的项未在表中发现
	6	MS-DOS 系统文件未发现
	7	无效驱动器
	8	命令行参数太多

续表

命　令	退出状态值	含　义
	9	少了命令行参数
	10	MS-DOS 系统文件读出错
	11	MS-DOS 系统文件中的版本表出错
	12	MS-DOS 系统文件不支持版本表
	13	版本表中空间不够
	14	MS-DOS 系统文件写出错
XCOPY	0	成功拷贝
	1	没找到欲拷贝的文件
	2	用户按了 Ctrl-C，拷贝未完成
	4	初始化错误(没有足够的内存，不正确的驱动器，文件或路径未找到，或语法出错)
	5	磁盘写出错

*　MS-DOS 6 及以后版本不支持

* *　其他值被赋给命令行上列出的字符

附录 B

ANSI.SYS 命令汇总

在本书中，我们大量使用了 ANSI.SYS 转义序列。下表汇总了 MS-DOS 6 及以后版本中的 ANSI.SYS 命令。左箭头←表示转义字符，即 ASCII 值 27。

序　列	功　能
←[rowsA	将光标上移 rows 行
←[rowsB	将光标下移 rows 行
←[rowsC	将光标右移 rows 列
←[rowsD	将光标左移 rows 列
←[row;colH	将光标移到 row 行，column 列位置
←[row;colf	同上
←[s	保存光标位置
←[u	恢复光标位置
←[2J	清屏并将光标放回起始位置
←[K	删除至行尾
←[color;...m	设置屏幕属性
←[=mode;...h	设置视频方式
←[=mode;...l	重置视频方式（注意最后一个字符是小写字母 L，不是大写 I 或数字 1）
←[0;key;"string"p	用串(string)来定义一功能键(key)

下表描述了设置屏幕属性的转义序列←[color m 中 color 参数的各值：

color 值	属　性	color 值	属　性
0	缺省颜色	35	洋红前景
1	粗体正文	36	青色前景
2 *	低亮度正文	37	白色背景
3 *	斜体正文	40	黑色背景
4	对 IBM 单显为下划线；对 VGA 是下划线颜色(蓝)	41	红色背景
5	闪烁正文	42	绿色背景
6 *	快速闪烁正文	43	黄色背景
7	反转视频正文	44	蓝色背景
8	被隐藏的正文	45	紫色背景
30	黑色前景	46	青色背景
31	红色前景	47	白色前景
33	黄色前景	73 *	上标
34	蓝色前景	74 *	下标

* VGA 不支持。

注意：不是所有的值都为 ANSI.SYS 的所有版本支持。

下表描述了设置视频方式的转义序列←[=mode h 中 mode 参数的可能值：

mode 值	描　述	mode 值	描　述
0	40X25 单色正文	13	320X200 16 色图形
1	40X25 16 色正文	14	640X200 16 色图形
2	80X25 单色正文	15	640X350 单色图形
3	80X25 16 色正文	16	640X350 16 色图形
4	320X200 4 色图形	17	640X480 单色图形
5	320X200 单色图形	18	640X480 16 色图形
6	640X200 单色图形	19	320X200 256 色图形
7	允许行翻滚		

注意：←[=7h 允许行翻滚，←[=7l 不允许行翻滚。

作者简介

Kris Jamsa 是一位编著了四十多本计算机书籍的多产作家，其著作的内容覆盖了范围很广的技术领域，包括 MS-DOS、硬盘管理、MS-DOS 批处理文件、Microsoft Windows、图形学、程序设计语言、WordPerfect 和 WordPerfect for Windows。他的许多书在世界上都很畅销，总共已销售了一百多万册。

Jamsa 在 Seattle 长大，然后搬到 Arizona 的 Phoenix 读高中。1983 年他获得了美国空军学会计算机科学学士学位。毕业后，Jamsa 在 Las Vegas 工作，为空军部门做一名 VAX/VMS 的系统管理员。1986 年，他获得了 Las Vegas 的 Nevade 大学的计算机科学硕士学位。1988 年，Jamsa 离开空军系统从事专职写作。现在，他正在 Arizona 州立大学攻读博士学位，研究多处理机操作系统。

作者简介

Kris Jamsa 是一位编著了四十多本计算机书籍的多产作家，其著作的内容覆盖了范围很广的技术领域，包括 MS-DOS、硬盘管理、MS-DOS 批处理文件、Microsoft Windows、图形学、程序设计语言、WordPerfect 和 WordPerfect for Windows。他的许多书在世界上都很畅销，总共已销售了一百多万册。

Jamsa 在 Seattle 长大，然后搬到 Arizona 的 Phoenix 读高中。1983 年他获得了美国空军学会计算机科学学士学位。毕业后，Jamsa 在 Las Vegas 工作，为空军部门做一名 VAX/VMS 的系统管理员。1986 年，他获得了 Las Vegas 的 Nevade 大学的计算机科学硕士学位。1988 年，Jamsa 离开空军系统从事专职写作。现在，他正在 Arizona 州立大学攻读博士学位，研究多处理机操作系统。

作者简介

Kris Jamsa 是一位编著了四十多本计算机书籍的多产作家，其著作的内容覆盖了范围很广的技术领域，包括 MS-DOS、硬盘管理、MS-DOS 批处理文件、Microsoft Windows、图形学、程序设计语言、WordPerfect 和 WordPerfect for Windows。他的许多书在世界上都很畅销，总共已销售了一百多万册。

Jamsa 在 Seattle 长大，然后搬到 Arizona 的 Phoenix 读高中。1983 年他获得了美国空军学会计算机科学学士学位。毕业后，Jamsa 在 Las Vegas 工作，为空军部门做一名 VAX/VMS 的系统管理员。1986 年，他获得了 Las Vegas 的 Nevade 大学的计算机科学硕士学位。1988 年，Jamsa 离开空军系统从事专职写作。现在，他正在 Arizona 州立大学攻读博士学位，研究多处理机操作系统。

作者简介

Kris Jamsa 是一位编著了四十多本计算机书籍的多产作家，其著作的内容覆盖了范围很广的技术领域，包括 MS-DOS、硬盘管理、MS-DOS 批处理文件、Microsoft Windows、图形学、程序设计语言、WordPerfect 和 WordPerfect for Windows。他的许多书在世界上都很畅销，总共已销售了一百多万册。

Jamsa 在 Seattle 长大，然后搬到 Arizona 的 Phoenix 读高中。1983 年他获得了美国空军学会计算机科学学士学位。毕业后，Jamsa 在 Las Vegas 工作，为空军部门做一名 VAX/VMS 的系统管理员。1986 年，他获得了 Las Vegas 的 Nevade 大学的计算机科学硕士学位。1988 年，Jamsa 离开空军系统从事专职写作。现在，他正在 Arizona 州立大学攻读博士学位，研究多处理机操作系统。

作者简介

Kris Jamsa 是一位编著了四十多本计算机书籍的多产作家，其著作的内容覆盖了范围很广的技术领域，包括 MS-DOS、硬盘管理、MS-DOS 批处理文件、Microsoft Windows、图形学、程序设计语言、WordPerfect 和 WordPerfect for Windows。他的许多书在世界上都很畅销，总共已销售了一百多万册。

Jamsa 在 Seattle 长大，然后搬到 Arizona 的 Phoenix 读高中。1983 年他获得了美国空军学会计算机科学学士学位。毕业后，Jamsa 在 Las Vegas 工作，为空军部门做一名 VAX/VMS 的系统管理员。1986 年，他获得了 Las Vegas 的 Nevade 大学的计算机科学硕士学位。1988 年，Jamsa 离开空军系统从事专职写作。现在，他正在 Arizona 州立大学攻读博士学位，研究多处理机操作系统。

作者简介

Kris Jamsa 是一位编著了四十多本计算机书籍的多产作家，其著作的内容覆盖了范围很广的技术领域，包括 MS-DOS、硬盘管理、MS-DOS 批处理文件、Microsoft Windows、图形学、程序设计语言、WordPerfect 和 WordPerfect for Windows。他的许多书在世界上都很畅销，总共已销售了一百多万册。

Jamsa 在 Seattle 长大，然后搬到 Arizona 的 Phoenix 读高中。1983 年他获得了美国空军学会计算机科学学士学位。毕业后，Jamsa 在 Las Vegas 工作，为空军部门做一名 VAX/VMS 的系统管理员。1986 年，他获得了 Las Vegas 的 Nevade 大学的计算机科学硕士学位。1988 年，Jamsa 离开空军系统从事专职写作。现在，他正在 Arizona 州立大学攻读博士学位，研究多处理机操作系统。

作者简介

Kris Jamsa 是一位编著了四十多本计算机书籍的多产作家，其著作的内容覆盖了范围很广的技术领域，包括 MS-DOS、硬盘管理、MS-DOS 批处理文件、Microsoft Windows、图形学、程序设计语言、WordPerfect 和 WordPerfect for Windows。他的许多书在世界上都很畅销，总共已销售了一百多万册。

Jamsa 在 Seattle 长大，然后搬到 Arizona 的 Phoenix 读高中。1983 年他获得了美国空军学会计算机科学学士学位。毕业后，Jamsa 在 Las Vegas 工作，为空军部门做一名 VAX/VMS 的系统管理员。1986 年，他获得了 Las Vegas 的 Nevade 大学的计算机科学硕士学位。1988 年，Jamsa 离开空军系统从事专职写作。现在，他正在 Arizona 州立大学攻读博士学位，研究多处理机操作系统。

作者简介

Kris Jamsa 是一位编著了四十多本计算机书籍的多产作家，其著作的内容覆盖了范围很广的技术领域，包括 MS-DOS、硬盘管理、MS-DOS 批处理文件、Microsoft Windows、图形学、程序设计语言、WordPerfect 和 WordPerfect for Windows。他的许多书在世界上都很畅销，总共已销售了一百多万册。

Jamsa 在 Seattle 长大，然后搬到 Arizona 的 Phoenix 读高中。1983 年他获得了美国空军学会计算机科学学士学位。毕业后，Jamsa 在 Las Vegas 工作，为空军部门做一名 VAX/VMS 的系统管理员。1986 年，他获得了 Las Vegas 的 Nevade 大学的计算机科学硕士学位。1988 年，Jamsa 离开空军系统从事专职写作。现在，他正在 Arizona 州立大学攻读博士学位，研究多处理机操作系统。

作者简介

Kris Jamsa 是一位编著了四十多本计算机书籍的多产作家，其著作的内容覆盖了范围很广的技术领域，包括 MS-DOS、硬盘管理、MS-DOS 批处理文件、Microsoft Windows、图形学、程序设计语言、WordPerfect 和 WordPerfect for Windows。他的许多书在世界上都很畅销，总共已销售了一百多万册。

Jamsa 在 Seattle 长大，然后搬到 Arizona 的 Phoenix 读高中。1983 年他获得了美国空军学会计算机科学学士学位。毕业后，Jamsa 在 Las Vegas 工作，为空军部门做一名 VAX/VMS 的系统管理员。1986 年，他获得了 Las Vegas 的 Nevade 大学的计算机科学硕士学位。1988 年，Jamsa 离开空军系统从事专职写作。现在，他正在 Arizona 州立大学攻读博士学位，研究多处理机操作系统。

作者简介

Kris Jamsa 是一位编著了四十多本计算机书籍的多产作家，其著作的内容覆盖了范围很广的技术领域，包括 MS-DOS、硬盘管理、MS-DOS 批处理文件、Microsoft Windows、图形学、程序设计语言、WordPerfect 和 WordPerfect for Windows。他的许多书在世界上都很畅销，总共已销售了一百多万册。

Jamsa 在 Seattle 长大，然后搬到 Arizona 的 Phoenix 读高中。1983 年他获得了美国空军学会计算机科学学士学位。毕业后，Jamsa 在 Las Vegas 工作，为空军部门做一名 VAX/VMS 的系统管理员。1986 年，他获得了 Las Vegas 的 Nevade 大学的计算机科学硕士学位。1988 年，Jamsa 离开空军系统从事专职写作。现在，他正在 Arizona 州立大学攻读博士学位，研究多处理机操作系统。