4DOS for Windows NT Help Index

4DOS for Windows NT On-line Documentation (Version 1.0, 4/22/93)

This on-line help system covers all 4DOS for Windows NT features and internal commands. It includes reference information to assist you in using 4DOS for Windows NT and developing batch files; however it does not include all of the details which are included in the 4DOS for Windows NT and 4DOS manuals.

The help system is fully indexed and cross-indexed. It is also context-sensitive; if you press F1 with a command already on the 4DOS for Windows NT command line, the help system will display help for that command. If the line is blank when you press F1, the help topic index is displayed.

Our product is named "4DOS", and this version is more fully described as "4DOS for the Windows NT Operating System". For brevity, in the help system we refer to "4DOS for Windows NT", or to the program file name, 4NT.EXE or just 4NT. When "4DOS" is used by itself the reference is to our DOS product.

If you are already familiar with 4DOS, please note that 4DOS for Windows NT is a complete Windows NT application, and not simply a version of 4DOS running under Windows NT's DOS capabilities.

Differences between 4DOS for Windows NT and 4DOS

Most 4DOS commands and features work the same way in 4DOS for Windows NT. However if you're an experienced 4DOS user or want to use the same batch files and aliases in both 4NT and 4DOS, there are some differences between 4DOS and 4NT you should be aware of. Those differences are listed here.

(Several items below list elements of 4DOS which are not available under 4DOS for Windows NT. In all cases these items have been removed from 4NT because they are specific to DOS or for other similar reasons are not relevant or cannot be implemented under Windows NT.)

* New directives available in 4NT.INI which are not available in 4DOS.INI include:
  LocalAliases
  LocalHistory
  WindowState

* The default command separator is **^** in 4DOS; in 4DOS for Windows NT it is **&** (for compatibility with CMD.EXE). You can change it in 4NT.INI or with the SETDOS /C command.

* The default escape separator is Ctrl-X in 4DOS; in 4DOS for Windows NT it is **^** (for compatibility with CMD.EXE). You can change it in 4NT.INI or with the SETDOS /E command.

* The alias and variable parameter syntax is **%n&** in 4DOS; in 4DOS for Windows NT it is **%n$** (because **&** is defined in 4NT as the command separator). You can change it with the ParameterChar directive in 4NT.INI or with the SETDOS /P command.

* The following 4DOS commands do not exist in 4DOS for Windows NT:
  BREAK
  CHCP
  CTTY
  KEYSTACK
  LOADHIGH / LH
  SWAPPING
  TRUENAME

* The following 4DOS internal variables and variable functions do not exist in 4DOS for Windows NT:
  %_ALIAS
  %_DV
  %_ENV
  %_MONITOR
  %_VIDEO
  %@EMS[]
  %@EXTENDED[]
  %@LPT[]
  %@REMOVABLE[]
  %@TRUENAME
  %@XMS[]

* New internal variables and variable functions available only in 4DOS for Windows NT include:
  %@FSTYPE[]
  %_PID

  See 4DOS for Windows NT Internal Variables and 4DOS for Windows NT Variable Functions for more information.

* The following 4DOS.INI directives are not used in 4NT.INI:

Alias	HelpPath
ANSI	LineInput
AutoExecPath	MessageServer
ChangeTitle	NetwareNames
CopyEA	Reduce
DiskReset	ReserveTPA
DRSets	StackSize
DVCleanup	Swapping
EnvFree	SwapReopen
Environment	UMBEnvironment
FineSwap	UMBLoad
FullINT2E	UniqueSwapName
HelpOptions	

(The alias list size and environment size are resized automatically by 4DOS for Windows NT; other directives refer to features which are not relevant under Windows NT.)

Command line editing

The 4DOS for Windows NT command line offers full editing, and can also display the command history in a window (see below). The command line accepts up to 1023 characters.

The following keys have special meaning when entered at the command line (the caret ^ means press the Ctrl key together with the specified key):

Left Moves the cursor left one character.
Right Moves the cursor right one character.
Up Recalls the previous command from the history list, or match a partial command entry with a history list entry.
Down Recalls the next command from the history list, or match a partial command entry with a history list entry.
^Left Moves the cursor left one word.
^Right Moves the cursor right one word.
Home Moves the cursor to the beginning of the line.
^Home Deletes from the beginning of the line to the character preceding the cursor.
End Moves the cursor to the end of the line.
^End Deletes from the cursor to the end of the line.
Ins Toggles between insert and overstrike mode.
Del Deletes the character at the cursor.
Backspace Deletes the character to the left of the cursor.
Enter Executes the command.
Esc Erases the entire line.
^L Deletes the word to the left of the cursor.
^R or **^Backspace** Deletes the word to the right of the cursor.
^C Aborts the command.
^D Deletes the history list entry, and erase the line.
^E Goes to the last entry in the history list.
^K Saves the command to the history list without executing it, and erase the line.
F1 Calls the on-line help.
F3 Fills in the rest of the line (beyond any characters that have already been typed) from the last line in the history.
Tab, Shift-Tab, F10 A Tab (or F9) will scan the filename (which may include the wildcard characters at or immediately to the left of the cursor position, and replaces it with the first matching filename. Pressing Tab again replaces it with the next matching filename; pressing F10 appends the next matching filename at the current cursor position. Pressing Shift-Tab (or F8) will retrieve the previous matching filename. If you don't enter a filename, Tab defaults to *.*. If you don't enter an extension, Tab appends a .*.
PgUp Popup a history window, showing those commands that match characters already entered on the command line.
^PgUp Popup a directory window, showing those directories recently changed to with the CD, CDD, and PUSHD commands.
Alt-255 (Hold down the Alt key and enter 255 on the numeric keypad.) Accept the next character "as-is" from the keyboard and enter it on the command line. This allows you to enter characters normally interpreted as editing keys (for example, Esc). To enter the Alt-255 character itself into the line, you must type it twice.
^ (caret) "Escape" the next character (the default escape character for 4NT and CMD.EXE is the caret ^; it can be changed in 4NT with the SETDOS command or in the 4NT.INI file. The escape character forces the parser to ignore any special meaning of the next character (for example, redirection characters). Note that 4NT will not recognize the escape character inside double quotes (for compatibility with CMD.EXE).

Escaping the following characters will insert the specified control code:

b bell (ASCII 7)
c comma
e escape (ASCII 27)
f form feed (ASCII 12)
n line feed (ASCII 10)
r carriage return (ASCII 13)
s space
t tab (ASCII 9)

The following keys have special meaning when entered from inside the popup history and directory windows. The history list in a window is not circular as it is at the command line; scrolling and paging stop at the beginning or end of the history list.

Up Moves the highlight up one line.
Down Moves the highlight down one line.
PgUp Moves up one page.
PgDn Moves down one page.
Home Moves to the first line in the list.
End Moves to the last line in the list.
Enter Executes the highlighted command (history window) or change to the highlighted directory (directory window).
^Enter Closes the window and move the highlighted command to the prompt for editing (history window only).
Esc Closes the window and return to the previous command line at the prompt.
^C Closes the window, abort any partially entered command line, and redisplay the prompt.

Most of the command line and popup window keys, as well as the popup window location and size, can be redefined in the 4NT.INI initialization file.

Internal and Environment Variables

Environment variables are referenced in a command by starting the variable name with a percent sign (%). (If you want to enter a percent sign without referring to a variable, use two percent signs (%%).) An environment variable name is terminated by either another percent sign, or by an invalid character (see below). To pass a variable name (instead of the value) to a command (for example, in INKEY or INPUT), you must precede it with two percent signs.

Environment variable names may be composed of any alphanumeric character, plus the underscore (_), and $ characters. In addition to the standard variable name characters, you can force 4NT to accept any sequence of characters as a valid variable name by enclosing them in square brackets. For example, %[AB##1] refers to an environment variable named AB##1.

Batch file variables are only active inside a batch file. They are referenced as %0 to %127, and expand to the matching argument on the command line that started the batch file. The parameter %n$ is a special case, and expands to all arguments in the command line tail, starting at argument number n. If n is not specified, it defaults to 1 (so %$ will expand to all arguments in the command line tail).

There are some variable names that have special meanings in 4DOS for Windows NT. Only CDPATH, CMDLINE, and PATH are actually stored in the environment; the remainder are internal to 4NT, and cannot be viewed or modified with SET or ESET. (The internal variable names are checked after variable expansion, so they can be overridden by creating an environment variable of the same name.)

The 4DOS for Windows NT internal variables are:

$= Returns the current escape character.
$+ Returns the current command separator.
%# Returns the number of command line arguments in a batch file.
%? Returns the exit code of the previous external command.
%_? Returns the exit code of the previous internal command. (Save this value immediately; it is overwritten by the next internal command.)
%CDPATH Tells 4NT where to search for directories names specified by a CD or CDD command. 4NT will append the specified directory name to each directory in CDPATH and attempt to change to that directory. If you have already have a CDPATH environment variable, you can use %_CDPATH instead for the directory search.
%CMDLINE Is the fully expanded 1023-character command line for the external command. You can access it from an external program by searching the environment.
%COLORDIR Is the "directory colorization" variable. See DIR for details on the COLORDIR format.
%PATH Tells 4NT where to search for executable files: .COM, .EXE, .BTM, .BAT, and .CMD (and optionally .REX), in that order, not found in the current directory. Some applications also use the PATH variable to find their files.
%_4VER Returns the 4NT version number (for example, 1.0).
%_ANSI Returns 1 if an ANSI screen driver is active (4NT will return 0 because ANSI support is not currently available in Windows NT).
%_BATCH Returns the current batch nesting level (0 if not in a batch file).
%_BG Returns a string containing the screen background color at the current cursor position.
%_BOOT Returns the boot disk (for example, C).
%_CODEPAGE Returns the current code page number.
%_COLUMN Returns the current cursor column.
%_COLUMNS Returns the current number of screen columns.
%_COMSPEC Returns the full pathname of the current command shell (4NT.EXE).
%_CPU Returns the cpu type (386 or 486).
%_CWD Returns the current directory in the format d:\pathname.
%_CWDS Has the same value as CWD, except it ensures the pathname ends in a backslash (\).

%_CWP Returns the current directory in the format \pathname.
%_CWPS Has the same value as CWP, except it ensures the pathname ends in a backslash (\).
%_DATE Returns the current system date, in the format mm-dd-yy (U.S.), dd-mm-yy (Europe), or yy-mm-dd (Japan).
%_DAY Returns the current day (1 - 31).
%_DISK Returns the current disk (for example, C).
%_DOS Returns the operating system type (DOS, OS2, or NT). This is useful if you have .BTM files running in different environments..
%_DOSVER Returns the current Windows NT version (for example, 3.1).
%_DOW Returns the current day (Mon, Tue, Wed, etc.).
%_FG Returns a string containing the screen foreground color at the current cursor position.
%_HOUR Returns the current hour (0-23).
%_LASTDISK Returns the last disk on the system (A: - Z:).
%_MINUTE Returns the current minute (0 - 59).
%_MONTH Returns the current month (1 - 12).
%_MOUSE Returns 1 if a mouse is installed.
%_NDP Returns the coprocessor type (0 - no coprocessor; 387 - 80387 or 80486).
%_PID Returns the process ID for 4NT.
%_ROW Returns the current cursor row.
%_ROWS Returns the current number of screen rows.
%_SECOND Returns the current second (0 - 59).
%_SHELL Returns the current shell nesting level for that 4NT window. The first copy of 4NT is shell 0.
%_TIME Returns the current system time in the format hh:mm:ss. The separator character may vary depending upon your country information.
%_TRANSIENT Returns a 1 if the current shell is transient (started with a /C).
%_YEAR Returns the current year (1980-2099).

Variable Functions

Variable Functions are pseudo-variables that take one or more arguments (which can themselves be environment variables or variable functions), and return a value. The variable function name must be preceded by an %@ (%@eval, %@len, etc.). All variable functions must have square brackets enclosing their argument(s).

Some functions return the number of bytes, kilobytes, or megabytes based on a "b|k|m" argument. If you append a '**c**' to the "b|k|m" argument, 4NT will format the returned argument by inserting commas.

- **b** bytes
- **k** bytes / 1000
- **K** kilobytes (bytes / 1024)
- **m** bytes / 1,000,000
- **M** megabytes (bytes / 1,048,576)

The 4DOS for Windows NT variable functions are:

%@ALIAS[name] returns the alias argument for the specified name.

%@ASCII[c] returns the ASCII value of the specified character.

%@ATTRIB[filename,attrib] returns a 1 if the specified file has the matching attribute(s). The attributes (other than N) can be combined; ATTRIB will only return a 1 if all the attributes match. The attributes are:

- **N** Normal (no attribute bits set)
- **R** Read-only
- **H** Hidden
- **S** System
- **D** Directory
- **A** Archive

%@CHAR[n] returns the character for the specified ASCII value.

%@DATE[mm/dd/yy] returns the number of days since 1/1/80 for the specified date. DATE will use the date format mandated by your country code (dd/mm/yy in Europe; yy/mm/dd in Japan).

%@DESCRIPT[filename] returns the file's description.

%@DEVICE[name] returns a 1 if the specified name is a character device.

%@DISKFREE[d:,b|k|m] returns the free disk space for the specified drive, in bytes, kilobytes, or megabytes.

%@DISKTOTAL[d:,b|k|m] returns the total disk space for the specified drive, in bytes, kilobytes, or megabytes.

%@DISKUSED[d:,b|k|m] returns the disk space used on the specified drive, in bytes, kilobytes, or megabytes.

%@DOSMEM[b|k|m] returns the size of the largest free Windows NT memory block in bytes, kilobytes, or megabytes. Due to Windows NT's virtual memory system, this value is only a general reflection of available memory and not a precise statement of the RAM available for applications.

%@EVAL[expression] evaluates an arithmetic expression. It supports addition (+), subtraction (-), multiplication (*), division (/), and modulo (%%). The expression can contain environment variables, including other variable functions. EVAL supports commas and decimal places; the maximum size is 16.8 (16 integer and 8 decimal places). EVAL strips leading and trailing zeros from the result.

%@EXEC[command] executes the command and returns its exit code. The command can be an alias, an internal 4NT command, or an external program or batch file. This is a back door entry to 4NT command processing -- **USE WITH CAUTION!**

%@EXT[filename] returns the file extension (without a leading period).

%@FILEDATE[filename] returns the last modification date for the file, in the default country format (mm-dd-yy for US).

%@FILESIZE[filename,b|k|m] returns the size of the file in bytes, kilobytes, or megabytes, or -1 if the file doesn't exist.
%@FILETIME[filename] returns the last modification time for the file, in hh:mm format.
%@FULL[filename] returns the fully qualified path name.
%@FSTYPE[d:] returns the drive's file system type (for example, FAT, HPFS, NTFS, and CDFS).
%@INDEX[string1,string2] returns the position of string2 within string1, or -1 if string2 is not found. The first position in string1 is 0.
%@INT[n] returns the integer part of "n". See %@EVAL.
%@LABEL[d:] returns the volume label of the specified disk.
%@LEN[string] returns the length of the string.
%@LINE[filename,n] returns line "n" from the specified file. If you specify "CON" for the filename, it will read from standard input.
%@LINES[filename] returns the number of lines in the specified file, base 0. If there are no lines, it returns -1.
%@LOWER[string] returns the string converted to lower case.
%@MAKEDATE[n] returns the date given the number of days since 1/1/80.
%@MAKETIME[n] returns the time given the number of seconds since midnight.
%@NAME[filename] returns the filename only (no path or extension).
%@PATH[filename] returns the path only (including the trailing backslash).
%@READSCR[row,column,length] returns the text on the screen at the specified row and column, for the specified length.
%@READY[d:] returns a 1 if the specified drive is ready.
%@REMOTE[d:] returns a 1 if the specified drive is remote (LAN).
%@SEARCH[filename] searches for an executable file using the PATH environment variable, appending the extension if one wasn't specified.
%@SELECT[filename,top,left,bottom,right,title] displays the file as a popup selection list. If you press Enter, it returns the string highlighted by the selection bar; if you press Esc it returns an empty string. If you specify "CON" for the filename, it will read from standard input.
%@SUBSTR[string,start,length] returns a substring, starting at the position "start" and continuing for "length" characters. If the length is negative, the start is relative to the right side. If the length isn't specified, SUBSTR will return the remainder of the string. For example, @SUBSTR[%_time,0,2] gets the current time and extracts the hour. If the string includes commas it must be quoted with double quotes or single back-quotes (" or `).
%@TIME[hh:mm:ss] returns the number of seconds since midnight for the specified time. The time must be in 24-hour format.
%@UNIQUE[d:\path] returns a unique filename in the specified drive and path.
%@UPPER[string] returns the string converted to upper case.
%@WORD[n,string] returns the "nth" word in the string (base 0). If "n" is negative, WORD starts from the end and counts backwards.

Redirection

In 4DOS for Windows NT input comes from the keyboard and output goes to the display. The keyboard is referred to as standard input, and the display is referred to as standard output. You can change the default standard input and standard output by using the < and > symbols on the command line. 4NT also allows you to redirect the standard error by appending an & character.

To get standard input from a file instead of the keyboard:

    < filename

To redirect standard output to a file:

    > filename

To redirect standard output and standard error to a file:

    >& filename

To redirect standard error only to a file:

    >&> filename

To append standard output or standard error to a file, use >> in place of the first >. If NOCLOBBER is set, the file must exist before it can be appended to (unless overridden by a !). Otherwise, 4NT will create the file.

4NT also supports the CMD.EXE syntax:

    n>file

and

    n>&m

where n and m are digits from 0 to 9. You may not put any spaces between the n and the >, or between the & and the m in the second form. The digits represent file handles: 0, 1, and 2 are predefined as standard input, standard output, and standard error respectively; 3 through 9 can be defined by applications.

The n>file syntax redirects output from handle n to a file. The n>&m syntax redirects handle n to the same location as the previously assigned handle m. In many cases you can perform the same operations by using 4NT's enhanced redirection features. See the 4DOS for Windows NT and 4DOS manuals for more information on these options.

Piping

Piping makes the standard output of one command the standard input for a second command. To send the standard output of "cmd1" to the standard input of "cmd2":

    cmd1 | cmd2

To send the standard output and standard error of "cmd1" to the standard input of "cmd2":

```
cmd1 |& cmd2
```

Unlike 4DOS, which creates a temporary file for pipes, 4NT connects a pipe directly to a secondary process; both processes execute simultaneously.

Command Grouping

Command grouping allows you to logically group a set of commands together by enclosing them with parentheses. For example, this grouping:

```
(global /iq echo %_cwd) > dirlist
```

allows you to redirect all output from GLOBAL to the specified file. The following command will take the output of DIR and TYPE, sort the whole mess, and save it in the file AZ:

```
(dir *.com & type autoexec.bat) | sort > AZ
```

To group commands, the first character of the command must begin with a (Note that if you have a closing parenthesis) in a filename, you'll need to escape it with the 4NT escape character (^) to keep 4NT from assuming it's the end of the command group.

You can also use command grouping to split commands over several lines. For example:

```
for %x in (*.c) do (
      echo %x
      dir *.c
)
```

If you enter a command group on the command line without a closing parenthesis, 4NT will prompt you with **More?** for the remainder of the line.

Wildcards

Wildcards let you specify a file or group of files by typing a partial filename. Most internal commands accept filenames with wildcards anywhere a full filename can be used. 4DOS for Windows NT recognizes 3 wildcard types: asterisk *, question mark ?, and square brackets [ ]. You can combine various wildcard types in a single filename. Note that the 4DOS for Windows NT wildcard extensions may not be interpreted properly by external programs, so when passing file names to external programs you should use the traditional wildcard syntax.

A * in a filename means "any zero or more characters in this position." You can use both leading and trailing asterisks. For example, to get a directory of all files beginning with "ab":

```
dir ab*.*
```

To get a directory of all files with an "ab" anywhere in the filename:

```
dir *ab*.*
```

A ? matches any single filename character. A ? doesn't require that a character exist if it's at the end of the filename or extension.

A set of characters in brackets [ ] acts like a ? (match any single character), but allows you to specify or exclude specific characters. If you specify one or more characters in the brackets (e.g. [abc]), the specification will only match files whose names have one of the listed characters in the corresponding position. For example, to get a directory of all files whose names have either a "p" or "q" as the first letter:

```
dir [pq]*.*
```

Two characters with a dash (-) between them represent a range of characters, and will match any character within the range (including the beginning and end characters). For example, to get a directory of all files with names beginning with a number from 1 to 6, or the letters "a" or "b", and ending with a "9":

```
dir [1-6ab]*9.*
```

If you enter a ! as the first character following the left bracket, the test is reversed (only match names that do not have the listed characters in the corresponding position). For example, to get a directory of all files with names of two characters or more that begin with an "a" and do not have a digit as the second character:

```
dir a[!0-9]*.*
```

The special wildcard construct [?] will match any character (like ?) but will require that the character exist, even at the end of a filename or extension. For example, if you have the files XYZ.DAT and XY.DAT in the current directory, then:

```
dir xy?.dat          will display both file names
dir xy[?].dat        will display only XYZ.DAT
```

4DOS for Windows NT Initialization FILE (4NT.INI)

The 4NT.INI file is an ASCII file containing directives to control the configuration of 4DOS for Windows NT in primary and secondary shells. Blank lines are ignored and can be used to separate groups of directives. You can place comments in the file by beginning a line with a semicolon (;). You can also place comments at the end of any line except one containing a text string value by entering at least one space or tab after the value, a semicolon, and your comment.

The file has two sections, identified by a name in square brackets on a line by itself. The section names are:

[Primary] Directives in this section will be used when 4DOS for Windows NT is running as the primary shell (i.e., when a new 4NT window is started).

[Secondary] Directives in this section are used in secondary shells only (i.e., a second invocation of 4NT within the same window), and override any corresponding primary shell settings.

Lines before a section name are used in both primary and secondary shells.

When 4DOS for Windows NT is loaded, whether as the primary or secondary shell, it first checks for an "@d:\path\inifile" option on the command line. If this option is found 4NT looks for the specified file, and skips the search for the default 4NT.INI file.

If no INI file name is specified on the command line, 4NT looks for 4NT.INI in the same directory as 4NT itself, then in the root directory of the system boot drive.

Directives in 4NT.INI are divided into four types:

See the individual topics for details.

Initialization Directives

These directives control how 4DOS for Windows NT starts and where it looks for its files.

4StartPath = Path: Sets the drive and directory where 4NT will look for the 4START and 4EXIT batch files.

History = nnnn (1024): Sets the history list size (in bytes). The range is 512 to 8192 bytes.

LocalAliases = Yes | No: "Yes" tells 4NT to create a local alias list for this 4NT window. "No" causes 4NT to use the global alias list shared by all active 4NT windows.

LocalHistory = Yes | No: "Yes" tells 4NT to create a local history list for this 4NT window. "No" causes 4NT to use the global history list shared by all active 4NT windows.

LogName = File (none): Sets the log file name and/or path. If only a path is given, 4NT will use the default log file name (4NTLOG). Using LogName does not turn logging on, you must still use LOG ON to do so.

NextINIFile = File (none): Full path and name must be specified. All subsequent shells will read the specified INI file, and ignore any [Secondary] section in the original 4NT.INI. Allows workstation users to shift 4NT.INI to a network drive for secondary shells.

PauseOnError = Yes | No: "Yes" tells 4NT to pause with the message "Error in 4NT.INI, press any key to continue processing" after displaying any error message related to a specific line in the 4NT.INI file. "No" continues processing with no pause after an error message is displayed.

WindowState = Standard | Maximize | Minimize: Switches the 4NT window to a maximized or minimized state at startup. "Standard" means leave the window wherever Windows NT puts it.

Configuration Directives

These directives control the way that 4DOS for Windows NT operates. Some can be changed with the SETDOS command while 4DOS for Windows NT is running.

AmPm = Yes | No | Auto: Sets the time display (in DIR, PROMPT, SELECT, and TIME) to 12-hour (am/pm) format or 24-hour format. AUTO sets the time format to the default format for your country code.

BatchEcho = Yes | No: Sets the default batch ECHO mode. Also see SETDOS /V.

BeepFreq = nnnn (440): Sets the default BEEP command frequency in Hz. To disable all 4NT error beeps set this or BeepLength to 0;

BeepLength = nnnn (2): Sets the default BEEP length in system clock ticks (approximately 1/18 of a second per tick).

CursorIns = nnnn (100): Sets the percentage of the character cell filled by the 4NT cursor in insert mode. Also see SETDOS /S.

CursorOver = nnnn (10): Sets the percentage of the character cell filled by the 4NT cursor in overtype mode. Also see SETDOS /S.

CommandSep = c (&): This is the character used to separate multiple commands on the same line. Also see SETDOS /C.

DescriptionMax = nnnn (40): Sets the maximum file description length accepted by DESCRIBE; the range is 20 - 120 characters.

Descriptions = Yes | No: Specifies whether Osprey should process descriptions for file handling commands (COPY, DEL, MOVE, etc.). See also SETDOS /D.

EditMode = Insert | OVERSTRIKE: Lets you start the command line editor in either insert or overstrike mode. Also see SETDOS /M.

EscapeChar = c (^): Sets the character used to suppress the normal meaning of the following character. Also see SETDOS /E.

HistMin = nnnn (0): Sets the minimum command length to save in the history list. 0 saves all commands, 300 disables all history saves.

HistWinColor = Color: Sets the default colors for the command line history window.

HistWinHeight = nn (10): Sets the height of the command line history window in lines, including the border.

HistWinLeft = nn (50): Sets the horizontal position of the left side of the command line history window. The left edge of the screen is 0.

HistWinTop = nn (0): Sets the vertical position of the top of the command line history window. The top of the screen is 0.

HistWinWidth = nn (30): Sets the width of the command line history window in characters, including the border.

NoClobber = Yes | No: If set to Yes, prevents standard output redirection from overwriting an existing file, and requires the output file already exist for append redirection. Also see SETDOS /N.

ParameterChar = c ($): Sets the character used to specify all or all remaining command line arguments (e.g. %$ or %2$). The default is $ in 4NT and & in 4DOS. Also see SETDOS /P.

UpperCase = Yes | No: "Yes" specifies filenames should be displayed in the traditional upper-case by internal commands like COPY and DIR. "No" allows the normal 4NT lower-case style. Also see SETDOS /U.

Color Directives

These directives control the colors that 4DOS for Windows NT uses for its displays. The color format is:

[BRIght] fg ON [BRIGHT] bg

Where fg is the foreground color and bg is the background color. The color names are:

Black	Blue	Green	Red
Magenta	Cyan	Yellow	White

The color names and the keyword BRIGHT can be shortened to three letters.

ColorDir = ext1 ext2 ...:colora;ext3 ext4 ... :colorb ... (none): Sets the directory colors. The format is the same as that used for the COLORDIR environment variable (see 4DOS for Windows NT Internal Variables).

ListColors = Color: Sets the colors used by the LIST and SELECT commands.

StdColors = Color: Sets the standard colors to be used when CLS is used without a color specification, and for LIST and SELECT if ListColors is not used.

Key Mapping Directives

The directives in this group allow you to change the keys used for 4DOS for Windows NT command line editing and other internal functions. They take effect only inside 4NT itself and do not affect other programs (including the help system). 4NT processes all command line editing key assignments before looking for keystroke aliases. For example, if you assign Shift-F1 to HELP and also assign Shift-F1 to a key alias, the key alias will be ignored. (Use the "Normal" directives -- NormalKey, NormalEditKey, etc. -- to disable the preassigned function for a key so it can be used for a keystroke alias.)

There are three pre-mapped keys: Tab and Shift-Tab (mapped to NextFile and PrevFile respectively) and Ctrl-Bksp (mapped to DelWordRight). If you need to clear out these assignments so you can assign Tab, Shift-Tab, or Ctrl-Bksp to a keystroke alias, use the ClearKeyMap directive described at the end of this section.

The format for a key name is: [Prefix-]Keyname. The prefix and key name must be separated by a dash. The prefix can be left out, or it can be any of the following:

Alt followed by A - Z, 0 - 9, F1 - F12, or Bksp
Ctrl followed by A-Z, F1-F12, Bksp, Enter, Left, Right, Home, End, PgUp, PgDn, Ins, or Del
Shift followed by F1 - F12 or Tab

The possible key names are:

A - Z	Esc	Up	PgUp
0 - 9	Bksp	Down	PgDn
F1 - F12	Tab	Left	Home
Ins	Enter	Right	End
Del			

General Input Keys

These directives are effective whenever 4DOS for Windows NT requests input from the keyboard, including command line editing and the DESCRIBE, ESET, INPUT, LIST, and SELECT commands. (Scrolling through the command history list is controlled by NextHist and PrevHist (see below), not by the Up and Down directives below.)

Backspace = Key (Bksp): Deletes the character to the left of the cursor.
BeginLine = Key (Home): Moves the cursor to the beginning of the line.
Del = Key (Del): Deletes the character at the cursor.
DelToBeginning = Key (Ctrl-Home): Deletes from the cursor to the start of the line.
DelToEnd = Key (Ctrl-End): Deletes from the cursor to the end of the line.
DelWordLeft = Key (Ctrl-L): Deletes the word to the left of the cursor.
DelWordRight = Key (Ctrl-R, Ctrl-Bksp): Deletes the word to the right of the cursor. See ClearKeyMap at the end of this section if you need to remove the mapping of Ctrl-Bksp to DelWordRight.
Down = Key (Down): Scrolls the display down one line in LIST; moves the cursor down one line in SELECT and in the command history window.
EndLine = Key (End): Moves the cursor to the end of the line.
EraseLine = Key (Esc): Deletes the entire line.
ExecLine = Key (Enter): Executes or accepts a line.
Ins = Key (Ins): Toggles insert / overstrike mode during line editing.
Left = Key (Left): Moves the cursor left one character; moves the display left 8 columns in LIST.
NormalKey = Key (none): Deassigns a general input key in order to disable the usual meaning of the key within 4NT and / or make it available for keystroke aliases. This will cause 4NT to treat

the keystroke as a "normal" key with no special function.

Right = Key (Right): Moves the cursor right one character; scrolls the display right 8 columns in LIST.

Up = Key (Up): Scrolls the display up one line in LIST; moves the cursor up one line in SELECT and in the command history window.

WordLeft = Key (Ctrl-Left): Moves the cursor left one word; scrolls the display left 40 columns in LIST.

WordRight = Key (Ctrl-Right): Moves the cursor right one word; scrolls the display right 40 columns in LIST.

Command Line Editing Keys

The following directives apply only to command line editing (see Command Line Editing). They are only effective at the 4DOS for Windows NT prompt.

AddFile = Key (F10): Keeps the current filename completion entry and inserts the next matching name.

CommandEscape = Key (Alt-255): Allows direct entry of a keystroke that would normally be interpreted as an editor command.

DelHistory = Key (Ctrl-D): Deletes the displayed history list entry and displays the previous entry.

EndHistory = Key (Ctrl-E): Displays the last history list entry.

Help = Key (F1): Invokes the 4NT HELP facility.

NextFile = Key (F9, Tab): Gets the next matching filename. See ClearKeyMap at the end of this section if you need to remove the mapping of Tab to NextFile.

NextHistory = Key (Down): Recalls the next command from the command history.

NormalEditKey = Key (none): Deassigns a command line editing key in order to disable the usual meaning of the key while editing a command line and / or make it available for keystroke aliases. This will cause 4NT to treat the keystroke as a "normal" key with no special function.

PrevFile = Key (F8, Shift-Tab): Gets the previous matching filename. See ClearKeyMap at the end of this section if you need to remove the mapping of Shift-Tab to PrevFile.

PrevHistory = Key (Up): Recalls the previous command from the command history.

SaveHistory = Key (Ctrl-K): Saves the command line in the history list without executing it.

History Window Keys

HistWinBegin = Key (Ctrl-PgUp): Moves to the first line of the history when in the history window.

HistWinEdit = Key (Ctrl-Enter): Moves a line from the history window to the prompt for editing.

HistWinEnd = Key (Ctrl-PgDn): Moves to the last line of the history when in the history window.

HistWinExec = Key (Enter): Executes the selected line in the history window.

HistWinOpen = Key (PgUp): Brings up the history window while at the command line.

NormalHWinKey = Key (none): Deassigns a history window key in order to disable the usual meaning of the key within the history window. This will cause 4NT to treat the keystroke as a "normal" key with no special function.

LIST Keys

These directives control the keystrokes used within the LIST command.

ListFind = Key (F): Prompts and searches for a string.

ListHighBit = Key (H): Toggles LIST's "strip high bit" option, which can aid in displaying files from certain word processors.

ListNext = Key (N): Finds the next matching string.

ListPrint = Key (P): Prints the file on LPT1.

ListWrap = Key (W): Toggles LIST's wrap option on and off. The wrap option wraps text at the right margin.

NormalListKey = Key (none): Deassigns a LIST key in order to disable the usual meaning of the key within LIST. This will cause 4NT to treat the keystroke as a "normal" key with no special function.

ClearKeyMap Directive

ClearKeyMap: Clears all current key mappings. ClearKeyMap is a special directive which has no value or "=" after it. Use ClearKeyMap to make one of the keys in 4NT's default map (Tab, Shift-Tab, or Ctrl-Bksp) available for a keystroke alias; or in the [Secondary] section to clear key mappings inherited from the primary shell. ClearKeyMap should appear before any other key mapping directives. If you only need to clear some of the default mappings, use ClearKeyMap, then recreate the mappings you want to retain (e.g. with "Tab=NextFile", etc.).

The following example gives you an idea of the types of things that can be done with the 4NT.INI file. The comments on each directive explain what it does.

```
History = 2048          ; set history size
BatchEcho = No          ; default is ECHO OFF
EditMode = Insert       ; editor in insert mode
CursorO = 100           ; overstrike cursor 100%
CursorI = 10            ; insert cursor 10%
ListFind = S            ; use "S" to search in LIST
ListNext = A            ; use "A" to search again in LIST
```

4DOS for Windows NT and Personal REXX for Windows NT (Quercus Systems)

4NT includes internal support for Personal REXX for Windows NT (available from Quercus Systems, P.O. Box 2157, Saratoga, CA 95070, phone 408-867-REXX, or GO QUERCUS on CompuServe).

When 4NT starts, it searches for WREXX32.DLL, and if found, adds the .REX extension to the list of executable files. (The search order is .COM, .EXE, .BTM, .BAT, .CMD, and .REX.) 4NT also checks the first line of .CMD files, and if they begin with a /* 4NT will invoke REXX to execute them.

4NT works as the default REXX subcommand handler (making it unnecessary to call another copy of the command processor to run internal commands such as DIR or COPY), and handles REXX input and output in the 4NT console window.

SYNTAX (External Windows NT)

[[d:]path]4NT [/C /K /L /LA /LH /Q /S][@inifile][command ...]

PURPOSE

Start a new copy of the 4DOS for Windows NT command processor.

COMMENTS

4DOS for Windows NT replaces CMD.EXE, the default command processor provided with Windows NT. The 4DOS for Windows NT options are:

/C	Execute the command following the /C, and automatically return to the previous command processor.
/K	Execute the command following the /K, but do not return to the previous command processor. 4NT defaults to /K if /C is not specified.
/L	Use local history and alias lists.
/LA	Use local alias list. 4NT normally shares its alias list with all active 4NT windows; specifying /LA shares it only within the same window.
/LH	Use local history list. 4NT normally shares its history list with all active 4NT windows; specifying /LH shares it only within the same window.
/Q	Don't echo redirected standard input.
/S	Don't install a ^C signal handler.
@inifile	The 4NT.INI file specification.
command	The command to execute..

EXAMPLES

Start 4DOS for Windows NT with local alias and history lists, using the 4NT.INI file in D:\4NT:

```
4NT @D:\4NT\4NT.INI /LA /LH
```

SYNTAX ***(Internal 4NT)***

?

PURPOSE

Display the internal commands. Commands disabled with SETDOS /I will not be displayed.

SYNTAX (Internal 4NT)

ALIAS [/P] [/R [d:][path]filename...] [name[=][value]]

PURPOSE

Load or display the alias list, or define name as a substitute for value.

COMMENTS

Aliases are useful as a means of executing a complex series of commands with a few keystrokes. Aliases can also be used as very fast in-memory batch files, and will run much faster than disk-based batch files.

4DOS for Windows NT supports two types of aliases; command aliases (where the alias is substituted for the first argument on the command line), and "keystroke aliases", where the command line editor will immediately substitute the alias when the key is pressed.

Keystroke alias names are composed of an @ followed by the key name. You can add a carriage return to a key alias by ending the alias with a ^^r (^ is the 4NT escape character) if defined at the command line or in a batch file, or ^r if defined in an ALIAS /R file. The valid key names are the same as those listed in the section on Key Mapping Directives in Extended Key Codes.

If you only specify "name", ALIAS displays the current alias value for "name". Otherwise, ALIAS assigns the command(s) in "value" to "name". "Name" can now be used as if it were a built-in or external command. If you don't specify any arguments, ALIAS displays the current alias list.

"Name" is limited to no more than 80 characters, and "value" to no more than 255 characters.

The ALIAS options are:

/P Pause after displaying each page and wait for a key to be pressed.
/R Load an alias list from a file. This is much faster than loading aliases from a batch file. The file is in the same format as the ALIAS display, so ALIAS /R can accept as input a file generated by redirecting ALIAS output. You can add comments to an alias file by starting the comment line with a colon (:). If the last character on a line is an escape character (default ^X), 4NT will remove the escape character and append the next line. (This allows you to write your aliases in a more readable format.)

For example, the following commands will save the aliases to a file, and then reload them from that file:

```
alias > alist
alias /r alist
```

(You can load aliases from multiple files by listing all the filenames after the /R.)

When defining aliases at the command line, back quotes must be used around the alias arguments that contain multiple commands or variable references (%2, %2$, etc.) to prevent premature expansion. Backquotes should NOT be used when defining aliases in a file to be read with ALIAS /R.

Aliases may be nested; i.e., an alias can refer to another alias, but they cannot refer back to themselves (a=b=a). You can stop alias expansion by prefacing the alias with an asterisk (*). This also allows an alias to refer to a command of the same name (see below for an example).

Alias names can be truncated by including an asterisk (*) in the name.

To edit an alias, use ESET; to remove an alias, use UNALIAS.

For more information on aliases (including the use of variables), see the Aliases section in the printed manual, and the sample file ALIASES distributed with 4DOS for Windows NT.

See also ESET and UNALIAS.

EXAMPLES

Define D as an alias for DIR /AP:

```
alias d dir /ap
```

Rename the LIST command to DISPLAY, and alias LIST to an external program:

```
alias display *list
alias list c:\util\list.com
```

The following examples show the use of alias arguments:

```
alias zap `erase %$ & chkdsk & dir /w`
alias reverse `echo %5 %4 %3 %2 %1`
```

The following keystroke alias will insert a "dir /2 /v /p " on the command line when the Alt-F1 key is pressed:

```
alias @Alt-F1 `dir /2 /v /p `
```

The same alias set up to execute immediately when Alt-F1 is pressed.

```
alias @Alt-F1 `dir /2 /v /p^r`
```

SYNTAX (Internal 4NT)

ATTRIB [/D /P /Q /S -|+[AHRS]] [d:][path]filename...

PURPOSE

Change the file or subdirectory attributes.

COMMENTS

The ATTRIB options are:

/D	Also modify subdirectory attributes
/P	Pause after displaying each page
/Q	Quiet mode - don't display filenames
/S	Modify files in the current directory and its subdirectories
+A	Set the archive attribute
-A	Clear the archive attribute
+H	Set the hidden attribute
-H	Clear the hidden attribute
+R	Set the read-only attribute
-R	Clear the read-only attribute
+S	Set the system file attribute
-S	Clear the system file attribute

If you don't specify the /D option, ATTRIB only modifies file attributes. If you don't specify any attributes to change, ATTRIB displays the current file attributes. You can also display the file attributes using the /T option in DIR.

ATTRIB will preserve the previous file attributes and change only the specified attributes. New attribute values are allowed between filenames; otherwise ATTRIB uses the same attributes specified for the previous file(s). You cannot modify the directory or volume label attributes.

EXAMPLES

Set the read-only and hidden attributes for the file MEMO:

```
attrib +rh memo
```

Set the archive attribute (file has been modified) for MEMO and change TEXT.COM to system and not modified:

```
attrib +a memo +s -a test.com
```

SYNTAX (Internal 4NT)

BEEP [frequency [duration] ...]

PURPOSE

Beep the speaker.

COMMENTS

BEEP is normally used in batch files to signal that an operation has been completed, or that the computer needs attention (for example, to change disks). Because BEEP allows you to specify the frequency and duration, you can use it to play simple music. You can specify multiple frequency and duration pairs on the command line.

The frequency is specified in Hertz, and the duration in 1/18th second intervals. No sound will be generated for frequencies less than 37 Hz, allowing you to insert short delays. The default value for frequency is 440 Hz; the default value for duration is 2.

The following table gives the frequency values for a five octave range (middle C is 262 Hz):

C	131	262	523	1046	2093
C#/Db	139	277	554	1108	2217
D	147	294	587	1175	2349
D#/Eb	156	311	622	1244	2489
E	165	330	659	1318	2637
F	175	349	698	1397	2794
F#/Gb	185	370	740	1480	2960
G	196	392	784	1568	3136
G#/Ab	208	415	831	1662	3322
A	220	440	880	1760	3520
A#/Bb	233	466	932	1866	3729
B	248	494	988	1973	3951

EXAMPLES

The following batch file fragment runs the program DEMO, plays a few notes, and waits for you to press a key:

```
demo
beep 440 4 587 2 1040 6
pause Finished with the demo - press a key to continue
```

SYNTAX (Internal 4NT)

CALL [d:][path]filename

PURPOSE

Call a secondary batch file.

COMMENTS

CALL allows batch files to call other batch files (batch file nesting) without invoking a secondary copy of the command processor. The calling batch file is suspended while the called batch file runs. When the called batch file finishes, the calling batch file resumes execution at the next command. If you execute a batch file from another batch file without using CALL, the first batch file is terminated before the second one starts.

The current ECHO state will be inherited by a called batch file.

See also CANCEL and QUIT.

EXAMPLES

The following batch file fragment compares an input line to "wp" and calls a secondary batch file if it matches:

```
input Enter your choice:  %%option
if "%option" == "wp" call wp
```

SYNTAX (Internal 4NT)

CANCEL [value]

PURPOSE

Terminate batch processing.

COMMENTS

The CANCEL command will end all batch file processing, regardless of the batch nesting level. (Use QUIT to end a nested batch file and return to the previous batch file.) If you enter a value, CANCEL will set the ERRORLEVEL to that value. The value also affects the internal variable %_?, and the conditional command separators && and ||.

You can CANCEL at any point in a batch file.

See also CALL and QUIT.

EXAMPLES

The following batch file fragment compares an input line to "end" and terminates all batch file processing if it matches:

```
input Enter your choice:  %%option
if "%option" == "end" cancel
```

SYNTAX (Internal 4NT)

CD [[d:][pathname]
CHDIR [[d:][pathname]

PURPOSE

Display or change the current directory.

COMMENTS

Entering CD with no argument or only a drive name will display the current directory. Entering CD and a pathname will change the current directory.

If CD can't change to the specified directory, it will look for the CDPATH (or _CDPATH) environment variable. CD will append the specified directory name to each directory in CDPATH and attempt to change to that directory, until the first match or the end of the CDPATH argument.

The previous directory is saved on each CD, and you can switch back to it with "CD -". You can switch between two directories by repeatedly entering "CD -". Note that the saved directory is the same for both CD and CDD.

You can change directories by pressing ^PgUp at the prompt, which will display a popup window of the most recent directory changes; you can select a directory and switch to it by pressing Enter.

You can change to the parent directory with "CD ..". You can also go up one additional directory level with each additional ".". For example, "CD ...." will go up three directory levels.

Every disk drive on the system has its own current directory. Specifying both a drive and a directory in the CD command will change the current directory on the specified drive, but will not change the default drive. Use CDD to change both the drive and directory.

See also %CDPATH, CDD, and PUSHD.

EXAMPLES

Change to the subdirectory C:\FINANCE\MYFILES:

```
cd \finance\myfiles
```

Change the default directory on drive A:

```
cd a:\utility
```

Set the CDPATH environment variable and then CD to a directory called DOCS (CD will first attempt to change to DOCS in the current directory, then C:\DOCS, then C:\OS\DOCS, etc.):

```
set cdpath=c:\;c:\os;c:\util;c:\wp
cd docs
```

SYNTAX (Internal 4NT)

CDD [d:]pathname

PURPOSE

Change the current disk drive and directory.

COMMENTS

CDD is similar to CD, except it can also change the default disk drive.

To start at the root directory, start the pathname with a backslash (\). To start at the parent directory, start the pathname with two periods (..). All other pathnames start at the current directory.

If CDD can't change to the specified directory, it will look for the CDPATH (or _CDPATH) environment variable. CDD will append the specified directory name to each directory in CDPATH and attempt to change to that directory, until the first match or the end of the CDPATH argument.

The previous directory is saved on each CDD, and you can switch back to it with "CDD -". You can switch between two directories by repeatedly entering "CDD -". Note that the saved directory is the same for both CDD and CD.

You can change directories by pressing ^PgUp at the prompt, which will display a popup window of the most recent directory changes; you can select a directory and switch to it by pressing Enter.

You can change to the parent directory with "CD ..". You can also go up one additional directory level with each additional ".". For example, "CD ...." will go up three directory levels.

See also %CDPATH, CD and PUSHD.

EXAMPLES

Change to the subdirectory C:\WP:

```
cdd c:\wp
```

Set the CDPATH environment variable and then CDD to a directory called DOCS (CDD will first attempt to change to DOCS in the current directory, then C:\DOCS, then C:\OS\DOCS, etc.):

```
set cdpath=c:\;c:\os;c:\util;c:\wp
cdd docs
```

SYNTAX (Internal 4NT)

CLS [[BRIGHT] fg ON [BRIGHT] bg

PURPOSE

Clear the video display, optionally to the specified colors.

COMMENTS

CLS clears the display and moves the cursor to the upper left corner. **fg** is the foreground color, **bg** the background color. Only the first three characters of the color name and attribute ("bright") are required. You can set the default colors in the 4NT.INI file.

The available colors are:

Black	Blue	Green	Red
Magenta	Cyan	Yellow	White

CLS is normally used in batch files to clear the screen before displaying text.

See also COLOR and 4NT.INI Color Directives.

EXAMPLES

Clear the display:

```
cls
```

Clear the display to a blue background, and set white characters as the new default:

```
cls white on blue.
```

SYNTAX (Internal 4NT)

COLOR [[BRIGHT] fg ON [BRIGHT] bg]

PURPOSE

Set the screen display colors.

COMMENTS

fg is the foreground color, bg the background color. Only the first three characters of the color name and attribute ("bright") are required.

The available colors are:

Black	Blue	Green	Red
Magenta	Cyan	Yellow	White

See also CLS.

EXAMPLES

Set the default screen colors to bright white text on a blue background:

```
color bright white on blue
```

SYNTAX (Internal 4NT)

COPY [d:][path]filename[+]...[/A /B] [[d:][path]filename] [/A /B /C /M /N /P /Q /R /S /U /V]

PURPOSE

Copy or append one or more files.

COMMENTS

4DOS for Windows NT allows you to copy several unrelated files to a target directory with a single COPY command. If there are two or more arguments on the command line, COPY assumes the last argument is the target. If there is only one argument, the target is assumed to be the current directory.

The plus (+) tells COPY to append two or more files to a single target file. If you don't specify a target, COPY will append each subsequent file to the first file.

If you specify more than one source file, and the target is NOT a directory, COPY will automatically append the files to the target.

The /A(SCII) or /B(inary) options apply to the preceding filename and to all subsequent filenames on the command line until the filename preceding another /A or /B, if any. All other options apply to all filenames on the command line.

The COPY options are:

- **/A** If used with a source filename, COPY will copy the file up to, but not including, the first ^Z character in the file. If you use /A with a target filename, COPY will add a ^Z to the end of the file. COPY defaults to /A when appending files.
- **/B** If used with a source filename, COPY will copy the entire file. Using /B with a target filename prevents COPY from appending a ^Z to the target file. COPY defaults to /B for normal file copies.
- **/C** Copy only those files where the target exists and is older than the source (see /U).
- **/H** Copy hidden and system files too.
- **/M** Copy only those files with the archive bit set (see also ATTRIB). The archive bit will NOT be cleared after copying.
- **/N** Do everything except actually perform the copy (for testing what the result of the COPY would be).
- **/P** Confirm each file copy (Y or N). A third option (R) turns off the confirmation and processes the remainder of the files normally.
- **/Q** Don't display filenames as they are copied.
- **/R** Prompt before overwriting an existing file.
- **/S** Copy subdirectories - the target must be a directory (COPY will create it if it doesn't exist). COPY will copy each subdirectory to a matching subdirectory of the target.
- **/U** Copy only those source files that are newer than a matching target file, or where a matching target file doesn't exist (see /C).
- **/V** Verify each disk write. This option doesn't have any effect in Windows NT; it is included for compatibility with DOS batch files.

See also MOVE.

EXAMPLES

Copy the files MEMO1 and PROJECT8.WKS to the root directory on drive A:

```
copy memo1 project8.wks a:\
```

Append the files MEMO1, MEMO2, and MEMO3 and store the result in BIGMEMO:

```
copy memo1+memo2+memo3 bigmemo
```

Copy only those files in the root directory on drive A that are newer than the matching files in the current directory, or that don't exist in the current directory:

```
copy /u a:\*.* c:\*.*
```

Copy files from the root directory on drive A to the current directory, but prompt before overwriting existing files:

```
copy /r a:\*.*
```

Copy a downloadable font file to the printer in binary mode:

```
copy myfont.dat lpt1: /b
```

SYNTAX (Internal 4NT)

DATE [mm-dd-yy]

PURPOSE

Display and (optionally) change the system date.

COMMENTS

If you don't enter any parameters, DATE will display the current system date and time, and prompt for the new date. Press ENTER if you don't wish to change the date, otherwise enter the new date.

The format for the date entry depends on the country code. The default format is U.S. (mm-dd-yy). The European format is dd-mm-yy; the Japanese is yy-mm-dd.

The parameters for the DATE command are:

mm Month (1 - 12)
dd Day (1 - 31)
yy Year (80 - 199 or 1980 - 2099)

You can use hyphens, slashes, or periods to separate the month, day, and year entries.

See also TIME..

EXAMPLES

Enter the date from the command line:

```
date 12/25/92
```

To be prompted for the date:

```
date
```

SYNTAX (Internal 4NT)

DEL [/N /P /Q /S /X /Y /Z] [d:][path]filename...
ERASE [/N /P /Q /S /X /Y /Z] [d:][path]filename...

PURPOSE

Erase the specified file(s) from the disk.

COMMENTS

The DEL options are:

- **/N** Do everything except actually delete the file(s) (for testing what the result of a DEL would be).
- **/P** Confirm file deletion (Y or N) for each file. A third option (R) turns off the confirmation and processes the remainder of the files normally.
- **/Q** Don't display filenames as they are deleted.
- **/S** Delete matching files in the subdirectories.
- **/X** Remove empty subdirectories after deleting (use with /S).
- **/Y** The reverse of /P - it assumes a Y response to everything, including deleting an entire subdirectory. Use with caution!
- **/Z** Delete all files, including read-only, hidden and system files.

If you enter a subdirectory name, or a filename composed only of wildcards, DEL asks for confirmation (Y or N), unless you specified the /Y(es) option. If you respond with a Y, DEL will delete all the files in that subdirectory (except for hidden, system, and read-only files).

DEL will display the number of files deleted, and (if any) the amount of disk space freed.

EXAMPLES

Erase all the files in the current directory with a .BAK or .PRN extension:

```
del *.bak *.prn
```

Delete the entire subdirectory tree starting with C:\UTIL, including hidden and read-only files:

```
del /sxyz c:\util
```

SYNTAX (Internal 4NT)

DELAY [seconds]

PURPOSE

Pause for a specified period of time.

COMMENTS

DELAY is useful in batch file loops while waiting for a condition to occur. The default value is one second.

If you need a shorter delay, use BEEP with a frequency < 37 Hz.

You can cancel a DELAY loop by pressing ^C.

See also BEEP.

EXAMPLES

Wait for 10 seconds:

```
delay 10
```

SYNTAX (Internal 4NT)

DESCRIBE [d:][path]filename... ["description"]

PURPOSE

Create, modify, or delete file and subdirectory descriptions.

COMMENTS

DESCRIBE adds descriptions to filenames and subdirectories. The descriptions will be displayed when using DIR with the default single column option, or when using SELECT. If you're using 4DOS for Windows NT on an NTFS or HPFS volume, descriptions will only be displayed if you use the DIR or SELECT /Z (FAT format) option. The default maximum description length is 40 characters; if you have a screen capable of displaying more than 80 columns, you can increase the maximum description length with the 4NT.INI DescriptionMax option.

File descriptions allow you to identify your files in much more meaningful ways than an eight character filename.

You can enter a description on the command line by entering the filename followed by the description enclosed in quotes. Using wildcards and/or multiple filenames with a description on the command line will give all matching files the same description.

Descriptions are stored in each directory in a hidden file called DESCRIPT.ION. Use the ATTRIB command to "unhide" this file if you need to copy or delete it.

The description file is modified appropriately whenever you perform an internal command (such as COPY, DEL, MOVE, or RENAME), but not if you use an external command (such as REPLACE or XCOPY).

See also DIR and SELECT.

EXAMPLES

Create a description for the file MEMO.TXT interactively:

```
describe memo.txt
```

Create the same description in a single command:

```
describe memo.txt "Memo to Bob about party"
```

SYNTAX (Internal 4NT)

DETACH [d:][filename[.ext]]

PURPOSE

Start an Windows NT program in detached mode.

COMMENTS

A program started with DETACH cannot use the keyboard, mouse, or video display. You can redirect standard I/O to other devices if necessary.

4DOS for Windows NT detaches a copy of 4NT.EXE to execute the command, so it can be an internal command, alias, batch file, or external command.

EXAMPLES

Detach a CHKDSK command and redirect its output to the file DIR.DAT:

```
detach chkdsk >& dir.dat
```

SYNTAX (Internal 4NT)

DIR [/1 /2 /4 /A[:-rhsda] /B /C /D /F /J /K /L /M /N /O:-deginrsu /P /S /T[:acw] /U /V /W /X /Z] [[d:] [path]filename...]

PURPOSE

Display information about files and subdirectories.

COMMENTS

DIR displays information about the files and subdirectories in the specified directory. Depending upon the options specified, DIR can show the filename, file attributes, size, date and time of the most recent change to the file, and the file description. If the file description would exceed the screen width, DIR continues it on the next line.

The DIR options are:

- **/1** Single column display - display the filename, size, date, time, and description. This is the default.
- **/2** Two column display - display the filename, size, date, and time.
- **/4** Four column display - display the filename and size, in K(ilobytes) or M(egabytes).
- **/A** Display only those files that have the specified attribute set. A /A with no attributes will display all files, including hidden and system files. Preceding the attribute character with a '-' will display those files that do not have that attribute set. Attributes can also be combined. The attributes are:
  - **R** Read only
  - **H** Hidden
  - **S** System
  - **D** Directory
  - **A** Archive
- **/B** Suppress header line and summaries, and display file or subdirectory names only, in a single column (useful when redirecting output to a file or another program).
- **/C** Display FAT filenames in upper case (like CMD.EXE). See also SETDOS /U.
- **/D** Disable directory colorization.
- **/F** Display fully expanded filenames (including drive & path) in one column.
- **/J** Justify filenames (same format as CMD.EXE).
- **/K** Suppress the header display.
- **/L** Display FAT filenames in lower case. This option will not convert extended ASCII characters to lower case.
- **/M** Suppress the footer display.
- **/N** Display a FAT volume in NTFS format.
- **/O** Sort order; any combination of the following options:
  - **-** Reverse the sort order for the next option
  - **a** Sort by ASCII value rather than numeric value
  - **d** Sort by date and time (oldest first)
  - **e** Sort by extension
  - **g** Group subdirectories together
  - **i** Sort by the file description
  - **n** Sort by the filename (this is the default)
  - **r** Reverse the sort order for all options
  - **s** Sort by size
  - **u** Unsorted
- **/P** Pause after each screen page and wait for a key to be pressed (useful for displaying long directory listings).

/S Display contents of the current directory and all of its subdirectories. DIR will only display headers and summaries for those directories with matching filenames.

/T Display the filenames and attributes only (see ATTRIB), in the format RHSA, where:

- **R** Read only
- **H** Hidden
- **S** System
- **A** Archive

When displaying attributes, DIR cannot display file descriptions.

/T:acw Specify which date and time field (on NTFS and HPFS drives) should be displayed and used for sorting, where:

- **A** last access time
- **C** creation time
- **W** last write time

/U Display summary only (number of files & bytes used; both actual file size and the disk space used).

/V Display filenames sorted vertically rather than horizontally (with the /2, /4, or /W option).

/W Wide display - display the filenames only, horizontally across the screen (5 columns on an 80-character display).

/X Display both the short (8.3) and long filenames on an NTFS drive.

/Z Display an HPFS or NTFS volume in FAT format. Filenames longer than 12 characters will be truncated.

DIR allows wildcard characters in the filename. If you don't specify a filename, DIR defaults to *.* (display all files and subdirectories in the current directory).

If you append filenames with a ; (an "include list"), DIR will display the matching filenames in a single directory listing. Only the first file in an include list can have a path.

You can display the subdirectories and files in color by setting the COLORDIR variable. The format for COLORDIR is:

```
ext [...]:[BRIGHT] fg [ON [BRIGHT] bg]; ...
```

where "ext" is the file extension, or one of the following file attributes:

```
DIRS - directory
RDONLY - read-only file
HIDDEN - hidden file
SYSTEM - system file
ARCHIVE - file modified since last backup
```

For example, to display the .COM and .EXE files in red, the .C and .ASM files in bright cyan, and the read-only files in green:

```
set colordir=com exe:red; c asm:bright cyan; rdonly:green
```

If you don't select a background color, DIR will use the current screen background color.

If a country code was defined, DIR will display the date in the format for that country. The default date format is U.S. (mm-dd-yy).

Options on the command line apply only to the filenames that follow the option, except that options at the end of the line apply to the preceding filename only. This allows you to specify several options for a group of files, and retains compatibility with CMD.EXE when a single filename is specified.

See also DESCRIBE and SELECT.

EXAMPLES

Display the .WKS files, and then the .WK1 files in the current directory:

```
dir *.wks *.wk1
```

Display the .WKS and .WK1 files together (an "include list"):

```
dir *.wks;*.wk1
```

Display the files on drive C, including hidden and system files, pausing after each page:

```
dir /a/sp c:\*.*
```

SYNTAX (Internal 4NT)

DIRS

PURPOSE

Display the current directory stack.

COMMENTS

DIRS displays the directory stack used by PUSHD and POPD, oldest entries first. The stack holds 255 characters (about 10 to 20 entries).

See also PUSHD and POPD.

EXAMPLES

Change directories and then display the directory stack:

```
pushd c:\database
pushd d:\wordp\memos
dirs
```

SYNTAX (Internal 4NT)

DPATH [[d:]path][[;[d:]path]...]]
DPATH ;

PURPOSE

Tells applications where to search for their files.

COMMENTS

If you enter DPATH with no parameters, 4DOS for Windows NT displays the current search path. If you enter DPATH and a semicolon, 4NT clears the search path.

You can edit an existing DPATH with the ESET command.

EXAMPLES

The following DPATH command directs applications to search for their data files in the following order: the current directory, the root directory on drive C, the NT subdirectory on drive C, and the UTIL subdirectory on drive C:

```
dpath c:\;c:\nt;c:\util
```

Display the current DPATH:

```
dpath
```

SYNTAX (Internal 4NT)

DRAWBOX ulrow ulcol lrrow lrcol style [BRIGHT] fg ON [BRIGHT] bg [FILL bgfill] [SHADOW]

PURPOSE

Draw a box on the screen.

COMMENTS

DRAWBOX is useful for creating attractive screen displays in batch files. DRAWBOX detects other lines and boxes on the display, and creates the appropriate connector characters when possible (not all types of lines can be connected with the available characters).

The row and column numbering is zero-based, so on a standard 25 line by 80 column display, valid rows are 0 - 24 and valid columns are 0 - 79.

The DRAWBOX parameters are:

ulrow Row for upper left corner
ulcol Column for upper left corner
lrrow Row for lower right corner
lrcol Column for lower right corner
style Box drawing style:
- **0** no line drawing characters (box is drawn with blanks)
- **1** single line
- **2** double line
- **3** single line on top and bottom, double on sides
- **4** double line on top and bottom, single on sides

fg Foreground character color
bg Background character color
bgfill Background fill color (for the inside of the box)
SHADOW Include a transparent drop shadow

Only the first three characters of the color name and attribute ("bright") are required.

The available colors are:

Black	Blue	Green	Red
Magenta	Cyan	Yellow	White

See also DRAWHLINE and DRAWVLINE.

EXAMPLES

Draw a single line box around the entire screen with bright white lines on a blue background:

```
drawbox 0 0 24 79 1 bright white on blue fill blue
```

SYNTAX (Internal 4NT)

DRAWHLINE row column length style [BRIGHT] fg ON [BRIGHT] bg

PURPOSE

Draw a horizontal line on the screen.

COMMENTS

DRAWHLINE is useful for creating attractive screen displays in batch files. DRAWHLINE detects other lines and boxes on the display, and creates the appropriate connector characters when possible (not all types of lines can be connected with the available characters).

The row and column numbering is zero-based, so on a standard 25 line by 80 column display, valid rows are 0 - 24 and valid columns are 0 - 79.

The DRAWHLINE parameters are:

row	Start row
column	Start column
length	Length of line
style	Line drawing style: **1** Single line **2** Double line
fg	Foreground character color
bg	Background character color

Only the first three characters of the color name and attribute ("bright") are required.

The available colors are:

Black	Blue	Green	Red
Magenta	Cyan	Yellow	White

See also DRAWBOX, DRAWVLINE and SCRPUT..

EXAMPLES

Draw a double line along the top row of the display with green characters on a blue background:

```
drawhline 0 0 79 2 green on blue
```

SYNTAX (Internal 4NT)

DRAWVLINE row column length style [BRIGHT] fg ON [BRIGHT] bg

PURPOSE

Draw a vertical line on the screen.

COMMENTS

DRAWVLINE is useful for creating attractive screen displays in batch files. DRAWVLINE detects other lines and boxes on the display, and creates the appropriate connector characters when possible (not all types of lines can be connected with the available characters).

The row and column numbering is zero-based, so on a standard 25 line by 80 column display, valid rows are 0 - 24 and valid columns are 0 - 79.

The DRAWVLINE parameters are:

row	Start row
column	Start column
length	Length of line
style	Line drawing style: **1** Single line **2** Double line
fg	Foreground character color
bg	Background character color

Only the first three characters of the color name and attribute ("bright") are required.

The available colors are:

Black	Blue	Green	Red
Magenta	Cyan	Yellow	White

See also DRAWBOX, DRAWHLINE, and VSCRPUT.

EXAMPLES

Draw a double line along the left margin of the display with bright red characters on a black background:

```
drawvline 0 0 24 2 bright red on black
```

SYNTAX (Internal 4NT)

ECHO [on | off | message]

PURPOSE

Display the echo status, enable or disable batch file or command line echoing, or display a message.

COMMENTS

ECHO defaults to ON in batch files. To prevent a line from being echoed, preface it with the @ symbol. You can default to ECHO OFF by setting the SETDOS /V option to 0. (Setting SETDOS /V to 2 will echo all lines, regardless of the echo state or @ symbols.) The current ECHO state is inherited by called batch files.

ECHO defaults to OFF during keyboard input. If you set ECHO ON from the command line, the fully parsed and expanded commands (including aliases and variables) will be displayed before they are executed. The keyboard ECHO state is independent of the batch file ECHO state; changing ECHO in a batch file has no effect on the display at the command prompt, and vice versa.

If no arguments are entered, ECHO displays the current echo state.

ECHO commands in a batch file will send messages to the screen while the batch file executes, even if ECHO is set OFF. You cannot use the command separator character or the redirection symbols (|><) in an ECHO message, unless you enclose them in quotes or precede them with the escape character. If you want to echo a blank line, enter:

```
echo.
```

See also ECHOS, SETDOS, SCREEN, SCRPUT, TEXT and VSCRPUT.

EXAMPLES

Enable command line echoing:

```
echo on
```

Display a message in a batch file:

```
echo Processing your print files...
```

Turn off batch file echoing, without displaying the ECHO command itself:

```
@echo off
```

SYNTAX (Internal 4NT)

ECHOS message

PURPOSE

Display a message, without printing a trailing CR/LF.

COMMENTS

ECHOS is useful for outputting text when you don't want ECHO to add a carriage return / linefeed pair (for example, when redirecting control sequences to a printer). You cannot use the command separator character or the redirection symbols (|><) in an ECHOS message, unless you enclose them in quotes or precede them with the escape character.

See also ECHO.

EXAMPLES

Send a control sequence (an Esc+H) to the printer.

```
echos ^eH > lpt1:
```

SYNTAX (Internal 4NT)

ENDLOCAL

PURPOSE

Restore the saved disk drive, directory, aliases, and environment.

COMMENTS

ENDLOCAL restores the disk drive, directory, aliases, and environment variables saved by the previous SETLOCAL command. SETLOCAL and ENDLOCAL can only be used in batch files, not in aliases.

See also SETLOCAL.

EXAMPLES

This batch file fragment saves the aliases, environment, drive, and current working directory, changes the drive and directory, modifies some environment variables, runs the program TEST1, and then restores the original values:

```
setlocal
cdd d:\test
set path=c:\;c:\dos;c:\util
set lib=d:\lib
test1
endlocal
```

SYNTAX (Internal 4NT)

ESET [/A] varname...

PURPOSE

Edit environment variables and/or aliases.

COMMENTS

ESET allows you to edit your environment variables and aliases using the line editing commands. The cursor will be positioned at the first character of the variable or alias.

The only ESET option is:

/A Assume the argument is an alias. This allows you to edit an alias with the same name as an environment variable.

ESET will search for environment variables first, and then aliases.

Environment variable and alias names are limited to 80 characters, and their arguments to 255 characters.

See also Editing the Command Line, ALIAS, UNALIAS, SET, and UNSET.

EXAMPLES

Edit the executable file search path:

```
eset path
```

Create and then edit an alias:

```
alias d dir /djp
eset d
```

SYNTAX (Internal 4NT)

EXCEPT ([d:][path]filename...) command

PURPOSE

Perform a command, except on the file(s) specified.

COMMENTS

EXCEPT provides a means of executing a command on a group of files and/or subdirectories, except those enclosed within the parentheses.

The command can be an internal command or alias, an external command, or a batch file.

You may use wildcard characters in a filename.

When using filename completion (TAB or F9) to get the filenames inside the parentheses, type a space after the open parenthesis before entering a partial filename or pressing TAB. Otherwise the command line editor will treat the open parenthesis as the first character of the filename to be completed.

EXCEPT prevents operations on the specified file(s) by setting the hidden attribute, performing the command, and then clearing the hidden attribute. If EXCEPT is aborted in an unusual way, you may need to use the ATTRIB command to "unhide" (-H) the file(s).

EXCEPT will not work with programs or commands that ignore the hidden attribute.

See also ATTRIB.

EXAMPLES

Erase all but the files beginning with MEMO and those ending in .WKS:

```
except (memo*.* *.wks) del *.*
```

SYNTAX (Internal 4NT)

EXIT [value]

PURPOSE

Exit the current command processor.

COMMENTS

Some application programs will start a secondary copy of the command processor to allow you execute commands. To return to the application again, type EXIT.

If you EXIT from a primary command processor, you will be returned to the Windows NT desktop.

If you specify a value, EXIT will return that value to the parent process.

EXAMPLES

Return to the parent process:

```
exit
```

SYNTAX (Internal 4NT)

FOR [/A[:-RHSDA]] %%var IN (set) [DO] command

PURPOSE

Repeat a command for several variables.

COMMENTS

FOR sets var sequentially to each member of (set), and then evaluates and executes command for every argument in (set). If an argument in (set) contains wildcard characters, var will be set sequentially to each matching filename on the disk. If a filename in (set) begins with an @, var will be set sequentially to each line in the file. You can nest multiple FOR statements.

The only option for FOR is:

/A Retrieve only those files with the specified attribute. A /A with no attributes will retrieve all files, including hidden and system files, and subdirectories. Preceding the attribute character with a '-' will retrieve those files that DON'T have that attribute set. Attributes can also be combined. The attributes are:

- **R** Read only
- **H** Hidden
- **S** System
- **D** Directory
- **A** Archive

The command can be an internal command or alias, an external command, or a batch file.

In CMD.EXE you must use a single % for the variable name from the command line, and %% when in a batch file. 4NT will accept either % or %% in either case.

CMD.EXE requires the variable name to be a single character; 4NT supports variable names up to 80 characters.

CMD.EXE requires the word DO on the command line; it is optional in 4NT.

EXAMPLES

The following example compiles the C programs in the current directory that have a "2" or a "3" somewhere in their name:

```
for %a in (*2*.c *3*.c) do cl %a ;
```

The following example uses variable functions to delete the .BAK files for which a corresponding .TXT file exists in the current directory:

```
for %a in (*.txt) if exist %@name[%a].bak del %@name[%a].bak
```

SYNTAX (Internal 4NT)

FREE [d:] ...

PURPOSE

Display the total disk space, total bytes used, and total bytes free on the specified (or default) drive(s).

COMMENTS

FREE provides the same disk information as the external command CHKDSK, but without the wait.

EXAMPLES

Display the status of drives A, B, and C:

```
free a: b: c:
```

SYNTAX (Internal 4NT)

GLOBAL [/H /I /P /Q] command

PURPOSE

Execute a command in the current directory and its subdirectories.

COMMENTS

GLOBAL performs the specified command first in the current directory, and then in every subdirectory under the current directory.

The command can be an internal command or alias, an external command, or a batch file.

The GLOBAL options are:

- **/H** Process hidden subdirectories.
- **/I** Ignore exit codes. If this option is not specified, GLOBAL will terminate if the command returns a non-zero exit code.
- **/P** Prompt for a Y/N whether to execute the command in each directory. A third option (R) turns off the confirmation and processes the remainder of the files normally.
- **/Q** Do not display the directory names as each directory is accessed.

EXAMPLES

Copy the files in every directory on drive A to the directory C:\TEMP:

```
cd a:\ & global copy a:*.* c:\temp
```

SYNTAX (Internal 4NT)

GOSUB label

PURPOSE

Call a subroutine in a batch file.

COMMENTS

GOSUB calls the specified label as a subroutine. The subroutine must end with a RETURN statement. After the RETURN, the batch file continues with the command following the GOSUB command.

GOSUB searches for the label starting at the first line of the batch file. If the label doesn't exist, the batch file is terminated with the error message "Label not found."

The label must begin with a colon (:), and appear by itself on the line. GOSUB ignores case differences when matching labels.

See also GOTO and RETURN.

EXAMPLES

The following batch file fragment calls a subroutine that displays the directory and returns:

```
echo Calling a subroutine
gosub subr1
echo Returned from the subroutine
quit
:subr1
dir /hw
return
```

SYNTAX (Internal 4NT)

GOTO [/I] label

PURPOSE

Continue batch file processing at the line following the label.

COMMENTS

GOTO changes the current position in the batch file to the line immediately following the label.

The search for the label starts at the first line of the batch file. If the label doesn't exist, the batch file is terminated with the error message "Label not found."

The label must begin with a colon (:), and appear by itself on the line. GOTO ignores case differences when matching labels.

A GOTO will cancel all IFF nesting, unless you specify the /I option.

See also GOSUB.

EXAMPLES

The following batch file fragment checks for the existence of the file CONFIG.SYS. If CONFIG.SYS exists, GOTO jumps to C_EXISTS and copies all the files from the current directory to the root directory on A. If CONFIG.SYS doesn't exist, the batch file prints an error message and exits.

```
if exist config.sys goto C_EXISTS
echo CONFIG.SYS doesn't exist - exiting
quit
:c_exists
copy *.* a:\
```

SYNTAX (Internal 4NT)

HELP [command]

PURPOSE

Display the 4DOS for Windows NT help file.

COMMENTS

HELP uses the Windows NT Help Manager to display the nelp file (4NT.HLP).

EXAMPLES

Display the HELP table of contents:

```
help
```

Display help for the DIR command:

```
help dir
```

SYNTAX (Internal 4NT)

HISTORY [/A command] [/F /P] [/R [d:][path]filename]

PURPOSE

Display, read, or clear the history list.

COMMENTS

If no parameters are entered, HISTORY displays the current history list.

The HISTORY options are:

/A Add the specified command to the history list.
/F Clear the command history list.
/P Prompt for a key after displaying each page.
/R Read the command history from the specified file. You can save the history list by redirecting the output of HISTORY to a file.

The number of commands saved in the history list depends on the length of each command line. The history list size can be specified at startup from 256 to 8192 characters. The default size is 1024 characters.

You can disable saving commands to the history list, or specify a minimum command line length to save, with the HistMin directive in 4NT.INI.

EXAMPLES

Display the history list:

```
history
```

Clear the history entries:

```
history /f
```

Save the history list to the file HISTFILE, and then read it in again:

```
history > histfile
history /r histfile
```

SYNTAX ***(Internal 4NT)***

IF [NOT] condition [.AND. | .OR. | .XOR. [NOT] ...] command

PURPOSE

Allow conditional execution of commands.

COMMENTS

IF first tests the given condition. If the condition is true, IF executes the specified command, otherwise the command will be ignored. If you use the NOT option, the command is executed only when the condition is false. IF statements can be nested.

The command can be an internal command or alias, an external command, or a batch file.

For the string tests, case differences are ignored. When comparing strings, you should enclose them in double quotes (see examples). The use of double quotes reduces problems when the strings being compared contain characters that may have another meaning to the parser. If the strings begin with a digit, IF will do a numeric comparison. Otherwise, IF does an ASCII comparison.

The .AND., .OR., and .XOR. tests allow you to combine tests in an IF statement. The expressions are scanned from left to right.

The condition can be any of the following:

string1 == string2	
or	
string1 EQ string2	If string1 is equal to string2, the condition is true.
string1 != string2	
or	
string1 NE string2	If string1 is not equal to string2, the condition is true.
string1 LT string2	If string1 is < string2, the condition is true.
string1 LE string2	If string1 is <= string2, the condition is true.
string1 GE string2	If string1 is >= string2, the condition is true.
string1 GT string2	If string1 is > string2, the condition is true.

In the tests below, the term "condition" refers to one of the relational operators described for the string tests above (==, EQ, NE, LT, etc.).

ERRORLEVEL [condition] n Test the exit code of the preceding external program. If no relational operator (EQ, GT, etc.) is specified, the default is GE. NOTE: Not all programs return an explicit exit code. In those cases, the behavior of ERRORLEVEL is undefined.

EXIST [d:][path]filename If the file exists, the condition is true. You can use wildcard characters in the filename, in which case the condition is true if any file matching the wildcards exists.

ISALIAS aliasname If the specified name is an alias, the condition is true.

ISDIR [d:]path If the subdirectory exists, the condition is true.

ISINTERNAL command If the specified command is an internal command, the condition is true.

ISLABEL labelname If the specified label exists in the current batch file, the condition is true.

See also IFF

EXAMPLES

Test for the presence of A:\JAN.DOC and copy it to the root directory on drive C if it exists:

```
if exist a:\jan.doc copy a:\jan.doc c:\
```

This batch file fragment tests for a string value:

```
if "%cmd" == "wp" goto wordproc
if "%cmd" NE "graphics" goto badentry
```

Test for more than 500K of free disk space on drive A:

```
if %@diskfree[a:] gt 500 echo More than 500K free
```

SYNTAX (Internal 4NT)

IFF [NOT] condition [.AND. | .OR. | .XOR. [NOT] ...] THEN & ... & ELSE[[IFF] ... THEN] & ... & ENDIFF

PURPOSE

Allow IF/THEN/ELSE conditional execution of commands.

COMMENTS

IFF first tests the given condition. If the condition is true, IFF executes the specified command(s) until the next ELSE, ELSEIFF, or ENDIFF; otherwise the command(s) will be ignored and parsing will continue at the next ELSE, ELSEIFF, or ENDIFF. If you use the NOT option, commands are executed only when the condition is false. IFF statements can be nested up to 15 levels deep.

The .AND., .OR., and .XOR. tests allow you to combine tests in an IFF statement. The expressions are scanned from left to right; each new .AND., .OR., or .XOR. tests the combination of all preceding tests.

The command can be an internal command or alias, an external command, or a batch file.

If you do a GOTO inside an IFF, GOTO assumes you are jumping outside the IFF statement. You cannot GOTO another part of the same IFF, or inside another IFF statement.

See the IF command for a list of the tests available.

EXAMPLES

The following batch file fragment tests the monitor type (monochrome or color), and sets the appropriate colors and prompt:

```
iff %_monitor == color then
   color bright white on blue & cls
   prompt=$e[s$e[1;1f$e[41;1;37m$e[K   Path: $p$e[u$e[44;37m$n$g
else
   prompt=$e[s$e[1;1f$e[0;7m$e[K   Path: $p$e[u$e[0m$n$g
endiff
```

The following alias checks to see if the argument is a subdirectory. If so, the alias deletes the subdirectory's files and removes it (enter this on one line):

```
alias zap `iff isdir %1 then & del /s/x/z %1 & else &
    echo Not a directory! & endiff`
```

SYNTAX (Internal 4NT)

INKEY [/K"..." /Wn] [text] %%varname

PURPOSE

Get a single keystroke environment variable from standard input.

COMMENTS

You can optionally display prompt text before the variable name.

The INKEY options are:

/K The allowable keystrokes, enclosed in double quotes. If you want to read a control or function key, you must enclose it in square brackets (for example, /K"ab[F1][F10]"). The valid key names are the same as those listed in the section on Key Mapping Directives in KEYCODES.

/W Timeout period, where "n" is the number of seconds to wait for a response. If no keystroke is entered by the end of the timeout period, INKEY returns with the variable unchanged. You can specify /W0 to check if a keystroke is waiting, and return immediately.

ASCII values from 1 to 255 are stored as a character. Extended characters (for example, function keys & cursor keys) are stored as a string in decimal format, with a leading @ (for example, the F1 key is stored as @59). The ENTER character is a special case; it is stored as its scan code (@28). See KEYCODES for the common extended keystrokes.

INKEY and INPUT are normally used as batch file commands, allowing you great flexibility in entering or changing batch variables.

If you press ^C or ^BREAK while INKEY is waiting for a key, execution of an alias will be terminated, and execution of a batch file will be suspended while you are prompted whether to cancel the batch job.

See also INPUT.

EXAMPLES

Prompt for a number and store it in the variable NUM:

```
inkey /K"123456789" Enter a number from 1 to 9:  %%num
```

The following batch file fragment waits up to 10 seconds for a character, then tests to see if a "Y" was entered:

```
set net=N
inkey /w10 Do you want to load the network (Y/N)? %%net
if "%net"=="Y" goto load_net
```

SYNTAX (Internal 4NT)

INPUT [/E /Ln /Wn] [text] %%varname

PURPOSE

Enter an environment variable from the standard input.

COMMENTS

You can optionally display prompt text before the variable name.

The INPUT options are:

/E Echo and allow editing an existing value (if any) for "varname".

/L Specify the maximum length, where "n" is the limit of characters to accept.

/W Specify a timeout period, where "n" is the number of seconds to wait for a response. If no keystroke is entered by the end of the timeout period, INPUT returns with the variable unchanged. If you enter a key before the timeout period, INPUT will wait indefinitely for the remainder of the line. You can specify /W0 to check if a key is already in the buffer, and return immediately.

All characters entered up to, but not including, the carriage return are stored in the variable.

INPUT and INKEY are normally used as batch file commands, allowing you great flexibility in entering or changing batch variables.

See also INKEY.

EXAMPLES

The following batch file fragment prompts for a string and stores it in the variable FNAME:

```
input Enter the file name:  %%fname
```

SYNTAX (Internal 4NT)

KEYS [ON | OFF | LIST]

PURPOSE

Display the history list, or enable/disable the command line editing keys.

COMMENTS

4DOS for Windows NT includes this command for compatibility with CMD.EXE; for 4NT you should use the HISTORY command instead.

SYNTAX (Internal 4NT)

KEYSTACK ["text"] [keyname] [!]

PURPOSE

Feed keystrokes to an internal or external command.

COMMENTS

Keystack is not yet enabled in 4DOS for Windows NT.

EXAMPLES

SYNTAX (Internal 4NT)

LIST [/H /S /W] [d:][path]filename...

PURPOSE

Display a file with forward and backward paging and scrolling.

COMMENTS

LIST provides a much faster and more flexible way to view a file than TYPE, without the overhead of using a text editor.

The LIST options are:

/H Strip the high bit from each character before displaying. This is useful when displaying files created by some word processors that turn on the high bit for formatting purposes.
/S Read from the standard input rather than a file. This allows you to redirect command output and view it with LIST.
/W Wrap the text at the right margin. This option is useful when displaying non-text files.

LIST uses the cursor pad to scroll through the file. The LIST commands are (the caret ^ means press the Ctrl key together with the specified key):

HOME Display the first page of the file
END Display the last page of the file
PgUp Scroll back one page
PgDn Scroll forwards one page
Esc Exit the current file
^C Quit LIST
Up Scroll up one line
Down Scroll down one line
Left Scroll left 8 columns
Right Scroll right 8 columns
^Left Scroll left 40 columns
^Right Scroll right 40 columns
F1 Call the on-line help
F Prompt and search for a string (case is ignored)
H Toggle the "strip high bit" (/H) option
N Find next matching string (case is ignored). LIST saves the search string, so you can LIST multiple files and search for the same string by pressing N in each file.
P Print the file on LPT1
W Toggle the "line wrap" (/W) option

LIST is normally only useful for displaying ASCII text files; executable files (.COM and .EXE) and many data files will be unreadable due to the presence of non-alphanumeric characters.

See also TYPE.

EXAMPLES

Display the file MEMO.DOC:

```
list memo.doc
```

Display the output from a DIR command:

```
dir | list /s
```

SYNTAX (Internal 4NT)

LOADBTM [on | off]

PURPOSE

Switch a batch file to or from .BTM mode.

COMMENTS

LOADBTM switches a batch file (.CMD, .BAT, or.BTM) to and from .BTM mode. If no argument is given, it displays the current LOADBTM status.

.BTM mode runs from two to five times faster than normal batch mode, but should not be used for self-modifying batch files.

LOADBTM can only be used within a .CMD, .BAT, or .BTM file.

See the 4DOS for Windows NT / 4DOS manual for more information on .BTM files.

EXAMPLES

The following .CMD batch file fragment tests if 4NT is the current command processor; if so, it switches to BTM mode:

```
Rem The initial default state is LOADBTM OFF
if "%@eval[2+2]" == "4" loadbtm on
```

SYNTAX (Internal 4NT)

LOG [/W [d:]pathname] [ON | OFF | "text"]

PURPOSE

Save a log of commands to a disk file.

COMMENTS

The default 4DOS for Windows NT LOG filename is 4NTLOG in the root directory of the boot drive. The LOG status and log filename will be passed to secondary shells.

Entering LOG with no parameters displays the log status (ON or OFF). Entering LOG with text writes the text to the log file, even if LOG is set OFF. This allows you to enter headers in the log file.

You can specify a different filename with the /W(rite) option. /W automatically enables command logging.

The commands are stored in the log file as they are executed, after performing any alias or variable expansion.

The LOG file format looks like:

```
[mm/dd/yy  hh:mm:ss] command.
```

EXAMPLES

Display the LOG status:

```
log
```

Enable command logging:

```
log on
```

Enable command logging to the file C:\LOG\LOGFILE and insert a header:

```
log /w c:\log\logfile
log "Started work on the database system"
```

SYNTAX (Internal 4NT)

MD [/S] [d:]pathname...
MKDIR [/S] [d:]pathname...

PURPOSE

Create subdirectories.

COMMENTS

The /S option instructs MD to create any parts of the directory path that don't already exist.

To start at the root directory, start the pathname with a backslash (\). To start at the parent directory, start the pathname with two periods (..). All other pathnames start at the current directory.

See also RD.

EXAMPLES

Create a subdirectory called MYDIR in the root directory:

```
md \mydir
```

SYNTAX (Internal 4NT)

MEMORY

PURPOSE

Display the system RAM status.

COMMENTS

MEMORY displays the size of physical memory, the total free memory (including swap space), the largest free block of RAM, the total and free environment space, the total and free alias list space, and the total command history space.

Windows NT provides virtual memory and swaps applications to disk as needed. This makes the size of the largest available free block of RAM as reported by MEMORY only a general reflection of available memory resources, and not a precise statement of the amount of RAM available for applications.

EXAMPLES

Display your RAM totals:

```
memory
```

SYNTAX (Internal 4NT)

MOVE [d:][path]filename... [d:][path][filename] [/C /D /H /N /P /Q /R /S /U]

PURPOSE

Move files to other directories and drives.

COMMENTS

The MOVE command moves the specified file(s) to the last filename specified, which is designated as the target. If the target already exists, it is deleted when the file is moved. You cannot move a file to itself.

If there is more than one source file specification, the target must be a directory, and the files are moved to the directory with their original filenames. If the target is not a directory, MOVE will display an error message and exit.

USE CAUTION when using MOVE commands with commands like SELECT. If multiple files are selected and the target is not a subdirectory, each file will be moved in turn to the target, overwriting the previous file. The net result is that all files but the last will be deleted. If SELECT is invoked using square brackets instead of parentheses, the MOVE will be done in a single command and MOVE will detect the error.

The MOVE options are:

- **/C** Move only files that exist in the target directory, and where the source file is newer than the target.
- **/D** The target must be a directory. If it doesn't exist, you will be prompted whether to create it.
- **/H** Move all files, including hidden & system.
- **/N** Do everything except actually move the file(s) (for testing what the result of a MOVE would be).
- **/P** Prompt for a Y or N response to confirm each move. A third option (R) turns off the confirmation and processes the remainder of the files normally.
- **/Q** Don't display filenames as they are moved.
- **/R** Prompt for a Y or N response before overwriting an existing file.
- **/S** Move an entire subdirectory tree to another location. The target directories will be created if they don't exist (except for the first target, which must already exist). MOVE will remove empty subdirectories after the move.
- **/U** Move only those files that either don't exist in the target directory, or where the source file is newer than the target.

MOVE first attempts to rename the file(s). If that fails (the target is on a different drive, or the target already exists), MOVE will copy the file(s) and then delete the originals. If MOVE cannot delete the original (for example, a read-only file), it will display an error message, but the target file is still created.

See also COPY and RENAME.

EXAMPLES

Move some files to a different directory:

```
move *.wks *.txt c:\finance\myfiles
```

Move all the files in the current directory to A:\, prompting before overwriting any existing files:

```
move /r *.* a:\
```

SYNTAX (Internal 4NT)

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

PURPOSE

Tells 4DOS for Windows NT where to search for executable and batch files not in the current directory.

COMMENTS

When searching for an external command (.COM, .EXE, .BTM, .BAT, .CMD, and optionally .REX, in that order), 4NT searches the current directory first, then the directories you specify in the PATH, in the order you list them. The directory names are separated by semicolons (;).

If you enter PATH with no parameters, PATH displays the current search path. If you enter PATH and a semicolon, PATH clears the search path and will search only the current directory (this is the default at system startup).

If you specify an invalid directory, 4NT will skip that directory and continue searching with the next directory in the path.

If you have a directory '.' in the PATH, 4NT will not search the current directory first, but will wait to do so until it reaches that point in the PATH.

Some applications also use the PATH variable to search for their files. See also DPATH.

You can edit an existing path with the ESET command.

EXAMPLES

The following PATH command sets the search path to the following order: the current directory, the root directory on drive C, the WINNT subdirectory on drive C, and the UTIL subdirectory on drive C:

```
path c:\;c:\winnt;c:\util
```

Display the current search path:

```
path
```

SYNTAX (Internal 4NT)

PAUSE [message]

PURPOSE

Suspend alias or batch file execution.

COMMENTS

A PAUSE command will suspend execution, giving you the opportunity to perform activities such as changing disks, turning on the printer, etc.

PAUSE waits for any key (except ^C or ^BREAK) to be pressed and then continues executing. If a prompt message is specified, PAUSE will display the message and wait for a keystroke. Otherwise, PAUSE will prompt:

    Press a key when ready...

If you press a ^C or ^BREAK while PAUSE is waiting for a key, execution of an alias will be terminated, and execution of a batch file will be suspended while you are prompted whether to cancel the batch file.

EXAMPLES

The following batch file fragment prompts the user before erasing files:

```
pause ^C aborts, any other key erases the .LST files:
erase *.lst
```

SYNTAX (Internal 4NT)

POPD [*]

PURPOSE

Change to the disk drive and directory at the top of the directory stack.

COMMENTS

POPD restores the disk and directory saved using PUSHD (most recent first).

The * option clears the directory stack, without changing the current drive and directory.

Use the DIRS command to display the directory stack.

See also DIRS and PUSHD.

EXAMPLES

Save and change your disk drive with PUSHD, and then restore it:

```
pushd d:\database\test
popd
```

Clear the directory stack:

```
popd *
```

SYNTAX (Internal 4NT)

PROMPT [text]

PURPOSE

Change the command line prompt.

COMMENTS

If you enter PROMPT with no parameters, PROMPT sets the prompt on drives A and B to [$n] (display current disk only), and on all other drives to [$p] (display current disk and directory).

The prompt text can contain special strings of the form $?, where ? is one of the following:

b	The | character
c	The open parenthesis (
D	Current date, in the format: Fri Dec 25, 1992
d	Current date, in the format: Fri 12-25-92
e	The ASCII ESC character (decimal 27)
f	The close parenthesis)
g	The > character
h	BACKSPACE over the previous character
l	The < character
n	Default drive letter
P	Current disk and directory (upper case)
p	Current disk and directory (lower case)
q	The = character
r	The numeric exit code returned by the last external command
s	The space character
t	Current time, in the format hh:mm:ss
v	Windows NT version number, in the format: 3.1
Xd:	Current disk and directory (upper case) where d: is the drive specification
xd:	Current disk and directory (lower case) where d: is the drive specification
z	Display the current shell nesting level (you must be running 4NT as the primary shell). The primary command processor is 0.
$	The $ character
_	CR/LF (go to beginning of new line)

You can embed any environment variable, internal variable, or variable function in your PROMPT string to view system status or resources. When defining the PROMPT to include variables of this type, use two % signs before the variable name, or enclose the prompt string in back-quotes [`] (if you don't, the prompt display will show the value the variable had when the prompt was defined, not the value at the moment the prompt is displayed). For example, to show free RAM to the left of the standard pg prompt:

```
prompt [%%@dosmem[K]K] $p$g
  or
prompt `[%@dosmem[K]K] $p$g`
```

See 4DOS for Windows NT Internal Variables and 4DOS for Windows NT Variable Functions for lists of the available variables and functions.

EXAMPLES

Set the prompt to the current date and time:

```
prompt $d $t $g
```

SYNTAX (Internal 4NT)

PUSHD [[d:]pathname]

PURPOSE

Save the current disk and directory.

COMMENTS

PUSHD saves the current directory on the directory stack, and if a pathname is specified on the command line, changes the disk drive and directory. The directory stack can hold up to 255 characters (about 10 to 20 entries). If you exceed the directory stack size, the oldest directory is removed before adding the current directory.

The saved directory is restored on a "last in, first out" basis by the POPD command. The directory stack can be displayed with the DIRS command.

See also DIRS and POPD.

EXAMPLES

Save the current directory and change to C:\WORDP\MEMOS:

```
pushd c:\wordp\memos
```

SYNTAX (Internal 4NT)

QUIT [value]

PURPOSE

Terminate the current batch file.

COMMENTS

You can QUIT at any point in a batch file, not just the end. If you QUIT a batch file called from another batch file, you will be returned to the previous file at the command following the original call. To end all batch file processing, use CANCEL. If you specify a value, QUIT will set the ERRORLEVEL to that value. The value also affects the internal variable %_?, and the conditional command separators && and ||.

You can also use QUIT in an alias. If you QUIT an alias while inside a batch file, QUIT will end both the alias and the batch file.

See also CANCEL.

EXAMPLES

Check to see if the user entered "quit" and exit if true:

```
input Enter your choice:  %%option
if "%option" == "quit" quit
```

SYNTAX (Internal 4NT)

RD [/S] [d:]path...
RMDIR [/S] [d:]path...

PURPOSE

Remove one or more directories.

COMMENTS

The /S option will remove the specified subdirectory tree(s), including all files. It is equivalent to a DEL /QSXZ.

Before removing a subdirectory, you must delete all the files and subdirectories (and their files) in that directory (including any hidden or read-only files).

You can use wildcard characters in the directory names.

You cannot remove the root directory (\), the current directory (.), or the parent directory (..).

See also MD.

EXAMPLES

Remove the subdirectory MEMOS from the directory WP on the current drive:

```
rmdir \wp\memos
```

SYNTAX (Internal 4NT)

REBOOT [/V]

PURPOSE

Reboot the system.

COMMENTS

The only option for REBOOT is:

/V Prompt for confirmation (Y or N) before rebooting.

REBOOT shuts down the file systems before rebooting the system, to allow the disk caches to finish writing any cached data.

EXAMPLES

The following command prompts you to verify the reboot:

```
reboot /v
```

SYNTAX (Internal 4NT)

REM [comment]

PURPOSE

Put a comment in a batch file.

COMMENTS

If ECHO is ON, or the SETDOS /V option is set, REM will echo the comment. Otherwise, REM will ignore it. If you don't want to echo the line, preface REM with the @ character. (Setting SETDOS /V to 2 will echo all lines, regardless of the echo state or @ symbols.) The current ECHO state is inherited by called batch files.

Enter your comment following the REM, separated by a space or tab. Comments can be up to 295 characters. Everything following a REM will be ignored, including quote characters, the redirection symbols (|><) and the command separator character. An alternate way to put a comment in a batch file is to begin the line with a colon (:). The line will be interpreted as a label and ignored. (Make sure you don't have any actual labels beginning with the same word!)

Batch file comments are useful for documenting the purpose for a batch file and the procedures used.

EXAMPLES

Enter comments in a batch file:

```
rem This batch file provides a menu-based system
rem     for accessing the rem word processing utilities.
rem Clear the screen and get a selection
cls
```

SYNTAX (Internal 4NT)

REN [/H /N /P /Q /S] [d:][path]filename... [d:][path]filename
RENAME [/H /N /P /Q /S] [d:][path]filename... [d:][path]filename

PURPOSE

Rename files or subdirectories.

COMMENTS

The last filename is the new name; preceding names are the files or directories to be renamed. You can use wildcard characters in the filenames. The new filename must not already exist. You cannot rename a subdirectory to a new path!

The REN options are:

- **/H** Rename all matching files, including hidden ones.
- **/N** Do everything except actually rename the file(s) (for testing what the result of a REN would be).
- **/P** Confirm each rename (Y or N). A third option (R) turns off the confirmation and processes the remainder of the files normally.
- **/Q** Don't display filenames as they are renamed.
- **/S** Include subdirectories in rename. Without this switch, REN will only rename directories if the target filename doesn't include wildcards.

REN in CMD.EXE only allows a path to be specified with the first file name; the path for the target is always the same as the source. 4NT allows you to optionally select the target path, renaming the source file to a new directory on the same disk drive. If you don't specify a path for the target, REN will assume a "CMD.EXE compatible RENAME" and will rename the source file to the new name without moving it to another directory.

If you want to rename to a different drive, use MOVE.

See also MOVE.

EXAMPLES

Rename the file MEMO.TXT to OFFICE.TXT:

```
ren memo.txt office.txt
```

Rename the directory \WORDPROC to \WP:

```
ren \wordproc \wp
```

SYNTAX (Internal 4NT)

RETURN

PURPOSE

Return from a GOSUB (subroutine) call in a batch file.

COMMENTS

The RETURN command returns from a GOSUB call to the command following the original GOSUB.

See also GOSUB.

EXAMPLES

The following batch file fragment calls a subroutine that displays the current directory:

```
echo Calling a subroutine
gosub subr1
echo Returned from the subroutine
quit
:subr1
dir /hw
return
```

SYNTAX (Internal 4NT)

SCREEN row column [message]

PURPOSE

Position the cursor on the screen, and optionally display a message.

COMMENTS

The row and column numbering is zero-based, so on a standard 25 line by 80 column display, valid rows are 0 - 24 and valid columns are 0 - 79. You can optionally specify relative cursor positioning (to the current position) by prefixing the row argument with a + or -.

SCREEN does not change the default screen colors. To display text in specific colors, use SCRPUT.

SCREEN checks for a valid row and column, and displays an error message if either value is out of range.

See also ECHO, SCRPUT, TEXT, and VSCRPUT.

EXAMPLES

The following batch file fragment displays a menu:

```
@echo off
cls
screen 3 10 Select a number from 1 to 4:
screen 6 20 1 - Word Processing
screen 7 20 2 - Spreadsheet
screen 8 20 3 - Telecommunications
screen 9 20 4 - Quit
```

SYNTAX (Internal 4NT)

SCRPUT row column [BRIGHT] fg ON [BRIGHT] bg text

PURPOSE

Display text in color.

COMMENTS

The row and column numbering is zero-based, so on a standard 25 line by 80 column display, valid rows are 0 - 24 and valid columns are 0 - 79. You can optionally specify relative cursor positioning (to the current position) by prefixing the row argument with a + or -.

SCRPUT is similar to SCREEN, but allows you to specify the display colors, and writes directly to the screen.

The parameters are:

row	Start row
column	Start column
fg	Foreground character color
bg	Background character color
text	The text to display

Only the first three characters of the color name and attribute ("bright") are required.

The available colors are:

Black	Blue	Green	Red
Magenta	Cyan	Yellow	White

See also ECHO, SCREEN, TEXT, and VSCRPUT.

EXAMPLES

The following batch file fragment displays a menu in color:

```
@echo off & cls white on blue
scrput 3 10 bri white on blue Select a number from 1 to 4:
scrput 6 20 bri red on blue 1 - Word Processing
scrput 7 20 bri yellow on blue 2 - Spreadsheet
scrput 8 20 bri green on blue 3 - Telecommunications
scrput 9 20 bri magenta on blue 4 - Quit
```

SYNTAX (Internal 4NT)

SELECT [/A[:-rhsda] /C /D /O:-deginrsu /Z] command ([path]filename)

PURPOSE

Execute a command on the specified files.

COMMENTS

SELECT allows you to select command line file arguments by marking the files using a full-screen "point-and-shoot" display. SELECT substitutes the selected files for the argument enclosed in parentheses, and executes the command for each marked file. If you specify multiple arguments in the parentheses, SELECT will display the matching files for the first argument, prompt you to mark the files, execute the command for each marked file, and then continue the same procedure with the next argument.

The SELECT options are:

- **/A** Display only those files that have the specified attribute set. Preceding the attribute character with a '-' will display those files that DON'T have that attribute set. Attributes can also be combined. The attributes are:
  - **R** Read only
  - **H** Hidden
  - **S** System
  - **D** Directory
  - **A** Archive
- **/C** Display filenames in upper case.
- **/D** Disable directory colorization.
- **/O** Sort sequence, where ? can be any combination of the following:
  - **-** Reverse the sort order for the next option
  - **a** Sort by ASCII value rather than numeric value
  - **d** Sort by date and time (oldest first)
  - **e** Sort by extension
  - **g** Group subdirectories together
  - **i** Sort by the file description
  - **n** Sort by the filename (this is the default)
  - **r** Reverse the sort order for all options
  - **s** Sort by size
  - **u** Unsorted
- **/Z** Display NTFS and HPFS volumes in FAT format, including file descriptions and truncating filenames to 11 characters.

If you append filenames with a ; (an "include list"), SELECT will display the matching filenames in a single listing. Only the first filename in an include list can have a path.

You can display the file and subdirectory names in color by setting the COLORDIR variable. The format for COLORDIR is:

ext [...]:[BRIGHT] fg [ON [BRIGHT] bg]; ...

where "ext" is the file extension, or one of the following file attributes:

DIRS - directory
RDONLY - read-only file
HIDDEN - hidden file

SYSTEM - system file
ARCHIVE - file modified since last backup

For example, to display the .COM and .EXE files in red, the .C and .ASM files in bright cyan, and the read-only files in green:

```
set colordir=com exe:red; c asm:bright cyan; rdonly:green
```

If you don't select a background color, SELECT will use the current screen background color.

If you enclose the arguments in square brackets [ ] rather than parentheses, SELECT will insert all of the arguments into the command line and only execute the command once. You must be careful not to exceed the maximum line length (1023 characters).

SELECT uses the cursor up, cursor down, PgUp, and PgDn keys to scroll through the files matching the argument(s). Use the + key to select a file, and the - key to unselect a file. The * key will reverse all of the current marks (excluding subdirectories), and the / key will unmark everything. After marking the files, press ENTER to execute the command. You can select a single file by moving the scroll bar to the filename and pressing ENTER.

To cancel the SELECT command, type an Escape, ^C, or ^BREAK.

EXAMPLES

Select from the files in the current directory with the extension .COM, and then from the files with the extension .EXE to copy to the root directory on drive A:

```
select copy (*.com *.exe) a:\
```

Select and run an executable program (.COM or .EXE) from files in the C:\UTIL directory (an "include list"):

```
select (c:\util\*.com;*.exe)
```

SYNTAX (Internal 4NT)

SET [/P] [/R [d:][path]filename...] [variable=[value]]

PURPOSE

Display, create, modify, or delete environment variables.

COMMENTS

Entering SET with no variable or value displays the entire environment. If you don't enter a value, SET will display the value of that variable. Otherwise, the variable and value are placed in the environment. If the variable already exists, its old contents are replaced by the new value. Variable names are limited to a maximum of 80 characters, and the value to a maximum of 255 characters. The variable names are shifted to upper case before being stored in the environment; the value is stored exactly as entered.

The SET options are:

/P Pause after displaying a page of environment entries. Press ^C to quit, or any other key to display the next page.

/R Read environment variables from a file. This is much faster than loading variables in a batch file. The file is in the same format as the SET display, so SET /R can accept as input a file generated by redirecting SET output. If the last character on a line is an escape character (default ^X), 4NT will remove the escape character and append the next line.

For example, the following commands will save the environment variables to a file, and then reload them from that file:

```
set > varlist
set /r varlist
```

You can load variables from multiple files by listing the filenames individually after the /R.

You can add comments to a variable file by starting the comment line with a colon (:).

You can remove multiple environment variables with the UNSET command, and edit environment variables with the ESET command.

See also ESET and UNSET.

EXAMPLES

Display the current environment:

```
set
```

Add a variable to the environment:

```
set mine=c:\finance\myfiles
```

Remove a variable from the environment:

```
set mine=
```

SYNTAX (Internal 4NT)

SETDOS [/C? /D? /E? /I+|-command /M? /N? /P? /S?:? /U? /V?]

PURPOSE

Display or set 4DOS for Windows NT configuration options.

COMMENTS

Most of these options can also be set in 4NT.INI. The name of the corresponding 4NT.INI directive for each option, if any, is listed in square brackets [ ] at the end of the description for the option. Use SETDOS rather than 4NT.INI when you need to make temporary changes.

The SETDOS options are:

/C The COMPOUND option sets the character used for separating multiple commands on the same line. The default character is the ampersand (&). You cannot use any of the redirection characters (|><), or the whitespace characters (blank, tab, comma, semicolon, or equal) as the command separator. [CommandSep]

/D The DESCRIPTION option allows you to disable or enable description processing. 4NT normally updates the description files (DESCRIPT.ION) whenever you do a file command such as COPY, DEL, MOVE, REN, etc.

/E The ESCAPE option sets the character used to suppress the normal meaning of the following character. Any character following the escape character will be passed unmodified to the command line. The default escape character is the caret (^). You cannot use any of the redirection characters (|><) or the whitespace characters (blank, tab, comma, semicolon, or equal) as the escape character. [EscapeChar]

/I The INTERNAL option allows you to disable or enable internal 4NT commands. To disable a command, precede the command name with a minus (-). To reenable a command, precede it with a plus (+).

/M The MODE option controls the line editing mode. The default is overstrike mode (/M0). If MODE is set to 1, the default is insert mode. [EditMode]

/N The NOCLOBBER option controls output redirection. If NOCLOBBER is set to 1, existing files may not be destroyed by output redirection, and when appending with >>, the output file must exist. NOCLOBBER can be overridden with the ! character. The default value is 0. [NoClobber]

/P The PARAMETER option sets the character used to specify argument lists in alias and variable expansion. The default character is $ (in 4DOS it is &). [ParameterChar]

/S The SHAPE option sets the default cursor shape. The format is /So:i, where "o" is the cursor size for overstrike mode, and "i" the cursor size for insert mode. The size is entered as a percentage of the total character cell size. The default values are 10:100 (a thin underscore cursor for overstrike mode, and a block cursor for insert mode). Because of the way video BIOSes remap the cursor, you may not get a smooth progression in the cursor size from 0 - 100%. To disable the cursor, use /S0:0. [CursorOver, CursorIns]

/U The UPPER option controls the way filenames are displayed for the internal commands (COPY, DIR, etc.). If UPPER is 1, filenames will be displayed in upper case. The default is /U0 (filenames will be displayed in lower case). [UpperCase]

/V The VERBOSE option controls command echoing in batch files. If VERBOSE is set to 0, batch files will NOT be echoed unless ECHO is set ON. The default is 1 (batch file commands WILL be echoed). If VERBOSE is set to 2, all batch file lines will be echoed, including those prefaced by a '@', and when ECHO is OFF. [BatchEcho]

EXAMPLES

Change the COMPOUND character to a ~ (tilde):

```
setdos /c~
```

Change MODE to insert, VERBOSE to off, and set NOCLOBBER on:

```
setdos /m1 /v0 /n1
```

Disable the internal LIST command:

```
setdos /i-list
```

Change the the PARAMETER character to an &, the COMPOUND character to a ^, and the ESCAPE character to a _ (ctrl-X), so that 4NT syntax matches that used in 4DOS. Note the ^ escape characters used before the new parameter and compound characters. These prevent the new characters from being given their old meanings within the SETDOS command; for example, without them the & would be taken as the end of the SETDOS command.

```
setdos /p^& /c^^ /e_
```

SYNTAX (Internal 4NT)

SETLOCAL

PURPOSE

Save a copy of the current disk drive, directory, aliases, and environment variables.

COMMENTS

SETLOCAL is used in batch files to save the disk drive, directory, aliases, and environment variables to a reserved block of memory. You can then change their values, and later restore the original values with ENDLOCAL. You cannot use SETLOCAL in an alias.

SETLOCAL and ENDLOCAL are not nestable within the same batch file. However, you can have multiple SETLOCAL / ENDLOCAL pairs within a batch file, and nested batch files can each have their own SETLOCAL / ENDLOCAL. An ENDLOCAL will be automatically performed at the end of a batch file.

Note that if you invoke a batch from another without using CALL, the first batch file is implicitly terminated, and an automatic ENDLOCAL performed. The second batch file will inherit the drive, directory, aliases, and environment variables as they were prior to the SETLOCAL.

See also ENDLOCAL.

EXAMPLES

This batch file fragment saves the disk drive, directory, aliases, and environment variables, changes the disk and directory, modifies some variables, runs a program, and then restores the original values:

```
setlocal
cdd d:\test & set path=c:\;c:\os;c:\util & set lib=d:\lib
rem run some program here
endlocal
```

SYNTAX (Internal 4NT)

SHIFT [n]

PURPOSE

Allow the use of more than 10 variables in a batch file.

COMMENT

SHIFT is provided for compatibility with CMD.EXE, which only supports 10 parameters (%0 through %9); 4NT supports 128 (%0 to %127), so you may not need to use SHIFT for batch files running exclusively under 4NT.

SHIFT moves each batch file parameter n positions (i.e., if n is 1, then %1 becomes %0, %2 becomes %1, etc.). The default value for n is 1. You can also reverse a SHIFT by giving a negative value for n (i.e., if n is -1, the former value for %0 is restored, %1 becomes %1, %1 becomes %2, etc.).

SHIFT also affects the %$ (command line tail) and %# (number of command arguments) batch variables.

EXAMPLES

Create a batch file called TEST.BTM:

```
echo %1 %2 %3 %4
shift
echo %1 %2 %3 %4
shift 2
echo %1 %2 %3 %4
shift -1
echo %1 %2 %3 %4
```

Executing TEST.BTM with the arguments "zero one two three four five six" produces the following results:

```
zero one two three
one two three four
three four five six
two three four five
```

SYNTAX (Internal 4NT)

START ["title"] /B /C /Dpath /HIGH /I /INV /K /L /LA /LH /LOW /MAX /MIN /NORMAL /PGM
/POS=x,y,x1,y1 /REALTIME /SIZE=x,y /WAIT [command ...]

PURPOSE

Start a program in a new window.

COMMENTS

Entering START with no program name starts another 4DOS for Windows NT console mode window.

The START options are:

"title"	Specify the title to appear on the title bar. The title must be enclosed in double quotes. If you don't specify a title, the program name is used.
/B	Start the program without a new window.
/C	Start the application under 4NT, and close the new window when the application finishes.
/D	Specify the startup directory.
/HIGH	Start as a high priority program.
/I	Inherit the environment originally passed to 4NT, rather than the current 4NT environment.
/INV	Start the window hidden ("invisible").
/K	Start the program in a new 4NT window, and remain in the new window when the application finishes.
/L	Start 4NT with local alias and history lists.
/LA	Start 4NT.EXE with a local alias list. If you don't use this option, the new 4NT will share the alias list of the current 4NT shell, unless a LocalAliases=No directive is used in 4NT.INI or on the command line.
/LH	Start 4NT.EXE with a local history list. If you don't use this option, the new 4NT will share the history list of the current 4NT shell, unless a LocalHistory=No directive is used in 4NT.INI or on the command line.
/LOW	Start a low priority program (it is only run when the system is idle).
/MAX	Start the window maximized.
/MIN	Start the window minimized.
/NORMAL	Start the program as a normal application with no special scheduling needs. (this is the default).
/PGM	The next argument is the program name, not a title.
/POS	Specify the start position (x,y) in pixels, and the window size (x1,y1) in rows and columns.
/REALTIME	Start as a realtime priority program.
/SIZE	Specify the screen buffer size in rows and columns.
/WAIT	Wait for the program to end before continuing.

See also DETACH.

EXAMPLES

Start CHKDSK minimized, with the title "Analyzing the disk":

```
start "Analyzing the disk" /min chkdsk
```

SYNTAX (Internal 4NT)

TEE [/A] [d:][path]filename...

PURPOSE

Copy standard input to standard output, and save a copy in the specified file(s).

COMMENTS

The only option for TEE is:

/A Append output to the file(s) rather than overwriting it.

If you are typing at the keyboard, enter a ^Z to terminate the input.

TEE is often used to save the intermediate output of a pipe.

See also Y.

EXAMPLES

Search the file DOC for any lines containing the string "486", make a copy of the matching lines in 4.DAT, and write them to the output file 486.DAT:

```
find "486" doc | tee 4.dat | sort > 486.dat
```

SYNTAX ***(Internal 4NT)***

TEXT
.
.
.
ENDTEXT

PURPOSE

Display a block of text in a batch file.

COMMENTS

The TEXT command is useful for displaying menus or multiple-line messages. TEXT will display subsequent lines in the text until terminated by ENDTEXT. Both TEXT and ENDTEXT must be entered as the only command on that line.

You can change screen colors by inserting ANSI escape sequences into the text block.

If the output of TEXT is redirected, all lines in the text block will be written to the specified output file, and not to the screen.

See also ECHO, SCREEN, SCRPUT, and VSCRPUT.

EXAMPLES

The following batch file fragment displays a simple menu:

```
@echo off & cls & screen 2 0
text
Enter one of the following:

    1 - Spreadsheet
    2 - Word Processing
    3 - DOS Utilities

Enter your selection:
endtext
```

SYNTAX (Internal 4NT)

TIME [hh:mm:ss]

PURPOSE

Display or set the current system time.

COMMENTS

If you don't enter any parameters, TIME will display the current system time and prompt you for a new time. Press ENTER if you don't wish to change the time, otherwise enter the new time.

The parameters for the TIME command are:

hh Hour (0 - 23)
mm Minute (0 - 59)
ss Second (0 - 59)

TIME defaults to 24-hour format; you can optionally enter the time in 12-hour format by appending an "am" or "pm".

Whenever you create or modify a file, the system time is recorded in the directory entry.

See also DATE and TIMER.

EXAMPLES

Enter the time (9:30 am):

```
time 9:30
```

To be prompted for the time:

```
time
```

SYNTAX (Internal 4NT)

TIMER [/1 /2 /3 /S] [ON]

PURPOSE

System stopwatch.

COMMENTS

The TIMER command turns an internal stopwatch on and off. The first time you run TIMER, the stopwatch starts. When you run TIMER again, the stopwatch stops and the elapsed time is displayed. There are three timers available (1, 2, and 3), so you can time multiple overlapping events. The default timer is #1.

The options for TIMER are:

ON	Force the timer to restart
/1	Use timer #1
/2	Use timer #2
/3	Use timer #3
/S	Display split time without stopping the timer

TIMER is particularly useful for timing events in batch files.

The smallest interval TIMER can measure 0.03 seconds; the largest interval is 23:59:59.99.

See also TIME.

EXAMPLES

Start or stop the timer:

```
timer
```

Display a split time:

```
timer /s
```

SYNTAX (Internal 4NT)

TITLE text

PURPOSE

Change the current window title

COMMENTS

This command is included for compatibility with CMD.EXE. You can also change window titles in 4NT with the WINDOW command.

See also WINDOW.

EXAMPLES

Change the window title:

```
title Testing 4DOS for Windows NT
```

SYNTAX (Internal 4NT)

TYPE [/L /P] [d:][pathname]filename...

PURPOSE

Display the contents of the specified file(s).

COMMENTS

The TYPE command displays a file. Press ^S to suspend the display, and any character key to continue the display.

The TYPE options are:

/L Print line numbers preceding each line of text.
/P Pause after each page. Press ^C to quit, or any other key to display the next page.

TYPE is normally only useful for displaying ASCII text files; executable files (.COM and .EXE) and many data files will be unreadable due to the presence of non-alphanumeric characters.

You will probably find LIST to be more useful for displaying files.

See also LIST.

EXAMPLES

Display the files MEMO1 and MEMO2, pausing at the end of each page:

```
type /p memo1 memo2
```

SYNTAX (Internal 4NT)

UNALIAS [/Q] alias...
UNALIAS *

PURPOSE

Remove aliases from the alias list.

COMMENTS

UNALIAS also accepts the wildcard character * to delete all aliases.

The only option for UNALIAS is:

/Q Don't display an error message if the alias doesn't exist.

See also ALIAS and ESET.

EXAMPLES

Remove the alias DDIR:

```
unalias ddir
```

Remove the aliases DDIR and ZAP:

```
unalias ddir zap
```

Remove all the aliases:

```
unalias *
```

SYNTAX (Internal 4NT)

UNSET [/Q] name...
UNSET *

PURPOSE

Remove variables from the environment.

COMMENTS

UNSET also accepts the wildcard character * to delete all environment variables. Use caution with UNSET *; many programs are dependent on environment variables. (4DOS for Windows NT relies on PATH and COMSPEC.)

The only option for UNSET is:

/Q Don't display an error message if the environment variable doesn't exist.

UNSET is often used in conjunction with SETLOCAL / ENDLOCAL in order to clear the environment of variables that may cause problems for some applications.

See also ESET and SET.

EXAMPLES

Remove the variable CMDLINE:

```
unset cmdline
```

Remove the variables CMDLINE and PATH:

```
unset cmdline path
```

Remove all the environment variables:

```
unset *
```

SYNTAX (Internal 4NT)

VER [/R]

PURPOSE

Display the current 4DOS for Windows NT and Windows NT versions.

COMMENTS

The 4NT and Windows NT versions consist of a one digit major version number, a period, and a one or two digit minor version number.

The only option for VER is:

/R Display the Windows NT and 4DOS for Windows NT revision levels.

EXAMPLES

Get the current version of 4NT and Windows NT:

```
ver
```

SYNTAX (Internal 4NT)

VERIFY [on | off]

PURPOSE

Compatibility only; it has no effect in Windows NT.

COMMENTS

VERIFY is only included for compatibility with existing batch files.

EXAMPLES

Check the current verify status:

```
verify
```

SYNTAX (Internal 4NT)

VOL [d:] ...

PURPOSE

Display the disk volume label(s).

COMMENTS

If you don't enter a drive name, VOL displays the disk label and the volume serial number for the current drive.

If the disk doesn't have a volume name, VOL will report it as "unlabeled."

Volume labels can be created, changed or deleted with the LABEL command.

EXAMPLES

Display the disk labels for drives A and B:

```
vol a: b:
```

SYNTAX (Internal 4NT)

VSCRPUT row column [BRIGHT] fg ON [BRIGHT] bg text

PURPOSE

Display text in color in a vertical column.

COMMENTS

The row and column numbering is zero-based, so on a standard 25 row by 80 column display, valid rows are 0 - 24 and valid columns are 0 - 79. You can optionally specify relative cursor positioning (to the current position) by prefixing the row argument with a + or -.

VSCRPUT works like SCRPUT, but writes the text vertically.

The parameters are:

row	Start row
column	Start column
fg	Foreground character color
bg	Background character color
text	The text to display

Only the first three characters of the color name and attribute ("bright") are required. The available colors are:

Black	Blue	Green	Red
Magenta	Cyan	Yellow	White

See also ECHO, SCREEN, SCRPUT, and TEXT.

EXAMPLES

The following batch file fragment displays an X and Y axis and labels them:

```
drawhline 20 10 40 1 bright white on blue
drawvline 2 10 18 1 bright white on blue
scrput 21 20 bright red on blue X axis
vscrput 8 9 bright red on blue Y axis
```

SYNTAX (Internal 4NT)

WINDOW ["title" | MAX | MIN | RESTORE]

PURPOSE

Change the window title, maximize or minimize the current 4DOS for Windows NT window, or revert to the default size.

COMMENTS

If you specify a title, it must be enclosed in double quotes. The quotes will not appear in the actual title.

EXAMPLES

Minimize the 4NT window:

```
window min
```

SYNTAX (Internal 4NT)

Y [d:][path]filename...

PURPOSE

Copy standard input to standard output, and then copy the specified file(s) to standard output.

COMMENTS

Standard input from the console is terminated by a ^Z.

See also TEE.

EXAMPLES

Get text from standard input, append the files MEMO1 and MEMO2 to it, and send the output to MEMOS:

```
y memo1 memo2 > memos
```

Key mapping

The following table lists the common extended key codes used by INKEY, KEYSTACK, and key aliases:

F1 @59	Alt-F1 @104	Ctrl-F1 @94	Shift-F1 @84
F2 @60	Alt-F2 @105	Ctrl-F2 @95	Shift-F2 @85
F3 @61	Alt-F3 @106	Ctrl-F3 @96	Shift-F3 @86
F4 @62	Alt-F4 @107	Ctrl-F4 @97	Shift-F4 @87
F5 @63	Alt-F5 @108	Ctrl-F5 @98	Shift-F5 @88
F6 @64	Alt-F6 @109	Ctrl-F6 @99	Shift-F6 @89
F7 @65	Alt-F7 @110	Ctrl-F7 @100	Shift-F7 @90
F8 @66	Alt-F8 @111	Ctrl-F8 @101	Shift-F8 @91
F9 @67	Alt-F9 @112	Ctrl-F9 @102	Shift-F9 @92
F10 @68	Alt-F10 @113	Ctrl-F10 @103	Shift-F10 @93

Home @71	Up @72	PgUp @73
Left @75		Right @77
End @79	Down @80	PgDn @81
Ins @82	Del @83	

Ctrl-Home @119	Ctrl-PgUp @132
Ctrl-Left @115	Ctrl-Right @116
Ctrl-End @117	Ctrl-PgDn @118

Ctrl-A 1	Ctrl-N 14
Ctrl-B 2	Ctrl-O 15
Ctrl-C 3	Ctrl-P 16
Ctrl-D 4	Ctrl-Q 17
Ctrl-E 5	Ctrl-R 18
Ctrl-F 6	Ctrl-S 19
Ctrl-G 7	Ctrl-T 20
Ctrl-H 8	Ctrl-U 21
Ctrl-I 9	Ctrl-V 22
Ctrl-J 10	Ctrl-W 23
Ctrl-K 11	Ctrl-X 24
Ctrl-L 12	Ctrl-Y 25
Ctrl-M 13	Ctrl-Z 26

Alt-A @30	Alt-N @49	Alt-1 @120
Alt-B @48	Alt-O @24	Alt-2 @121
Alt-C @46	Alt-P @25	Alt-3 @122
Alt-D @32	Alt-Q @16	Alt-4 @123
Alt-E @18	Alt-R @19	Alt-5 @124
Alt-F @33	Alt-S @31	Alt-6 @125
Alt-G @34	Alt-T @20	Alt-7 @126
Alt-H @35	Alt-U @22	Alt-8 @127
Alt-I @23	Alt-V @47	Alt-9 @128
Alt-J @36	Alt-W @17	Alt-0 @129
Alt-K @37	Alt-X @45	
Alt-L @38	Alt-Y @21	
Alt-M @50	Alt-Z @44	