Dos: introduzione

DOS è acronimo di *Disk Operating System* e sta a indicare il nome di un sistema operativo per micro elaboratori basati su microprocessori i86, successore del vecchio CP/M. Probabilmente, data la sua estrema limitatezza, è un po' azzardato voler parlare di «sistema operativo», tanto che qualcuno lo appella: «gestore di interruzioni» (*interrupt*).

Questo sistema operativo nasce come software proprietario; tuttavia, attualmente il progetto più attivo attorno a questo tipo di sistema è FreeDOS, il cui scopo è quello di realizzarne un'edizione libera e completa.

Avvio del sistema

«

Un sistema Dos è composto essenzialmente da un kernel, un interprete dei comandi e da una serie di programmi di servizio. Questo concetto è analogo ai sistemi Unix, con la differenza che il kernel offre funzionalità molto scarse e solo per mezzo di interruzioni software (IRQ).

Nelle versioni proprietarie del Dos, il kernel è suddiviso in due file, che raccoglievano funzionalità distinte in base all'importanza relativa. I nomi usati sono stati differenti e nel caso di FreeDOS il kernel è contenuto tutto in un solo file (tabella u179.1).

Tabella u179.1. Comparazione tra i nomi dei file che compongono il kernel di un sistema Dos.

Microsoft	IBM	Novell, Caldera	RxDOS	FreeDOS
IO.SYS	IBMBIO.COM	IBMBIO.COM	RXDOSBIO.SYS	KERNEL.SYS
MSDOS.SYS	IBMDOS.COM	IBMDOS.COM	RXDOS.SYS	--

I file del kernel devono trovarsi nella directory radice della partizione o del dischetto per poter essere avviati. Per la precisione, l'avvio del kernel viene gestito direttamente dal codice inserito nel settore di avvio della partizione o del dischetto (512 Kibyte), che a sua volta viene avviato dal firmware (il BIOS, secondo la terminologia specifica dell'architettura i86 e successiva).

Il kernel, dopo essere stato avviato, non attiva una procedura di avvio, ma si limita a interpretare uno script speciale, '**CONFIG.SYS**', e subito dopo avvia l'interprete dei comandi, ovvero la shell. Tradizionalmente, il programma in questione è '**COMMAND.COM**'. Secondo la tradizione, l'interprete dei comandi che viene avviato dal kernel si occupa subito di eseguire lo script '**AUTOEXEC.BAT**'. Gli script '**CONFIG.SYS**' e '**AUTOEXEC.BAT**' devono trovarsi nella directory radice del disco o della partizione da cui si avvia il sistema, ovvero quella in cui si trova già il kernel che viene avviato.

L'interprete dei comandi, '**COMMAND.COM**', è in grado di eseguire direttamente alcune funzionalità, attraverso comandi interni che non si traducono in programmi di servizio veri e propri. Tradizionalmente '**COMMAND.COM**' si colloca nella directory radice del disco o della partizione in cui si trova il kernel stesso. Ciò non è propriamente in-

dispensabile, ma conviene attenersi a questa linea per evitare fastidi inutili.

Dispositivi secondo il Dos

«

I dispositivi secondo il Dos hanno un nome, composto da lettere e cifre numeriche, terminato da due punti opzionali:

nome_dispositivo [:]

Il nome in questione può essere indicato utilizzando lettere maiuscole o minuscole, senza che la cosa faccia differenza. I nomi più comuni sono elencati nella tabella u179.2. È il caso di osservare che i due punti che concludono il nome, vanno usati necessariamente quando questo viene abbinato ad altre informazioni da cui non potrebbe essere distinto (per esempio un percorso).

Tabella u179.2. Nomi dei dispositivi più comuni in Dos.

Dispositivo	Descrizione
'A:'	Disco nella prima unità a dischetti.
'B:'	Disco nella seconda unità a dischetti.
'C:'	Prima partizione Dos nel primo disco fisso.
'D:', 'E:',... 'Z:'	Partizione Dos o altro tipo di disco.
'CON:'	Console: tastiera e schermo.
'PRN:'	Porta stampante principale.
'LPT1:', 'LPT2:',...	Porte parallele.
'COM1:', 'COM2:',...	Porte seriali.

Il Dos mantiene distinti i dischi e le partizioni, nel senso che questi non devono creare una struttura unica come avviene nei sistemi Unix. Pertanto, quando si fa riferimento a un percorso di un file o

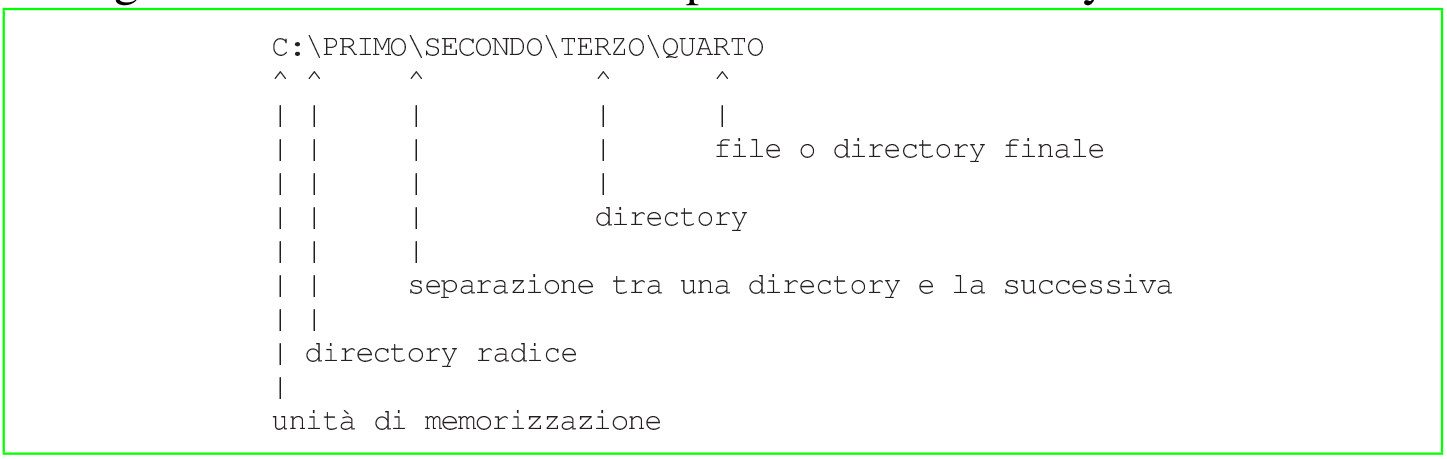
di una directory, si deve tenere in considerazione anche il disco o la partizione in cui si trova.

Il modo utilizzato dal Dos per identificare i dischi e le partizioni, di fatto impedisce di accedere a questi dispositivi in modo indipendente dal file system sottostante. Per intenderci, l'«unità» '`X:`' può essere una partizione Dos di un disco non meglio identificato; mentre non esiste un modo univoco per poter raggiungere il dispositivo fisico in cui si trova questo disco.

Directory, file ed eseguibili

Il Dos è nato dopo Unix e da questo sistema ha ereditato alcuni concetti elementari (forse troppo pochi). I percorsi di file e directory si separano con una barra obliqua, che però è inversa rispetto allo Unix. Anche con il Dos c'è una directory radice; tuttavia si aggiunge l'indicazione dell'unità di memorizzazione (il disco o la partizione). Si può osservare a questo proposito la figura u179.3.

Figura u179.3. Struttura di un percorso in un file system Dos.



I nomi di file e directory possono essere indicati utilizzando lettere maiuscole o minuscole, senza che la cosa possa fare differenza. Questi nomi possono essere composti utilizzando anche cifre numeriche e altri simboli (che comunque è bene usare con parsimonia).

Per la precisione, sono esclusi i simboli: '`/`', '`\`', '`[`', '`]`', '`<`', '`>`', '`+`', '`=`', '`;`', '`:`', '`,`', '`?`', '`*`', '`{`', '`}`' e il punto che va usato esattamente come descritto nel seguito.

Tradizionalmente, il Dos utilizza un tipo di file system elementare, denominato FAT (Dos-FAT), in cui i nomi dei file e delle directory possono essere composti utilizzando al massimo 11 caratteri, di cui otto compongono un prefisso e tre un suffisso. Il prefisso e il suffisso di questi nomi appaiono uniti attraverso un punto. Per esempio: '`CIAO.COM`', '`LETTERA.TXT`', '`PIPPO.NB`',... Questa conformazione dei nomi è una caratteristica fondamentale del Dos, da cui deriva una serie di consuetudini e di limitazioni molto importanti.

È importante osservare che non è opportuno che i nomi dei file coincidano con quelli dei dispositivi (senza i due punti finali). In pratica, non conviene creare file del tipo '`CON:`', '`PRN:`', ecc. Tutto dipende dal contesto, ma in generale è bene fare attenzione a questo particolare.

Come nei sistemi Unix il Dos annovera il concetto di directory corrente, a cui si aggiunge il concetto di unità di memorizzazione corrente. Infatti, la directory va collocata in un disco o in una partizione. In base a questo principio, si possono indicare dei percorsi relativi, che fanno riferimento alla posizione corrente (nell'unità di memorizzazione corrente). Tuttavia, in più, ogni unità di memorizzazione ha una sua directory corrente. Per esempio, fare riferimento a un file in una certa unità di memorizzazione '*x*`:`', senza specificare il

percorso, significa indicare implicitamente la directory corrente di quella unità.

Per esempio, supponendo che la directory corrente dell'unità '`X:`' sia '`X:\PRIMO\SECONDO\`', facendo riferimento al file '`X:CIAO`', si intende indicare implicitamente il file '`X:\PRIMO\SECONDO\CIAO`'.

In un percorso si possono usare anche i simboli '`.`' e '`..`', con lo stesso significato che hanno in un sistema Unix: la directory stessa e la directory genitrice.

Il file system tradizionale del Dos consente di annotare solo poche informazioni per i file e le directory: la data di modifica e quattro indicatori booleani, rappresentati da altrettante lettere:

- H file o directory nascosti;
- S file o directory di sistema;
- R file o directory in sola lettura e non cancellabile;
- A file o directory da archiviare (i dati sono stati modificati).

Si tratta di attributi completamente differenti da quelli di Unix. Si può osservare in particolare la mancanza di un attributo che specifichi la possibilità di eseguire un programma o di attraversare una directory. Secondo la tradizione Dos, gli attributi vanno considerati nel modo seguente:

- A viene attivato ogni volta che il file viene scritto o modificato e serve per automatizzare i sistemi di copia periodica;
- R se attivo, il Dos non consente la scrittura o la rimozione;
- S se attivo si tratta di un file di «sistema», ma in pratica si comporta come l'attributo H;

- H se attivo si tratta di un file «nascosto», che così non dovrebbe apparire nelle liste di file e directory.

In generale, file e directory nascosti o di sistema non dovrebbero essere spostati fisicamente, nemmeno nell'ambito della stessa unità di memorizzazione. Questa esigenza nasce in particolare per i file del kernel, che non possono essere spostati se si vuole poter riavviare il sistema operativo.

Dal momento che il file system non permette di determinare se un file è un eseguibile, l'unico modo per permettere al sistema di conoscere questa caratteristica sta nell'uso di suffissi convenzionali nei nomi: i file che terminano con l'estensione '`.COM`' e '`.EXE`' sono programmi binari (la differenza tra i due tipi di estensione riguarda il formato del binario); quelli che terminano per '`.BAT`' sono script dell'interprete dei comandi ('**`COMMAND.COM`**').

La prima stranezza che deriva da questa caratteristica del Dos sta nel fatto che per avviare un eseguibile di questi, è sufficiente indicare il nome del file senza l'estensione, che diventa così un componente opzionale agli occhi dell'utilizzatore.

Comandi e ambiente di avvio

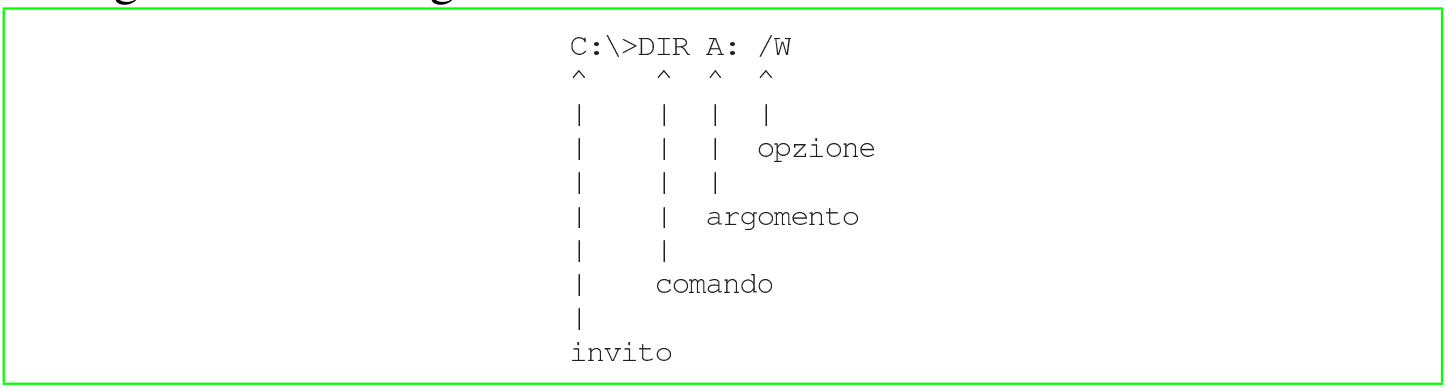
«

L'interprete dei comandi tradizionale dei sistemi Dos è il programma '**`COMMAND.COM`**', che viene avviato direttamente dal kernel. '**`COMMAND.COM`**' può essere avviato più volte successive, anche se di solito ciò è di scarsa utilità, dal momento che il Dos non è

un sistema operativo in multiprogrammazione. In ogni caso, quando viene avviato dal kernel, si occupa di interpretare ed eseguire lo script '**AUTOEXEC.BAT**' che si trova nella directory radice dell'unità di avvio.

'**COMMAND.COM**' mostra un invito simile idealmente a quello delle shell Unix, dopo il quale possono essere inseriti i comandi. A loro volta, questi possono essere riferiti a ***comandi interni*** corrispondenti a funzionalità offerte direttamente dall'interprete, oppure possono rappresentare la richiesta di avvio di un programma esterno.

Figura u179.4. Riga di comando.



Il Dos ha ereditato da Unix anche il concetto di variabile di ambiente. Il meccanismo è lo stesso ed è fondamentale la variabile di ambiente '**PATH**', con la quale si possono indicare i percorsi di ricerca degli eseguibili. Tuttavia, il Dos ha delle caratteristiche speciali, per cui, è il caso di fare alcuni esempi di comandi:

- `C:\>`**`C:\PRIMO\SECONDO.EXE`** [*Invio*]

  questo comando avvia l'esecuzione del file '`C:\PRIMO\SECONDO.EXE`';

- `C:\>`**`C:\PRIMO\SECONDO`** [*Invio*]

questo comando potrebbe avviare l'esecuzione del primo dei file seguenti che riesce a trovare;

- 'C:\PRIMO\SECONDO.COM'
- 'C:\PRIMO\SECONDO.EXE'
- 'C:\PRIMO\SECONDO.BAT'

- C:\>**SECONDO** [*Invio*]

  questo comando potrebbe avviare l'esecuzione del primo dei file seguenti che dovesse riuscire a trovare, ma in mancanza può continuare la ricerca nei percorsi indicati nella variabile di ambiente '**PATH**'.

  - 'C:.\SECONDO.COM'
  - 'C:.\SECONDO.EXE'
  - 'C:.\SECONDO.BAT'

I percorsi indicati nella variabile di ambiente '**PATH**' sono separati da un punto e virgola; per esempio:

```
C:\;C:\DOS;C:\FDOS\BIN
```

Di solito, il Dos dà per scontato che si cerchino gli eseguibili a cominciare dalla directory corrente. Per questo, occorre considerare che è sempre come se la variabile di ambiente '**PATH**' contenesse questa indicazione prima delle altre: '**.;C:\;C:\DOS;C:\FDOS\BIN**'. È da osservare che FreeDOS si comporta in maniera differente, in quanto richiede espressamente questa indicazione della directory corrente.

Caratteri jolly

«

Il Dos imita l'utilizzo dei caratteri jolly come avviene nei sistemi Unix per opera delle shell. Tuttavia, nel Dos non si tratta di un'espansione che avviene per opera della shell, ma vi deve provvedere ogni programma per conto proprio. Questo rappresenta una gravissima deficienza del Dos, che però è irrimediabile.

Su questa base, i comandi tendono a richiedere l'indicazione di un argomento che rappresenta il nome di uno o più file prima delle opzioni eventuali.

Ma c'è un altro problema. Il punto che divide in due i nomi dei file e delle directory è un muro insuperabile per i caratteri jolly.

I simboli che si possono utilizzare sono solo l'asterisco e il punto interrogativo. L'asterisco vale per una sequenza qualunque di caratteri, escluso il punto; il punto interrogativo vale per un carattere qualunque.[1]

Tabella u179.6. Alcuni esempi.

Modello	Corrispondenza
`*.*`	Corrisponde a un nome qualunque.
`*.COM`	Un nome che termina con l'estensione '`.COM`'.
`CIAO.X?X`	Tutti i nomi che iniziano per '`CIAO`' e hanno un'estensione composta da un lettera «X» iniziale e finale, senza specificare cosa ci sia al secondo posto.
`*`	Tutti i nomi che non hanno estensione (che non contengono il punto).

«

Esiste un'altra variabile di ambiente fondamentale per il Dos. Si tratta di '**PROMPT**', che consente di modificare l'aspetto dell'invito dell'interprete dei comandi. La cosa funziona un po' come nelle shell Unix, per cui si assegna una stringa che può contenere dei simboli speciali, praticamente delle sequenze di escape che vengono espanse prima della visualizzazione. La tabella u179.7 riepiloga questi simboli particolari. In origine, il Dos mostrava in modo predefinito un invito simile all'esempio seguente, in cui appare solo l'unità di memorizzazione corrente:

```
C:>
```

Questo tipo di impostazione corrisponderebbe alla stringa '**NG**'. In seguito, si è passati a un invito simile al prossimo esempio, in cui si aggiunge anche l'informazione della directory corrente:

```
C:\BIN\>
```

Questo corrisponde alla stringa '**PG**'.

Tabella u179.7. Sequenze di escape per definire dei componenti speciali all'interno di una stringa di invito.

Simbolo	Corrispondenza
`$Q`	=
`$$`	$
`$T`	Ora corrente.
`$D`	Data corrente.

Simbolo	Corrispondenza
`$V`	Numero della versione.
`$N`	Lettera dell'unità corrente.
`$G`	>
`$L`	<
`$B`	|
`$H`	<*BS*> (cancella il carattere precedente)
`$E`	<*ESC*> ($1B_{16}$)
`$_`	Codice di interruzione di riga.

Cancellando il contenuto della variabile di ambiente '**`PROMPT`**' si ripristina la stringa di invito predefinita.

Comandi interni principali

I comandi interni sono quelli che non corrispondono a programmi di servizio veri e propri, ma sono funzionalità svolte direttamente dall'interprete dei comandi. Nelle sezioni seguenti ne vengono descritti brevemente alcuni.

«

```
CH [percorso]
```

```
CHDIR [percorso]
```

‘**CH**’, o ‘**CHDIR**’, è un comando interno dell’interprete dei comandi, che consente di visualizzare o di cambiare la directory corrente. È indifferente l’uso di ‘**CD**’ o di ‘**CHDIR**’; se il comando non è seguito dal percorso, si ottiene solo la visualizzazione della directory corrente. Si osservi che se si indica un percorso assoluto di unità di memorizzazione, se questa non corrisponde a quella attuale, si cambia la directory corrente di quella unità.

Segue la descrizione di alcuni esempi.

- `C:\>`**CD** [*Invio*]

  Visualizza la directory corrente.

- `C:\>`**CD \TMP\LAVORO** [*Invio*]

  Sposta la directory corrente in ‘`\TMP\LAVORO\`’.

- `C:\TMP\LAVORO>`**CD DATI\LETTERE** [*Invio*]

  Sposta la directory corrente in ‘`DATI\LETTERE\`’ che a sua volta discende dalla posizione iniziale precedente.

- `C:\TMP\LAVORO\DATI\LETTERE>`**CD ..** [*Invio*]

  Sposta la directory corrente nella posizione della directory genitrice di quella iniziale.

- `C:\TMP\LAVORO\DATI>`**`CD F:\TMP`** [*Invio*]

  Cambia la directory corrente dell'unità '`F:`', senza intervenire nell'unità corrente.

X:

«

`{A|B|…|Z}:`

Il Dos gestisce le unità di memorizzazione in modo speciale. Per cambiare l'unità di memorizzazione corrente, non esiste un comando analogo a '**`CD`**': si deve indicare il nome dell'unità a cui si vuole accedere.

Segue la descrizione di alcuni esempi.

- `C:\>`**`A:`** [*Invio*]

  Cambia l'unità di memorizzazione attuale, facendola diventare '`A:`'.

- `A:\>`**`F:`** [*Invio*]

  Cambia l'unità di memorizzazione attuale, facendola diventare '`F:`'.

MD, MKDIR

«

`MD` ***directory***

`MKDIR` ***directory***

‘**MD**’, o ‘**MKDIR**’, è un comando interno dell’interprete dei comandi, che consente di creare una directory vuota.

Segue la descrizione di alcuni esempi.

- `C:\>`**`MD LAVORO`** [*Invio*]

  Crea la directory ‘`LAVORO\`’ a partire da quella corrente.

- `C:\>`**`MD \TMP\DATA`** [*Invio*]

  Crea la directory ‘`\TMP\DATA\`’ nell’unità corrente.

- `C:\>`**`MD F:\TMP\DATA`** [*Invio*]

  Crea la directory ‘`\TMP\DATA\`’ nell’unità ‘`F:`’.

RD, RMDIR

«

```
RM directory
```

```
RMDIR directory
```

‘**RD**’, o ‘**RMDIR**’, è un comando interno dell’interprete dei comandi, che consente di cancellare una directory vuota.

Segue la descrizione di alcuni esempi.

- `C:\>`**`RD LAVORO`** [*Invio*]

  Cancella la directory ‘`LAVORO\`’ a partire da quella corrente.

- `C:\>`**`RD \TMP\DATA`** [*Invio*]

  Cancella la directory ‘`\TMP\DATA\`’ nell’unità corrente.

- `C:\>`**`RD F:\TMP\DATA`** [*Invio*]

  Cancella la directory '`\TMP\DATA\`' nell'unità '`F:`'.

DIR

«

```
DIR [directory|file] [/P] [/W]
```

'**`DIR`**' è un comando interno dell'interprete dei comandi, che consente di visualizzare l'elenco del contenuto di una directory o l'elenco di un gruppo di file. L'argomento del comando può essere composto utilizzando caratteri jolly, secondo lo standard del Dos, ovvero i simboli '**`*`**' e '**`?`**'.

Tabella u179.8. Alcune opzioni.

Opzione	Descrizione
`/P`	Blocca lo scorrimento dell'elenco in attesa della pressione di un tasto quando questo è più lungo del numero di righe che possono apparire sullo schermo.
`/W`	Visualizza solo i nomi dei file e delle directory, senza altre informazioni, permettendo così di vedere più nomi assieme in un'unica schermata.

Segue la descrizione di alcuni esempi.

- `C:\>`**`DIR *.*`** [*Invio*]

  Visualizza l'elenco di tutti i file contenuti nella directory corrente.

- `C:\>`**`DIR ESEMPIO.*`** [*Invio*]

Visualizza l'elenco di tutti i file il cui nome inizia per '`ESEMPIO`' e continua con un'estensione qualunque.

- `C:\>`**`DIR *.DOC`** [*Invio*]

  Visualizza l'elenco di tutti i file il cui nome termina con l'estensione '`.DOC`'.

- `C:\>`**`DIR F:\DOC\*.*`** [*Invio*]

  Visualizza l'elenco di tutti i file contenuti nella directory '`\DOC\`' dell'unità '`F:`'.

- `C:\>`**`DIR F:`** [*Invio*]

  Visualizza l'elenco di tutti i file contenuti nella directory corrente dell'unità '`F:`'.

COPY

«

`COPY` ***file_origine*** [***file_destinazione***] [***opzioni***]

`COPY` ***file_1*** `+` ***file_2*** [`+` ...] [***file_destinazione***] [***opzioni***]

'**`COPY`**' è un comando interno dell'interprete dei comandi, che consente di copiare uno o più file (sono escluse le directory). Anche qui è consentito l'uso di caratteri jolly, ma al contrario dei sistemi Unix, i caratteri jolly possono essere usati anche nella destinazione. Il '**`COPY`**' del Dos consente anche di unire assieme più file.

Tabella u179.9. Alcune opzioni.

Opzione	Descrizione
`/V`	Fa in modo che venga verificato il risultato della copia.
`/B`	a sì che la copia avvenga in modo «binario». Questa opzione può servire quando si copia un file su un dispositivo e si vuole evitare che alcuni codici vengano interpretati in modo speciale.
`/Y`	Non chiede conferma prima di sovrascrivere i file, se questi esistono già nella destinazione.

Segue la descrizione di alcuni esempi.

- `C:\>`**`COPY ESEMPIO PROVA`**[*Invio*]

  Copia il file '`ESEMPIO`' nella directory corrente ottenendo il file '`PROVA`', sempre nella directory corrente.

- `C:\>`**`COPY C:\DOS\*.* C:\TMP`**[*Invio*]

  Copia tutto il contenuto della directory '`\DOS\`' dell'unità '`C:`' nella directory '`\TMP\`' nella stessa unità '`C:`', mantenendo gli stessi nomi.

- `C:\>`**`COPY TESTA+CORPO+CODA LETTERA`**[*Invio*]

  Copia, unendoli, i file '`TESTA`', '`CORPO`' e '`CODA`', ottenendo il file '`LETTERA`'.

- `C:\>`**`COPY *.DOC *.TXT`**[*Invio*]

Copia tutti i file che nella directory corrente hanno un nome che termina con l'estensione '`.DOC`', generando altrettanti file, con lo stesso prefisso, ma con l'estensione '`.TXT`'.

- `C:\>`**`COPY PROVA.PRN PRN: /B`** [*Invio*]

  Copia il file '`PROVA.PRN`' nel dispositivo '`PRN:`', ovvero sulla stampante, assicurandosi che la copia avvenga senza alterare alcunché.

DEL, ERASE

«

```
DEL file
```

```
ERASE file
```

'**`DEL`**', o '**`ERASE`**', è un comando interno dell'interprete dei comandi, che consente di cancellare uno o più file (sono escluse le directory). È da considerare che i file che hanno l'attributo di sola lettura attivo, non possono essere modificati e nemmeno cancellati.

Segue la descrizione di alcuni esempi.

- `C:\TMP>`**`DEL *.*`** [*Invio*]

  Cancella tutti i file nella directory corrente.

- `C:\TMP>`**`DEL ESEMPIO.*`** [*Invio*]

  Cancella tutti i file contenuti nella directory corrente, il cui nome inizia per '`ESEMPIO`' e termina con qualunque estensione.

- `C:\TMP>`**`DEL *.BAK`** [*Invio*]

Cancella tutti i file contenuti nella directory corrente, il cui nome termina con l'estensione '`.BAK`'.

REN, RENAME

```
REN file_origine  nome_nuovo
```

```
RENAME file_origine  nome_nuovo
```

'**REN**', o '**RENAME**', è un comando interno dell'interprete dei comandi, che consente di cambiare il nome di uno o più file (sono escluse le directory). Il primo argomento può essere un percorso relativo o assoluto, completo anche dell'indicazione dell'unità, mentre il secondo argomento è il nuovo nome, che implicitamente non può essere collocato altrove.

Segue la descrizione di alcuni esempi.

- `C:\>`**`REN ESEMPIO PROVA`** [*Invio*]

  Cambia il nome del file '`ESEMPIO`', che si trova nella directory corrente, in '`PROVA`'.

- `C:\>`**`REN *.TXT *.DOC`** [*Invio*]

  Cambia il nome di tutti i file che, nella directory corrente, hanno l'estensione '`.TXT`', trasformandoli in modo tale da avere un'estensione '`.DOC`'.

«

```
SET [variabile_di_ambiente=stringa]
```

‘**SET**’ è un comando interno dell’interprete dei comandi che ha lo scopo di assegnare un valore a una variabile di ambiente, oppure di leggere lo stato di tutte le variabili di ambiente esistenti. Quando si assegna un valore a una variabile, questa viene creata simultaneamente; quando non si assegna nulla a una variabile, la si elimina.

Segue la descrizione di alcuni esempi.

- `C:\>`**SET** [*Invio*]

  Elenca le variabili di ambiente esistenti assieme al loro valore.

- `C:\>`**SET PROMPT=PG$G** [*Invio*]

  Assegna alla variabile di ambiente ‘**PROMPT**’ la stringa ‘**PG$G**’. Questo si traduce nella modifica dell’aspetto dell’invito dell’interprete dei comandi.

- `C:\>`**SET PATH=.;C:\BIN;D:\BIN** [*Invio*]

  Assegna alla variabile di ambiente ‘**PATH**’ la stringa ‘**.;C:\BIN;D:\BIN**’.

- `C:\>`**SET PROMPT=** [*Invio*]

  Elimina la variabile di ambiente ‘**PROMPT**’, assegnandole la stringa nulla.

```
TYPE file
```

‘**TYPE**’ è un comando interno dell’interprete dei comandi, che consente di leggere ed emettere il contenuto di un file attraverso lo standard output. Questo si traduce in pratica nella visualizzazione del file in questione.

Segue la descrizione di alcuni esempi.

- `C:\>`**`TYPE LETTERA`** [*Invio*]

  Emette il contenuto del file ‘`LETTERA`’ che si trova nella directory e nell’unità corrente.

- `C:\>`**`TYPE C:\DOC\MANUALE`** [*Invio*]

  Emette il contenuto del file ‘`MANUALE`’ che si trova nella directory ‘`\DOC\`’ dell’unità ‘`C:`’.

Flussi standard

Il Dos ha ereditato da Unix anche i concetti legati ai flussi standard. In pratica, i programmi hanno a disposizione tre flussi predefiniti: uno in lettura rappresentato dallo standard input, due in scrittura rappresentati dallo standard output e dallo standard error. Il meccanismo è lo stesso di Unix, anche se non funziona altrettanto bene; infatti, non è possibile ridirigere lo standard error attraverso l’interprete dei comandi.

Secondo la tradizione delle shell Unix, la ridirezione dello standard output si ottiene con il simbolo ‘**>**’ posto alla fine del comando inte-

ressato, seguito poi dal nome del file che si vuole generare in questo modo. Per esempio,

`C:\>`**`TYPE LETTERA > PRN:`** [*Invio*]

invece di visualizzare il contenuto del file '`LETTERA`', lo invia al dispositivo di stampa corrispondente al nome '`PRN:`'; inoltre,

`C:\>`**`DIR *.* > ELENCO`** [*Invio*]

invece di visualizzare l'elenco dei file che si trovano nella directory corrente, crea il file '`ELENCO`' con questi dati.

La ridirezione dello standard output fatta in questo modo, va a cancellare completamente il contenuto del file di destinazione, se questo esiste già; al contrario, si può utilizzare anche '**>>**', con il quale, il file di destinazione viene creato se non esiste, oppure viene solo esteso.

Lo standard input viene ridiretto utilizzando il simbolo '**<**', con il quale è possibile inviare un file a un comando utilizzando il flusso dello standard input.

Alcuni comandi hanno la caratteristica di utilizzare esclusivamente i flussi standard. Si parla in questi casi di programmi filtro. Il programma di servizio tipico che si comporta in questo modo è '**SORT**', il quale riceve un file di testo dallo standard input e lo riordina restituendolo attraverso lo standard output. Si osservi l'esempio seguente:

`C:\>`**`SORT < ELENCO > ORDINATO`** [*Invio*]

In questo modo, '**SORT**' riceve dallo standard input il file '`ELENCO`' e genera attraverso la ridirezione dello standard output il file

‘ORDINATO’.

Per mettere in contatto lo standard output di un comando con lo standard input del successivo, si utilizza il simbolo ‘**|**’. L’esempio seguente mostra un modo alternativo di ottenere l’ordinamento di un file:

C:\>**TYPE ELENCO | SORT > ORDINATO** [*Invio*]

In generale, tutti i comandi che generano un risultato visuale che scorre sullo schermo, utilizzano semplicemente lo standard output, che può essere ridiretto in questo modo. Si osservi ancora l’esempio seguente che riordina il risultato del comando ‘**DIR**’, mostrandolo comunque sullo schermo:

C:\>**DIR *.DOC | SORT** [*Invio*]

Nelle sezioni seguenti vengono mostrati alcuni comandi filtro.

SORT

«

```
SORT [opzioni] < file_da_ordinare > file_ordinato
```

Il comando ‘**SORT**’, che dovrebbe corrispondere a un programma di servizio vero e proprio, riordina il file di testo che ottiene dallo standard input, generando un risultato che emette attraverso lo standard output.

Tabella u179.10. Alcune opzioni.

Opzione	Descrizione
/R	Riordina in modo decrescente

Opzione	Descrizione
/+*n_colonna*	Riordina in base al testo che inizia a partire dalla colonna indicata come argomento (si tratta di un numero a partire da uno, per indicare la prima colonna).

L'esempio seguente emette l'elenco della directory corrente riordinato in base all'estensione, che è un'informazione collocata a partire dalla decima colonna:

`C:\>`**`DIR *.DOC | SORT /+10`** [*Invio*]

MORE

«

```
MORE < file_da_leggere
```

```
MORE file_da_leggere
```

Il comando '**MORE**' legge un file, fornito come argomento o attraverso lo standard input, mostrandolo poi sullo schermo una pagina dopo l'altra. In questo modo, è possibile leggere il contenuto dei file più lunghi delle righe a disposizione sullo schermo.

Per passare alla pagina successiva, basta premere un tasto qualunque, oppure ciò che viene indicato espressamente.

Segue la descrizione di alcuni esempi.

- `C:\>`**`DIR | MORE`** [*Invio*]

  Permette di controllare lo scorrimento a video del risultato del comando '**DIR**'.

- `C:\>`**`MORE LETTERA.TXT`** [*Invio*]

  Permette di controllare lo scorrimento a video del contenuto del file '`LETTERA.TXT`'.

- `C:\>`**`TYPE LETTERA.TXT | MORE`** [*Invio*]

  Si ottiene lo stesso risultato dell'esempio precedente, attraverso l'uso di un condotto.

Accesso diretto ai dispositivi

Il Dos offre poche occasioni per accedere direttamente ai dispositivi. Si tratta generalmente solo della console e della porta parallela. L'esempio seguente mostra come «copiare» un file sul dispositivo di stampa, per ottenere così la sua stampa diretta:

`C:\>`**`COPY LETTERA PRN:`** [*Invio*]

La stessa cosa avrebbe potuto essere ottenuta con la ridirezione dei flussi standard:

`C:\>`**`TYPE LETTERA > PRN:`** [*Invio*]

Può essere interessante la possibilità di copiare il flusso di ingresso della console in un file:

`C:\>`**`COPY CON: LETTERA`** [*Invio*]

In questo caso, l'inserimento nel file '`LETTERA`' prosegue fino a quando viene ricevuto un codice EOF, che si ottiene qui con la combinazione di tasti [*Ctrl z*] seguita da [*Invio*].

È bene ricordare che la console, ovvero il dispositivo '`CON:`', riceve dati in ingresso attraverso la tastiera ed emette dati in uscita utilizzando lo schermo. In pratica, quando un programma attende dati

dallo standard input non ridiretto, li riceve dalla console, cioè dalla tastiera; nello stesso modo, quando un programma emette dati attraverso lo standard output non ridiretto, li invia alla console, cioè sullo schermo.

Riferimenti

«

- *FreeDOS*
  http://www.freedos.org
- *OpenDOS Unofficial Home Page*
  http://www.deltasoft.com/opendos.htm

[1] Ci sono programmi di origine Unix, portati in Dos, che non hanno questa limitazione riferita al punto che separa l'estensione.