Consiglio Nazionale delle Ricerche

Manuale di NSCRIPT

R. Medves

100

CNUCE

Divisione Servizio Elaborazione Dati

A cura di: Riccardo Medves

N S C R I P T

a cura di
Riccardo Medves

100 .
CNUCE

Questo manuale e' stato redatto sulla base di appunti preparati per vari corsi di Script effettuati al CNUCE e del manuale originale "NSCRIPT Reference Guide" della Universita' di Waterloo (Canada), che descrive in dettaglio le caratteristiche e le modalita' di uso di questa nuova versione di Script.
Il presente manuale costituisce una edizione riveduta e corretta del vecchio manuale CNUCE-30: SCRIPT e descrive l'uso dello Script esistente attualmente sul 370/168 sotto CMS. Logicamente si puo' suddividere in tre parti.

Una prima parte (Cap. 1-2-3-4) descrive molto schematicamente la struttura di un calcolatore; i concetti presentati non sono evidentemente esaurienti, in quanto presupporrebbero la presenza di un istruttore e di un dialogo docente-discente per il chiarimento di molti aspetti del problema, ma servono semplicemente a fornire il significato di alcune parole che ricorrono spesso nella terminologia dei calcolatori.

Una seconda parte (Cap. 5-6-7) concerne la creazione e l'aggiornamento di files CMS. Tale parte e' rivolta a chi non ha mai usato un terminale e quindi non sa come comportarsi di fronte ad esso. Chi ha gia' lavorato in CMS (anche se non in Script), potra' saltare le prime due parti, mentre per chi non ha mai lavorato in CMS ritengo essenziale in questa fase l'uso pratico di un terminale in parallelo alla lettura del manuale.

Nella terza parte infine (Cap. 8-9) si descrivono i comandi NSCRIPT propriamente detti e il loro uso.

INDICE

1) POSSIBILITA' E CARATTERISTICHE DEL LINGUAGGIO DI COMANDI SCRIPT.

Per prima cosa cerchiamo di capire a che cosa serve lo SCRIPT.
Supponiamo di dover battere con una macchina da scrivere un testo qualsiasi: ogni tasto premuto provoca sul foglio di carta l'immediata stampa di un carattere, giusto o sbagliato che sia.
Quando saremo arrivati in fondo al testo da battere, oltre al tempo perso per correggere gli errori di battitura man mano che si verificavano, dovremo controllare di non aver saltato parole o righe intere. Guai a dover inserire una frase o a dover fare una correzione che comporti un accorciamento o un allungamento del testo per piu' di qualche parola. Per non parlare poi della fatica che si deve fare per rispettare la struttura tipografica prestabilita: non si sa mai quando andare a capo, se lasciare la riga che stiamo scrivendo piu' corta delle altre o se inserire un'altra parola, e cosi' via.
Lo SCRIPT, usato in connessione con un calcolatore, fa si' che del testo battuto vengano fatte due copie: una direttamente sul foglio di carta del terminale a cui si lavora, e questa corrisponde alla stampa su carta che si ottiene con una macchina da scrivere normale, l'altra registrata su un disco magnetico del calcolatore.
Da questa copia del nostro testo registrata su disco possiamo poi ottenere tante stampe su carta quante vogliamo.
Se poi la copia su disco non ci soddisfa possiamo correggerla, allungarla o accorciarla in modo semplice, ottenendo cosi' stampe corrette: pensera' il calcolatore a sostituire o aggiungere le frasi che noi gli indichiamo, variando automaticamente la lunghezza dell'intero testo.
Non solo, ma inserendo dei comandi particolari si possono ottenere dei vantaggi tipografici non indifferenti: quello dell'allineamento automatico delle righe sulla destra, per esempio.
Le righe battute a terminale e inviate su disco possono essere di lunghezza varia, ma le righe che compaiono in stampa hanno tutte lunghezza costante: cio' e' ottenuto dallo SCRIPT in fase di stampa spostando le ultime parole a destra di una riga troppo lunga nella riga sottostante, o completando una riga troppo corta con le prime parole a sinistra della riga sottostante.
Inoltre in fase di stampa si possono far eseguire dei salti pagina, numerazioni progressive delle pagine, spostamenti rispetto al margine sinistro di stampa di interi capoversi, inserimento automatico di linee bianche, ecc.
L'insieme dei comandi per ottenere tutto cio' costituisce lo SCRIPT.

2) STRUTTURA GENERALE DI UN CALCOLATORE.

unita' di ingresso/uscita: permettono di colloquiare col calcolatore, di inserire i dati che formano il nostro programma, di ottenere la stampa dei risultati desiderati (lettori, perforatori, stampatrici, terminali ecc.)

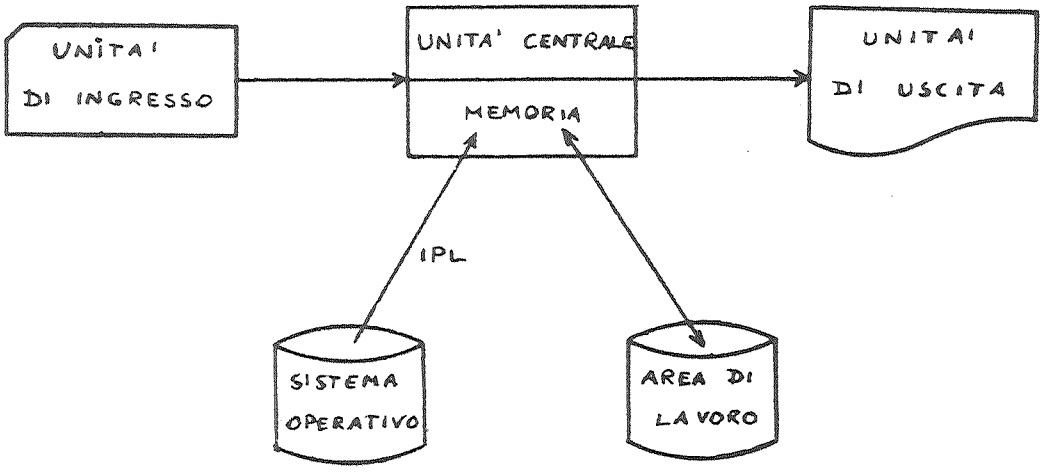
memoria: magazzino in cui vengono tenuti i dati che il calcolatore deve elaborare.

unita' centrale: parte di controllo e gestione di tutte le attivita' del calcolatore (lettura, scrittura, esecuzione dei calcoli, ecc.)

programma: insieme di istruzioni scritte in un particolare tipo di linguaggio, che indicano al calcolatore cosa deve fare, quali tipi di calcoli deve eseguire ecc. (tipi di linguaggi: FORTRAN, ASSEMBLER, PL/1 ....).

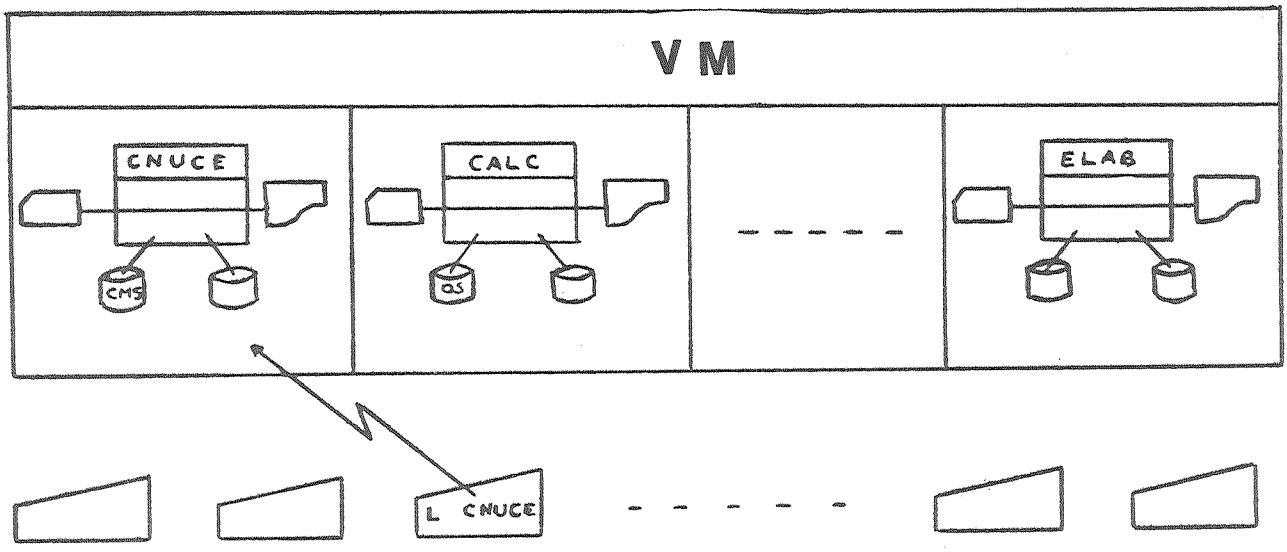
sistema operativo: e' un particolare programma, mantenuto su un determinato supporto esterno (disco, nastro...), che viene caricato in memoria all'inizio del lavoro e che gestisce tutti i programmi presentati degli utenti, fornendo loro le risorse richieste, passando il controllo dall'uno all'altro ecc.. (tipi di sistemi operativi: IBSYS, CMS, OS, APL...).

I.P.L.: (Initial Program Loading) e' la fase in cui il sistema operativo viene caricato dal supporto che lo contiene (disco, nastro, ecc.) in memoria.



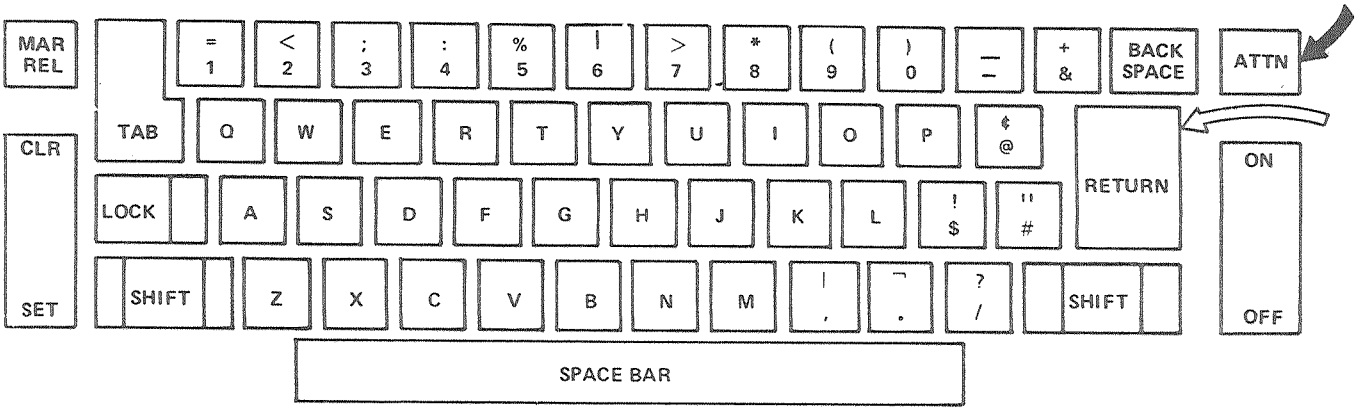
3) STRUTTURA DEL CALCOLATORE 370/168.

Il 370/168 e' un calcolatore un po' particolare: infatti invece di avere in memoria un unico sistema operativo che gestisce i lavori degli utenti, possiede un programma di controllo il CP/VM (Control Program), che ha la caratteristica di poter simulare in memoria vari calcolatori, chiamati Macchine Virtuali e contraddistinti ciascuno da un nome.
Ogni macchina virtuale si comporta come un calcolatore a se' stante, con una sua struttura, cioe' una sua configurazione di unita' e puo' essere gestito da uno qualsiasi dei sistemi operativi sopra descritti (OS, CMS, APL, ecc)



4) COLLEGAMENTO AD UNA MACCHINA VIRTUALE.

Il programmatore comunica al sistema cio' che vuole fare per mezzo di un terminale, costituito da una telescrivente collegata col calcolatore, la cui tastiera e' la seguente:



Da uno qualsiasi di tali terminali, il programmatore puo' collegarsi ad una macchina virtuale, cioe' ad un suo proprio calcolatore (identificato da un nome e da una parola chiave) e successivamente caricare il proprio sistema operativo: le elaborazioni che il programmatore fa eseguire sono del tutto indipendenti da quelle degli altri utenti che lavorano contemporaneamente sulle altre macchine virtuali. Questo tipo di accesso viene chiamato in linguaggio tecnico "time-sharing", in quanto il tempo di macchina reale disponibile viene suddiviso fra i programmatori presenti in modo che l'esecuzione dei programmi passi a brevi intervalli dall'uno all'altro.
Il programmatore che vuole collegarsi alla propria macchina virtuale, deve compiere le seguenti operazioni:

a) Accendere il terminale (tasto a destra in CN).

b) Premere il tasto ATTN per mettersi in comunicazione con il VM.
   Sul foglio comparira' una scritta del tipo:

   VM-370 ONLINE 1jh359 qsyosu

c) Scrivere LOGON (abbreviato L) seguito dal nome della sua macchina virtuale.
   Premere il tasto RETURN per inviare il messaggio al calcclatore.
   Sul foglio comparira' la scritta:

   ENTER PASSWORD:

   Se ora si preme una seconda volta il tasto RETURN, sul foglio comparira' una seguenza di caratteri del tipo:

   ****

   Questa ultima riga e' ottenuta dalla scrittura di tre

linee di otto caratteri, l'una sull'altra, e serve a coprire la parola chiave in modo che nessun altro, non autorizzato, possa accedere a tale macchina virtuale.

d) Scrivere la parola chiave, che viene cosi' a sovrapporsi alla riga scritta dal terminale, e premere il tasto RETURN.
Sul foglio comparira' la scritta:

```
LOGON AT ora SET giorno data LINE indirizzo
CMS  versione  livello  (data)
```

A questo punto il programmatore ha a disposizione un proprio calcolatore virtuale, sul quale e' stato automaticamente caricato il sistema operativo CMS che gli permettera' di gestire i suoi programmi e di lavorare in linguaggio SCRIPT.
Se ci si dovesse trovare nella necessita' di caricare personalmente il sistema operativo CMS sulla propria macchina, si dovrebbe dare il comando:

```
IPL CMS
(I)
```

e premere il tasto di RETURN.
La risposta del calcolatore sarebbe sempre:

```
CMS  versione  livello  (data)
```

e) Alla fine della sessione di lavoro, per interrompere il collegamento, si deve inviare il comando:

```
LOGOFF (abbreviato LOG)
```

Il calcolatore rispondera' con alcune informazioni riguardanti il tempo di connessione e interrompera' il collegamento.

f) Spengere il terminale (tasto a destra in CFF).

Tutti i comandi che si danno a terminale possono essere indifferentemente scritti con lettere maiuscole o minuscole; in questo manuale pero', per maggiore chiarezza, i comandi verranno sempre scritti con lettere maiuscole.

La forma di tutti i comandi e' quella di un "codice operativo" (es. LOGON, IPL ecc) seguito da uno o piu' "parametri" (es. nome della macchina virtuale, CMS ecc): codice operativo e parametri devono essere tra loro separati da uno o piu' spazi bianchi.

5) STRUTTURA DEI DATI CMS SU DISCO.

Un qualsiasi insieme di dati fornito al calcolatore e messo su un disco di lavoro si chiama "file".
Per distinguerlo dagli altri files che gia' sono su quel disco, ciascun file CMS e' identificato da tre etichette: un "nome", un "tipo" e un "modo".
In particolare per i file SCRIPT, si dovra' codificare:

nome: una qualsiasi sequenza di caratteri alfanumerici (da 1 a 8).
tipo: la parola SCRIPT.
modo: la parola A1 che indica il disco su cui si trova il file e che, nella maggior parte dei casi sara' omessa, lasciando al sistema operativo la ricerca.

Il CMS fornisce vari comandi che permettono di elaborare files memorizzati su disco: esistono per esempio comandi per la creazione, la correzione e la stampa di files.
Per ottenere l'esecuzione di un comando CMS si deve scrivere a terminale il codice operativo del comando, seguito dal nome, tipo ed eventuale modo di quel file che si vuol trattare.

Comandi (abbreviazioni) ed esempi:

```
LISTFILE  CASA  SCRIPT (DATE
(L)
```

con tale comando si effettua la ricerca, su disco, di un file che ha nome CASA e tipo SCRIPT; se il file esiste, vengono stampati come risposta il nome, il tipo, il modo del file, il formato, la lunghezza e il numero dei suoi records logici, la dimensione (in records di 800 bytes) e la data e l'ora dell'ultimo aggiornamento.
Se il file non esiste viene data la risposta
FILE NOT FOUND.

```
LISTFILE  *  SCRIPT  (DATE
(L)
```

si ottiene la ricerca di tutti i files che hanno tipo SCRIPT, qualunque sia il loro nome: per ciascuno di essi vengono stampate le informazioni come descritto nel precedente comando.

```
LISTFILE * * (DATE    oppure semplicemente    LISTFILE (DATE
(L)                                           (L)
```

si ottiene la ricerca di tutti i files esistenti su disco e

la stampa per ciascuno di essi delle informazioni descritte in precedenza.

Per tutti i comandi sopra descritti e' possibile ottenere una lista abbreviata contenente solo le indicazioni su nome, tipo e modo semplicemente omettendo l'opzione (DATE

```
TYPE  CASA  SCRIPT  3  9
(T)
```

si ottiene la stampa a terminale delle righe dalla 3 alla 9 del file indicato (nel nostro caso il file CASA SCRIPT).
Se i due numeri sono omessi, si ottiene la stampa a terminale di tutto il file, mentre se e' omesso solo il secondo si intende raggiungere la fine del file a partire dalla linea indicata.
Se, una volta inviato il comando di stampa, si desidera interrompere momentaneamente la scrittura del file, basta premere il tasto ATTN una volta.
Se si vuol riprendere la stampa, basta premere il tasto RETURN; altrimenti, se si vuol interrompere definitivamente la stampa, basta dare il comando HT prima di premere il tasto RETURN.

```
PRINT  CASA  SCRIPT
(PR)
```

si ottiene la stampa del file indicato sulla stampatrice centrale del calcolatore. Se la stampatrice possiede solo i caratteri maiuscoli, si deve far seguire il comando dalla opzione (UP) per la conversione dei caratteri minuscoli dello Script in caratteri maiuscoli.

```
ERASE CASA SCRIPT
```

si ottiene la cancellazione dal disco di lavoro del file indicato, rendendo cosi' disponibile l'area su disco precedentemente occupata.

```
COPYFILE nome1 SCRIPT A1 nome2 SCRIPT A1 (FROM 5 FOR 3
(COPY)
```

serve per creare un nuovo file (nome2 SCRIPT) prelevando tre linee a partire dalla quinta compresa del file nome1 SCRIPT. Se il secondo parametro e' omesso, si intende raggiungere la fine del file. Se il file nome2 esiste gia', per poterlo modificare occorre codificare i parametri REPLACE o APPEND: il parametro REPLACE (abbreviato REP) permette la sostituzione completa del file, con la perdita di quello precedente;

il parametro APPEND (abbreviato AP) permette di aggiungere il nuovo file alla fine di quello esistente.

COPYFILE nome1 SCRIPT A1 nome2 SCRIPT A1...nomeN SCRIPT A1
(COPY)

serve per creare un nuovo file (nomeN SCRIPT) unendo insieme gli altri files che precedono (da nome1 a nomeN-1) Se il file nomeN esiste gia', valgono le stesse considerazioni fatte per le opzioni REP e AP dell'esempio precedente.

RENAME nome1 SCRIPT A1 nome2 SCRIPT A1
(R)

serve per cambiare il nome "nome1" di un file esistente su disco in "nome2".

Nota: dopo l'invio di un comando al calcolatore (invio che si ottiene sempre premendo il tasto RETURN) si attenda che il comando sia eseguito ed accettato.
Il calcolatore rispondera' con

R; tempo di elaborazione

oppure

R(errore); tempo di elaborazione

a seconda che il comando sia stato eseguito correttamente o meno.

6) CREAZIONE DI UN FILE SCRIPT.

Per ottenere la registrazione su disco di un file con un certo nome e tipo, ci si deve servire di un "Editore", cioe' di un sistema appositamente creato per la gestione (memorizzazione, modifica, aggiornamento, ecc.) di un dato file.
Attualmente sul 370/168 sotto VM sono presenti due Editori, richiamabili rispettivamente con i comandi

EDIT nome tipo (E) e TED nome tipo

L'uso dell'uno o dell'altro Editore e' simile, sia nelle funzioni fornite, sia nei comandi da utilizzare.
Pertanto scriveremo ad esempio:

E CASA SCRIPT oppure TED CASA SCRIPT

e si premera' il tasto RETURN per inviare il comando.
Sul foglio compare la scritta:

NEW FILE.
EDIT:

e si predispone ad accettare ulteriori comandi. Se il file e' nuovo, cioe' non esisteva prima su disco e deve essere creato, si deve ora dare il comando

INPUT

e premere il tasto RETURN.
Il calcolatore risponde allora con

INPUT:

e si predispone ad accettare il file in ingresso.
A questo punto l'utente puo' battere a terminale il testo del suo file.
Non ha importanza la forma con cui le frasi vengono scritte a terminale, in quanto, come gia' abbiamo accennato, una caratteristica dello SCRIPT e' quella di fornire una stampa allineata sulla destra: e' pero' consigliabile tenere le righe abbastanza corte, affinche' durante la fase di correzione, la stampa a terminale di una riga, non porti via un tempo eccessivamente lungo.
Dopo ogni riga premere il tasto RETURN per inviarla al calcolatore.
Le righe che vengono in tal modo inviate al calcolatore, non vanno direttamente su disco, ma restano momentaneamente nella sua memoria.

Puo' accadere che per un improvviso inconveniente (guasto di macchina o interruzione di corrente o altro) il calcolatore si blocchi o, come si dice, il sistema operativo vada in "RESTART".
In tal caso tutto cio' che si trova in quel momento in memoria viene perduto, mentre cio' che e' registrato su disco rimane inalterato.
Allora, per evitare di lavorare a vuoto, e' buona norma, di tanto in tanto, salvare su disco il lavoro fatto in modo che il file si conservi anche in caso di restart.
In tal modo, se si verifica un restart, non sara' piu' necessario riscrivere il file dall'inizio, ma solo quella porzione del file che e' stata battuta dopo l'ultimo salvataggio.
Per salvare su disco il file che si trova in memoria si procede nel modo seguente: dopo avere inviato una linea di testo al calcolatore premendo il tasto RETURN, si deve, senza battere alcun carattere sulla linea, premere ancora una volta il tasto RETURN.
Il calcolatore risponde con:

EDIT:

Si deve allora dare il comando per memorizzare temporaneamente il file su disco, e cioe'

SAVE

Il sistema risponde con:

EDIT:

Per continuare a scrivere il resto del file dal punto in cui avevamc provveduto al salvataggio si deve dare di nuovo il comando INPUT.
Tale procedura, per i motivi su accennati, e' bene sia ripetuta di tanto in tanto.
Alla fine della sessione di lavoro vogliamo memorizzare definitivamente il file su disco, per poterlo ritrovare al successivo collegamento.
Per questo, dopo l'ultima riga inviata premendo RETURN, si prema ancora il tasto RETURN.
Il calcolatore risponde con

EDIT:

Diamo il comando:

FILE

Il calcolatore memorizza definitivamente il file su disco e risponde con

R; tempo elaborazione

Nota1: durante la stesura del testo, cioe' in fase di INPUT, oppure mentre scriviamo un comando CMS, puo' accadere di battere un tasto sbagliato, o anche di sbagliare una parola o tutta una riga. Se ce ne accorgiamo subito, possiamo porvi rimedio facendo uso dei due caratteri di controllo @ e ¢, descritti in dettaglio in Appendice A.

@: permette di cancellare dalla riga corrente l'ultimo carattere scritto (se ripetuto n volte di seguito cancella gli ultimi n caratteri scritti).

¢: permette di cancellare tutta la riga corrente scritta fino a quel momento.

Nota2: oltre al testo che vogliamo scrivere e che viene trattato successivamente dallo SCRIPT possiamo far uso di altri comandi particolari che permettono di gestire il testo secondo determinate richieste, ad esempio saltare delle righe bianche, andare a pagina nuova, vietare lo spostamento delle parole da una riga all' altra ecc. Tali comandi vengono inseriti nel testo nei punti in cui essi devono agire: iniziano sempre con un punto a colonna 1, e vengono inviati al calcolatore premendo il tasto RETURN come per una qualsiasi riga normale. E' chiaro che qualsiasi linea inizi con un punto a colonna 1 viene interpretata come comando SCRIPT: nel testo non devono pertanto comparire linee normali che inizino con un punto in colonna 1. I comandi dello SCRIPT saranno descritti in dettaglio in seguito.

Esempio:

```
                                        <----- (ATTN)
VM-370 online    1jh359 qsyosu
                                        <----- (ATTN)
l cnuce
ENTER PASSWORD:
                                        <----- (RETURN)
********
LOGON AT 20:56:00 SET FRIDAY 01/09/76 LINE 54F
CMS V 2.0 PLC 21 (8 GEN 76)

e casa script
R; T=0.01/0.01 20:56:35
NEW FILE.
EDIT:
input
INPUT:
Questa e' una prova di
creazione di un file SCRIPT.
.comando SCRIPT
                                        <----- (2 RETURN)
EDIT:
save
EDIT:
input
INPUT:
Questo fy@ile non
esisteva prima su disco.
Ogni volt¢Ogni riga del file
.comando SCRIPT
viene inviata al calcolatore
premendo il tost@@@asto RETURN.
                                        <----- (2 RETURN)
EDIT:
file
R; T=0.01/0.01 20:57:39
```

```
l casa script (date
FILENAME FILETYPE FM FORMAT  RECS BLOCKS  DATE    TIME
CASA     SCRIPT   A1 V      14    9       1    1/09/76 20:57
R; T=0.01/0.02 20:58:07

l casa script
CASA  SCRIPT  A1
R; T=0.01/0.01 20:58:15

t casa script 6 8

Questo file non
esisteva prima su disco.
Ogni riga del file

R; T=0.01/0.03 20:58:32

erase casa script
R; T=0.01/0.01 20:59:18

t casa script
FILE 'CASA SCRIPT A' not FOUND
R(00028); T=0.01/0.01 20:59:30

log
CONNECT= 00:03:59  VIRTCPU= 000:00.12  TOTCPU= 000:00.40
LOGOFF AT 20:59:59 SET FRIDAY 01/09/76
```

7) AGGIORNAMENTO DI UN FILE ESISTENTE.

Supponiamo di aver gia' scritto il file CASA SCRIPT, di averlo memorizzato su disco e di volerlo ora modificare, cioe' di voler aggiungere o togliere righe o di voler correggere errori commessi in fase di scrittura.
Per prima cosa e' necessario copiare in memoria dal disco il file desiderato, mediante uno dei comandi:

EDIT nome tipo (E) o TED nome tipo

nel nostro caso:

EDIT CASA SCRIPT o TED CASA SCRIPT

Sul foglio compare ora la scritta:

EDIT:

in quanto il file esiste gia' su disco, non deve essere creato di nuovo, ma solo corretto.
A questo punto il calcolatore e' pronto a ricevere qualsiasi comando EDIT per l'aggiornamento del file stesso.
Associato al file richiamato c'e' un indicatore di linea, che all'inizio e' posizionato in testa al file; per poter fare una modifica su una certa riga del file, si deve spostare l'indicatore in corrispondenza all'inizio di tale riga.

7a) I comandi per esequire spostamenti dell'indicatore di linea all'interno del file sono:

TOP

posiziona l'indicatore all'inizio del file, prima della prima riga in modo da permettere l'eventuale inserzione di altre righe in testa al file.

BOTTOM

posiziona l'indicatore sull'ultima riga del file.

UP n

posiziona l'indicatore all'n-ma riga precedente quella corrente. Se n e' omesso viene assunto uguale a 1 ed avviene il posizionamento sulla riga immediatamente precedente quella corrente.

NEXT n

posiziona l'indicatore sull'n-ma riga seguente quella corrente. Se n e' omesso, viene assunto uguale a 1 ed avviene il posizionamento sulla riga immediatamente seguente quella corrente.

LOCATE /stringa/ o semplicemente /stringa/

ricerca la sequenza di caratteri "stringa" nelle righe seguenti quella corrente a partire da quella immediatamente seguente fino alla fine del file.
La ricerca si arresta quando tale sequenza viene trovata per la prima volta, e l'indicatore viene posizionato su tale riga.

FIND ␣␣----␣ stringa

dove ␣␣----␣ e' una sequenza di n caratteri bianchi, ricerca nelle righe seguenti a quella corrente fino alla fine del file, la sequenza di caratteri "stringa" posizionata a partire dall'(n+1)-esimo carattere della riga. Cioe' ogni carattere non bianco della sequenza viene confrontato col carattere che si trova nella colonna corrispondente della linea. La ricerca si arresta quando tale sequenza viene trovata per la prima volta e l'indicatore viene posizionato alla riga trovata.

7b) I comandi che servono ad elaborare le linee di un file sono i seguenti:

CHANGE /stringa1/stringa2/ n

sostituisce la sequenza di caratteri "stringa1" con "stringa2" nella linea corrente e nelle (n-1) linee successive.
Se n e' omesso e' assunto uguale a 1 ed il cambiamento interessa la sola linea corrente; se n e' posto uguale ad * si intende raggiungere la fine del file.
Se le linee interessate contengono piu' di una sequenza "stringa1" solo la prima di esse (da sinistra) viene cambiata. Le due stringhe possono essere di lunghezza diversa. Dopo l'esecuzione di tale comando si resta posizionati sull'ultima riga corretta.

```
Esempi :        AMARE
c /m/r/       : ARARE
c /r/lt/      : ALTARE
c /lta/tto/   : ATTORE
```

```
c /tto//      : ARE          nota 1
c /r/var/     : AVARE
c /va/lte/    : ALTERE
c /e/o/       : ALTORE       nota 2
c /o/e/       : ALTERE
c /re/ro/     : ALTERC
c /alt/io /   : IO ERO       nota 3
```

nota1: la stringa di lunghezza nulla (due barre consecutive) permette di eliminare una sequenza dalla riga corrente.
nota2: per effettuare un cambiamento occorre fornire delle stringhe di lunghezza sufficiente ad individuare univocamente il punto da modificare.
nota3: si noti che lo spazio bianco e' un carattere normale, come ogni altro all'interno della stringa.

CHANGE /stringa1/stringa2/ n *

sostituisce tutte le sequenze di caratteri uguali a "stringa1" con "stringa2" nella linea corrente e nelle altre (n-1) successive. Se n e' posto uguale ad * si intende raggiungere la fine del file. (n non puo' essere omesso e deve essere esplicitamente scritto uguale a 1 nel caso si voglia modificare la sola linea corrente).

REPLACE linea

sostituisce la linea scritta a quella corrente. Dopo l'esecuzione l'indicatore resta posizionato sulla stessa riga.

DELETE n

cancella n linee a partire da quella corrente (compresa). Dopo l'esecuzione l'indicatore risulta posizionato alla prima riga non cancellata.
Se n e' omesso e' assunto uguale a 1; avviene allora la cancellazione della sola linea corrente e il posizionamento sulla successiva.

OVERLAY ␣␣----␣ stringa

ciascun carattere non bianco della stringa va a sostituire il carattere che si trova nella colonna corrispondente della linea corrente.
Il carattere "_" (sottolineatura) permette di sostituire il corrispondente carattere della stringa con uno spazio bianco.
(␣ indica uno spazio bianco)

INPUT linea

inserisce la linea scritta a lato dopo la linea corrente, prima della successiva e posiziona l'indicatore sulla linea inserita.

INPUT

non seguito da alcun carattere, fa si' che il calcolatore entri in fase di INPUT. Si possono allora battere quante righe si vogliono e tutto questo nuovo testo viene inserito fra la linea corrente e la successiva. Quando si vuol tornare in fase di correzione basta premere due volte il tasto RETURN e attendere la risposta EDIT: per continuare normalmente. L'indicatore risulta posizionato sull'ultima riga inserita.

7c) Comandi vari:

=

permette la ripetizione dell'ultimo comando effettuato, senza la necessita' di riscriverlo esplicitamente.

X comando e Y comando

salvano il comando indicato per un successivo utilizzo.

X n e Y n

eseguono n volte il comando salvato in precedenza senza la necessita' di riscriverlo esplicitamente.
Se n e' omesso e' assunto uguale a 1.
I due comandi compiono cioe' la stessa funzione del comando = ma senza la limitazione di riferirsi solo all'ultimo comando dato.

TYPE n

stampa a terminale la riga corrente e le (n-1) successive.
Se n e' omesso e' assunto uguale a 1, e si ha la stampa della sola riga corrente.

ZONE n1 n2

indica che il comando che segue dovra' operare solo tra le colonne n1 e n2 (n1 minore di n2) delle linee costituenti il file (mentre per default qualsiasi comando inizia sempre dalla colonna 1 della linea e termina, per i files SCRIPT, con la colonna 132).

Nota1: nelle descrizioni precedenti, i due caratteri / che racchiudono la sequenza di caratteri cercata, possono essere due qualsiasi caratteri uguali che non compaiano nella stringa.

Nota2: i comandi precedenti provocano la scrittura, a terminale, dell'intera linea ricercata o modificata, permettendo cosi' di verificare se tutto si e' svolto regolarmente ed eventualmente di tornare a correggere. Si puo' sopprimere la scrittura di tale linea facendo seguire il comando dal carattere punto (.), senza spazi bianchi intermedi. Esempio:
N. /stringa/
verra' ricercata la sequenza di caratteri "stringa"; se trovata , la tastiera si sblocchera' in attesa di nuovi comandi, senza che la linea in cui compare la sequenza venga stampata a terminale.

Valgono, durante la fase di correzione di un file (EDIT) le stesse regole di salvataggio descritte nel capitolo 6.
L'unica differenza e' che durante la fase di correzione ci troviamo gia' in EDIT: pertanto sara' sufficente dare di tanto in tanto il comando SAVE senza dover prima premere il tasto RETURN.
Il sistema memorizzera' su disco le modifiche fatte e rispondera' con EDIT:
Alla fine e' sufficiente scrivere FILE perche' il file sia memorizzato permanemtemente su disco.

I comandi fin qui descritti sono comuni ai due Editori presenti e variano dall'uno all'altro solo le abbreviazioni con le quali tali comandi sono richiamati:

Comando	EDIT	TED
BOTTOM	B	BC
CHANGE	C	C
DELETE n	DEL	D
FILE	FILE	FIL
FIND	F	F
INPUT	I	I
LOCATE	L	L
NEXT n	N	N
OVERLAY	O	O
REPLACE	R	R
SAVE	SAVE	SA
TOP	TOP	TO
TYPE	T	T
UP n	U	U
X	X	X
Y	Y	Y
ZONE	Z	Z
=	=	=

7d) Inoltre il TED possiede, oltre a quelli finora enumerati, in piu' rispetto all'Editore standard i seguenti comandi:

AGAIN n
(A)

permette la ripetizione n volte dell'ultimo comando effettuato, senza la necessita' di riscriverlo esplicitamente. Se n e' omesso e' assunto uguale a 1.

GOTO n
(G)

permette il trasferimento dalla linea corrente alla n-ma linea del file.

LINENO
(LI)

fornisce il numero della linea corrente a partire dall'inizio del file.

FUP //----/ stringa
(FUP)

agisce come il comando FIND ma si muove all'indietro, cioe' verso l'inizio del file.

NEXT /stringa/
(N)

ricerca la sequenza di caratteri "stringa" nelle righe seguenti quella corrente a partire da quella immediatamente seguente fino alla fine del file.
La ricerca si arresta quando tale sequenza viene trovata per la prima volta, e l'indicatore viene posizionato su tale riga.

UP /stringa/
(U)

ricerca la sequenza di caratteri "stringa" nelle righe precedenti quella corrente, a partire da quella immediatamente precedente, fino all'inizio del file. La ricerca si arresta quando tale sequenza viene trovata per la prima volta, e l'indicatore viene posizionato su tale riga.

DELETE /stringa/
(D)

cancella le linee del file a partire da quella corrente fino a quella che contiene la sequenza "stringa" (esclusa).

```
PUT n nome SCRIPT
(PU)
```

crea un nuovo file identificato da "nome" e tipo SCRIPT costituito dalla riga sulla quale siamo posizionati e dalle (n-1) righe successive. Se il file "nome" esiste gia', le linee vengono aggiunte alla fine del file.

```
PUT /stringa/ nome SCRIPT
(PU)
```

crea un nuovo file identificato da "nome" e tipo SCRIPT, costituito dalla riga sulla quale siamo posizionati e dalle seguenti fino a quella (esclusa) che contiene la sequenza di caratteri "stringa". Se la stringa non e' trovata il comando non ha effetto. Se il file "nome" esiste gia', le linee vengono aggiunte alla fine del file.

```
PUT n      oppure      PUT /stringa/
(PU)                   (PU)
```

crea un file temporaneo, come descritto sopra, e inseribile in altra posizione del file corrente tramite il comando GET. Il file resta su disco fino all'esecuzione di un nuovo comando PUT dello stesso tipo.

```
GET nome tipo A1 n m
(GE)
```

preleva m righe a partire dalla riga n-ma del file definito con "nome" e "tipo", le inserisce dopo la linea corrente, prima della successiva e si posiziona sull'ultima riga inserita. Se n e m sono omessi, viene prelevato l'intero file.

```
GET
(GE)
```

preleva il file temporaneo creato in precedenza col comando PUT n o PUT /stringa/.

8) COMANDI SCRIPT.

Come abbiamo gia' accennato, un file SCRIPT e' costituito da righe di due tipi:
i) righe che cominciano con un punto a colonna 1
ii) righe che non cominciano con un punto a colonna 1
Le prime contengono i comandi SCRIPT ed eventuali operandi, come piu' avanti descritto; le seconde contengono il testo che dovra' poi essere stampato.

Durante la stampa il testo contenuto nel file subisce delle trasformazioni tipografiche, allo scopo di rendere tutte le righe della stessa lunghezza: cio' e' ottenuto spostando le parole da una riga all'altra. Cosi' ad esempio se una riga e' troppo lunga le sue parole finali andranno nella riga successiva, mentre se una riga e' troppo corta, essa viene completata con le prime parole della riga successiva.
Questo procedimento di formattamento puo' essere interrotto in corrispondenza alle righe che contengono comandi SCRIPT: in tal caso l'ultima riga del testo prima del comando SCRIPT puo' rimanere piu' corta delle altre. Quando cio' accade si dice che si e' verificata un'interruzione del formattamento nelle righe fra le quali il comando e' inserito. Poiche' non tutti i comandi provocano l'interruzione, nella descrizione di ciascun comando verra' detto specificatamente se la provoca oppure no.

L'interruzione fra due righe di stampa si puo' anche causare senza l'uso di comandi, semplicemente iniziando la seconda con almeno uno spazio bianco: in tal caso la prima non viene completata con parole della seconda, mentre la seconda, compresi gli spazi vuoti iniziali comincia a capo riga.
Il formattamento, oltre a spostare le parole da una riga all'altra, provoca anche l'inserimento di spazi bianchi fra le parole in modo da raggiungere esattamente la lunghezza prevista per le righe di stampa. Come vedremo vi sono comandi con i quali e' possibile impedire che cio' avvenga.
Si consiglia, nella prima fase dell'apprendimento, di limitarsi allo studio dei soli comandi seguenti:

.SP, .PA, .FO, .NF, .BR, .CE, .IN, .UN

che, oltre ad essere quelli di uso piu' frequente, sono sufficienti, nella fase iniziale, per usufruire in maniera soddisfacente delle possibilita' offerte dallo SCRIPT.

8a) Comandi per la descrizione del formato della pagina di stampa:

.LL n (line length)

- definisce la lunghezza in caratteri di una linea di stampa ($n \leq 132$)
- se l'operando e' omesso viene assunto .LL 60
- se il comando e' omesso viene assunto .LL 60
- crea una interruzione nelle linee tra le quali e' inserito
- ha effetto dalla linea corrente in poi.

.PL n (page length)

- definisce la lunghezza in linee di una pagina di stampa
- se l'operando e' omesso viene assunto .PL 66
- se il comando e' omesso viene assunto .PL 66
- crea una interruzione nelle linee tra le quali e' inserito
- ha effetto dalla pagina corrente in poi.

.TM n (top margin)
.BM n (bottom margin)

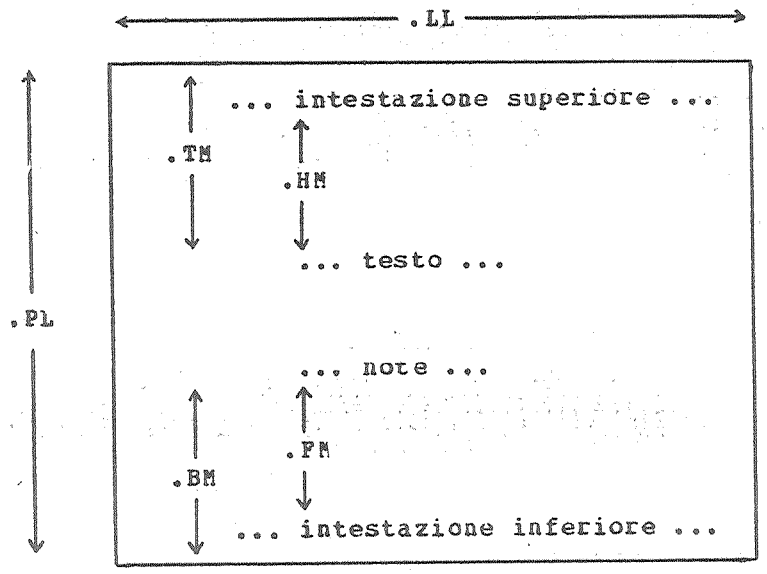
- definiscono (rispettivamente) la distanza in linee che deve esistere tra l'inizio (fine) della pagina fisica di stampa e la prima (ultima) riga del testo. I valori forniti devono, rispettivamente, essere maggiori dei valori assunti per gli heading e i footing margin (comandi .HM e .FM)
- se gli operandi sono omessi viene assunto .TM 5 e .BM 3
- se i comandi sono omessi viene assunto .TM 5 e .BM 3
- creano una interruzione nelle linee tra le quali sono inseriti
- .TM ha effetto dalla pagina successiva in poi (Eccezione: si applica anche alla pagina corrente se in tale pagina non e' ancora stato prodotto nessun output di testo)
  .BM ha effetto dalla pagina corrente in poi.

.HM n (heading margin)
.FM n (footing margin)

- definiscono (rispettivamente) la distanza in linee che deve esistere tra l'inizio (fine) della pagina linea di stampa e la riga di intestazione superiore (inferiore)
- se gli operandi sono omessi viene assunto .HM 1 e .FM 1

- se i comandi sono omessi viene assunto .HM 1 e .FM 1
- creano una interruzione nelle linee tra le quali sono inseriti
- .HM ha effetto dalla pagina successiva in poi
- .FM ha effetto dalla pagina corrente in poi.

Esempi relativi al paragrafo 8a) :



8b) Comandi per la descrizione delle intestazioni.

```
.HE 'S1'S2'S3'              (heading)
    oppure   .TT /S1/S2/S3/        (top title)
.FE 'S1'S2'S3'              (footing)
    oppure   .BT /S1/S2/S3/        (bottom title)
```

- consentono (rispettivamente) l'inserimento di una intestazione all'inizio (fine) delle pagine in uscita.
  S1, S2 e S3 sono delle stringhe di caratteri ciascuna lunga al massimo quanto la lunghezza definita per la linea in uscita (le tre sequenze si possono sovrapporre l'un l'altra) e vengono stampate in uscita allineando S1 a sinistra, centrando S2 e allineando S3 a destra sulla riga.
  Se uno o piu' campi sono omessi, si deve sempre

codificare esplicitamente i delimitatori. Esempio:
.HE 'S1'''
Le sequenze operandi dei comandi .HE e .FE non devono contenere il carattere (apice) che e' obbligatorio quale delimitatore; il delimitatore per le sequenze operandi del comando .TT e .ET invece puo' essere sostituito da un qualsiasi altro carattere alfanumerico o speciale, consentendo l'inserimento di qualsiasi carattere all'interno delle stringhe.
I comandi restano in effetto finche' non compare un nuovo comando che si sostituisce al primo.

- gli operandi non possono essere omessi: i delimitatori devono sempre essere presenti. Esempio: .HE ''''
- se i comandi sono omessi viene assunto .HE '''PAGE %' (.TT '''PAGE %') e .FE '''' (.ET '''') dove il segno % viene di volta in volta automaticamente sostituito dalla numerazione della pagina corrente.
- non creano una interruzione nelle linee tra le quali sono inseriti
- .HE (.TT) ha effetto dalla pagina successiva in poi (Eccezione: si applica anche alla pagina corrente se in tale pagina non e' ancora stato prodotto nessun output di testo, eccetto la prima pagina di un' uscita OFFLINE nel caso che .TM e .HM siano tali che l'intestazione debba essere contenuta nelle prime 3 linee)
.FE (.BT) ha effetto dalla pagina corrente in poi.

```
.HD 'S1'S2'S3'              (odd heading)
    oppure   .OT /S1/S2/S3/        (odd top title)
.FD 'S1'S2'S3'              (odd footing)
    oppure   .BO /S1/S2/S3/        (odd bottom title)
```

- valgono le stesse considerazioni fatte per i comandi .HE (.TT) e .FE (.BT) con la differenza che questi si applicano alle sole pagine la cui numerazione e' dispari (pagine 1, 3, 5 ecc.), cioe' le sequenze indicate vengono scritte in testa (in fondo) alle sole pagine con numerazione dispari, non influenzando in alcun modo le sequenze eventualmente presenti per le pagine con numerazione pari.

```
.HV 'S1'S2'S3'              (even heading)
    oppure   .ET /S1/S2/S3/        (even top title)
.FV 'S1'S2'S3'              (even footing)
    oppure   .EB /S1/S2/S3/        (even bottom title)
```

- valgono le stesse considerazioni fatte per i comandi .HE (.TT) e .FE (.BT) con la differenza che questi si

applicano alle sole pagine la cui numerazione e' pari (pagine 2, 4, 6 ecc), cioe' le sequenze indicate vengono scritte in testa (fondo) alle sole pagine con numerazione pari, non influenzando in alcun modo le sequenze eventualmente presenti per le pagine con numerazione dispari.

Esempi relativi al paragrafo 8b) :

```
.HE 'Stringa di prova'''
```

Causa la stampa della frase
  Stringa di prova
in alto a sinistra in ogni pagina, ma elimina la stringa di destra (che conteneva PAGE %) e pertanto elimina la stampa della numerazione delle pagine.

```
.HE 'Stringa di prova ''PAGE %'
```

Ha lo stesso effetto del comando precedente per quanto riguarda la stampa della frase
  Stringa di prova
in alto a sinistra in ogni pagina, ma a destra mantiene la numerazione delle pagine nella forma
  PAGE 1 , PAGE 2 , ecc.

```
.HE ''- % -''
```

Sostituisce la forma di numerazione delle pagine da
  PAGE 1 , PAGE 2 , ecc.
stampata in alto a destra, alla forma
  - 1 - , - 2 - , ecc.
stampata in alto al centro della linea

```
.FE '''Stringa di prova'
```

Causa la stampa della frase
  Stringa di prova
in basso a destra in ogni pagina.
Non influenza la stringa in testa alla pagina, che resta invariata e uguale il valore che era stato impostato in precedenza (es. PAGE % se non era stato definito niente esplicitamente)

Nota: come si vedra' in seguito, per lo NSCRIPT il segno ";" (punto e virgola) ha il particolare significato di separatore di linee logiche: cioe' nella stessa linea fisica in ingresso possono essere fornite piu' linee logiche, separate dal carattere ; .
Pertanto tale carattere non puo' comparire in una stringa di intestazione se prima non si dissocia da esso il significato di fine linea logica.
Per esempio il comando

```
.HE 'Stringa di prova; capitolo primo'''
```

verrebbe logicamente interpretato dallo NSCRIPT come due linee separate:

```
.HE 'Stringa di prova
capitolo primo'''
```

e pertanto si avrebbe una segnalazione di errore nella forma del comando.
La sequenza necessaria e' allora (vedi il rispettivo comando):

```
.C# ?
.HE 'Stringa di prova; capitolo primo'''
.C# :
```

dove il comando C# ridefinisce il carattere ? quale separatore logico prima della frase di intestazione contenente il ; (e non contenente il ?), e ripristina poi il valore default.

8c) Comandi per il controllo della spaziatura e del salto pagina.

.SP n (space)

- causa l'inserimento di n linee bianche nel punto in cui si trova
- se l'operando e' omesso viene assunto .SP 1
- crea una interruzione nelle linee tra le quali e' inserito.

.DS (double space)

- fa stampare il testo in doppia spaziatura e raddoppia l'effetto di successivi comandi .SP n
- crea una interruzione nelle linee tra le quali e' inserito.

.SS (single space)

- annulla l'effetto del comando precedente o di un comando .LS
- crea una interruzione nelle linee tra le quali e' inserito.

.LS { n / YES / NO } (line spacing)

- fa stampare il testo spaziandolo automaticamente di n linee bianche tra una riga e l'altra e moltiplica per n l'effetto di successivi comandi .sp m
  .LS YES corrisponde a .LS 1 cioe' .DS
  .LS NO corrisponde a .LS 0 cioe' .SS

- crea una interruzione nelle linee tra le quali e' inserito
- se l'operando viene omesso viene assunto .LS 0 . cioe' .SS

.LE { YES / NO } (heading blank line)

- consente la stampa o la soppressione di linee bianche in inizio pagina
  .LE YES consente l'inserimento di linee bianche all'inizio della pagina
  .LE NO ignora, in uscita, le eventuali linee bianche in inizio pagina
  Eccezione: linee bianche richieste all'inizio della prima pagina di uscita vengono sempre e comunque stampate
- non crea una interruzione nelle linee tra le quali e' inserito
- se l'operando e' omesso viene assunto .LE YES
- se il comando e' omesso viene assunto .IE NC

.PA { n / +n / -n } (page eject)

- produce un salto pagina, cioe' la linea che segue il comando e' stampata all'inizio di una nuova pagina, rinumerando le pagine successive a partire dal numero indicato dall'operando (pagina n, pagina corrente+n, pagina corrente-n)
  Eventuali note vengono stampate prima del salto pagina, cioe' nella pagina corrente
- se l'operando e' omesso viene assunto .PA +1, cioe' la numerazione delle pagine viene incrementata ogni volta di una unita'
- crea una interruzione nelle linee tra le quali e' inserito.

.PD (odd page force)
oppure .OP (odd page eject)
.PV (even page force)
oppure .EP (even page eject)

- causano (rispettivamente) il salto alla prima pagina dispari (pari) successiva alla corrente, e da li' continuano a stampare il testo
- creano una interruzione nelle linee tra le quali sono inseriti.

.CP n (conditional page)

- causa un salto pagina se mancano meno di n linee alla fine della pagina (o all'inizio delle eventuali note presenti).
- se l'operando e' omesso o non positivo, il comando viene ignorato.

.EM { YES / NO } (empty page)

- consente la stampa o la soppressione di pagine bianche, cioe' nelle quali sono presenti solo eventuali intestazioni, ma non testo o note
  .EM YES consente la stampa di pagine bianche
  .EM NO non stampa, in uscita le eventuali pagine bianche, pur continuando ad incrementare regolarmente la numerazione delle pagine
- se l'operando e' omesso viene assunto .EM YES
- se il comando e' omesso viene assunto .EM NO
- non crea una interruzione nelle linee tra le quali e' inserito.

Esempi relativi al paragrafo 8c) :

```
.CP 10
.SP 4
.DS
Capitolo 3
Descrizione dei comandi principali
.SS
Prima riga del testo e successive
```

Il comando .CP 10 provoca il salto pagina se nella pagina corrente ci sono meno di 10 righe di stampa. Lo scopo e' quello di impedire che il titolo del capitolo 3 venga scritto in fondo ad una pagina e non sia seguito da almeno due righe di testo. La riga contenente il numero di capitolo e' preceduta da 4 righe bianche e seguita da una riga bianca prima del titolo, il quale a sua volta e' separato dal testo da una riga bianca.

8d) Comandi per il controllo della numerazione delle pagine.

.PN { ON / OFF / OFFNO } (page number)

.PN { ARABIC / ROMAN } (page number)

- consentono il controllo della numerazione delle pagine
  ON produce la numerazione delle pagine, sia internamente che esternamente
  OFF sopprime la numerazione esternamente, ma continua quella interna
  OFFNO sopprime la numerazione sia esternamente che internamente
  Nota: la soppressione effettiva della stampa della numerazione si ottiene solo eliminando il carattere di paginazione % dalle linee di intestazione
  ARABIC la numerazione viene effettuata con i numeri arabi (1, 2, 3, 4 ecc)
  ROMAN la numerazione viene effettuata con i numeri romani in caratteri minuscoli (i, ii, iii, iv, ecc)
- se gli operandi sono omessi viene assunto .PN ON e .PN ARABIC
- se i comandi sono omessi viene assunto .PN ON e .PN ARABIC
- non creano una interruzione nelle linee tra le quali sono inseriti.

.AR (arabic)

- ha lo stesso effetto del comando .PN ARABIC

.RO (roman)

- ha lo stesso effetto del comando .PN ROMAN

Esempi relativi al paragrafo 8d) :

```
.HE ''''
Tutte le pagine che seguono questo comando, sono senza
numerazione esterna. (Indice dei capitoli, numerazione
esterna, ecc.)
Il comando cioe', annullando il segno % dall'intestazione,
elimina la stampa della numerazione delle pagine (ricordiamo
che il default e' .HE ''' PAGE %' che porta ad una
numerazione del tipo PAGE 1 , PAGE 2 , ecc.

.PA 1
.PN ROMAN
.FE ''%''

Le pagine seguenti questa serie di comandi hanno una
numerazione in fondo alla pagina (Introduzione ecc).
```

Infatti l'intestazione in alto resta cancellata dal precedente comando .HE '''', mentre in basso viene inserito, centrato in mezzo alla riga, il segno % che sara' sostituito, in stampa, dal numero corrente della pagina.
La numerazione viene effettuata in numeri romani (.PN ROMAN) e comincera' col numero "i" (uno) come stabilito dal comando .PA 1.
La mancanza di tale specifica non avrebbe riportato ad uno il contatore delle pagine, proseguendo la numerazione dal valore raggiunto nella precedente pagina (anche se questo non veniva stampato esternamente).

```
.PA 1
.HD '''%'
.HV '%'''
.FE ''''
.PN ARABIC
Capitolo 1°: Prova numerazione
.HD '''Prova numerazione % '
.HV ' % Prova numerazione '''
```

Le istruzioni utilizzate consentono l'annullamento della precedente numerazione in basso con i numeri romani e il ripristino di una numerazione in alto e in numeri arabi.
La numerazione comparira' allineata a sinistra o a destra nella riga a seconda che la pagina sia pari o dispari.
Inoltre le specifiche .HD '''%' e .HV '%''' hanno validita' dalla pagina corrente in quanto, al momento in cui sono emesse, non e' stato ancora scritto alcun testo in tale pagina. Il testo e' rappresentato dal titolo del capitolo e dal susseguente contenuto.
Le successive istruzioni .HD e .HV vengono invece eseguite quando gia' una parte di testo (il titolo del capitolo) e' stata scritta: la loro validita' pertanto comincera' dalla pagina seguente.
Scopo di tale gioco e' quello di avere riportato, oltre al numero della pagina, anche il titolo del capitolo (a sinistra o a destra a seconda che la pagina sia pari o dispari) in tutte le pagine seguenti ma non in quella corrente, che gia' contiene il titolo del capitolo: in questa pagina l'intestazione conterra' cioe' la sola numerazione.

8e) Comandi per il controllo del formato del testo.

.CO { YES / NO } (concatenate)

.NC { YES / NO } (no concatenate)

- le righe in uscita vengono approssimate, per difetto, alla lunghezza della linea .LL mediante lo

spostamento di parole dalla o alla linea seguente, senza pero' essere allineate alla lunghezza esatta mediante l'introduzione casuale tra le parole di spazi bianchi
- se gli operandi sono omessi viene assunto .CO YES e .NC YES
- se i comandi sono omessi viene assunto .CC YES e .NC NO
- creano una interruzione nelle linee tra le quali sono inseriti.

.JU { YES / NO } (justify)

.NJ { YES / NO } (no justify)

- le righe in uscita vengono portate esattamente alla lunghezza della linea (.LL) mediante l'aggiunta casuale tra le parole di spazi bianchi.
  Non viene pero' creato alcuno spostamento di parole tra una riga e la successiva.
  Se la linea corrente eccede la lunghezza stabilita per la linea, la riga e' stampata cosi' come e' scritta
- se gli operandi sono omessi viene assunto .JU YES e .NJ YES
- se i comandi sono omessi viene assunto .JU YES e .NJ NO
- creano una interruzione sulle linee tra le quali sono inseriti.

.FO { YES / NO } (format)

oppure .FI { YES / NO } (fill)

.NF { YES / NO } (no format)

- consentono l'effetto combinato dei comandi .CC e .JU. Le righe in uscita vengono formate mediante lo spostamento di parole dalla o alla linea seguente (.CO) e mediante l'aggiunta casuale tra le parole di spazi bianchi (.JU) in modo da avere un allineamento, a destra, alla lunghezza definita per la linea (.LL)
- se gli operandi sono omessi viene assunto .FO YES

(.FI YES) e .NF YES
- se i comandi sono omessi viene assunto .FO YES (.FI YES) e .NF NO
- creano una interruzione nelle linee tra le quali sono inseriti.

.BR (break)

- mantiene separata la linea che precede da quella che segue tale comando, impedendo lo spostamento di parole dall'una all'altra.
  La linea che precede non subisce ne' l'effetto di concatenazione (.CO) ne' quello di aggiustamento (.JU), rimanendo in genere in stampa piu' corta della lunghezza definita per la linea (.LL)
- crea una interruzione nelle linee tra le quali e' inserito.

.LA { n / YES / NO } (left adjust)

.RA { n / YES / NO } (right adjust)

oppure .RI { n / YES / NO } (right adjust)

- consentono rispettivamente di allineare a sinistra (destra) le n linee che seguono o le linee comprese tra i comandi con gli operandi YES e NO, senza che venga causato il formattamento del testo.
  L'allineamento avviene considerando anche eventuali spazi bianchi all'inizio o alla fine della riga interessata.
  Se la riga e' piu' lunga della lunghezza specificata per la linea, la riga viene troncata.
- se gli operandi sono omessi, viene assunto .LA 1 e .RA 1
- creano una interruzione nelle linee tra le quali sono inseriti.

.CE { n / YES / NO } (center)

- consente di centrare in mezzo alla linea nei margini definiti dalla lunghezza .LL e da eventuali comandi .IN le n righe seguenti o tutte le linee comprese tra i comandi .CE YES e .CE NO, senza che venga causato

il formattamento del testo.
Vengono conteggiati anche eventuali spazi bianchi all'inizio o alla fine della riga interessata.
Se la riga e' piu' lunga della lunghezza specificata per la linea, la riga viene troncata.
- se l'operando e' omesso viene assunto .CE 1
- crea una interruzione nelle linee tra le quali e' inserito.

.IN { n / +n / -n } (indent)
.UN m (undent)

- consentono di spostare le righe di un certo numero di spazi dal margine sinistro.
Il comando .IN agisce su tutte le linee che seguono fino al successivo comando .IN, stampando le righe a partire dalla colonna (n+1) della linea o modificando di +n o -n il valore precedente. (La prima colonna equivale a .IN 0)
Il comando resta in effetto anche per eventuali note, salti pagina, files inseriti col comando .IM o .AP.
E' inoltre valido anche con la specifica .NF
Il comando .UN agisce sulla sola linea che segue, spostandola a sinistra (rispetto ad un precedente valore .IN n) di m posizioni.
Si noti che il valore di m in un comando .UN non dovra' mai superare la somma dei valori n dei comandi .IN in effetto a quel punto, cioe' dovra' essere tale da non spostare l'inizio della riga prima del margine sinistro della stampa.
- se gli operandi sono omessi viene assunto .IN 0 e .UN 0
- se i comandi sono omessi viene assunto .IN 0 e .UN 0
- creano una interruzione nelle linee tra le quali sono inseriti.

.OF { n / +n / -n } (offset)

- consente di spostare le righe di un certo numero di spazi dal margine sinistro. Il comando agisce su tutte le linee seguenti eccetto la prima fino al successivo comando .OF o .IN, stampando tali righe a partire dalla colonna (n+1) della linea o modificando di +n o -n il valore precedente. (La prima colonna equivale a .OF 0)

- se l'operando e' omesso viene assunto .CF 0
- se il comando e' omesso viene assunto .OF 0
- crea una interruzione nelle linee tra le quali e' inserito.

Esempi relativi al paragrafo 8e) :

```
.NF
.IN 1
.UN 1
-ABCDE              -ABCDE
FGHIJ                FGHIJ
KLMNO                KLMNO
.UN 1               -PQRST
-PQRST               UVWXY
UVWXY
.IN
.FO
```

Si noti che il valore di m non puo' superare il valore di n dato precedentemente. Il comando

.OF n (offset)

fa si' che tutte le righe seguenti, eccetto la prima, inizino dopo n spazi bianchi dal margine sinistro. Esempio:

```
.NF
.OF 1
-ABCDE              -ABCDE
FGHIJ                FGHIJ
KLMNO                KLMNO
.OF 1               -PQRST
-PQRST               UVWXY
UVWXY
.OF
.FO
```

Ha validita' finche' non si trova un nuovo comando .OF n oppure il comando di chiusura .OF oppure un comando .IN
Si noti che se e' in azione un comando .IN m, il comando .OF n parte dalla colonna m+1.

.OF

annulla l'effetto del comando precedente.

8f) Comandi per l'inserimento di note.

.FN{BEGIN / END} (footnote)

- consente di salvare temporaneamente una o piu' linee di input e inviarle automaticamente alla fine della pagina corrente.
  Nel caso di esistenza di note la lunghezza del testo effettivo sara' accorciata e verra' generato un salto pagina per permettere la stampa della nota a fine pagina.
  Nel caso non esista spazio sufficiente la nota verra' scritta o continuata nella pagina successiva.
  Le linee salvate e che costituiscono la nota in oggetto sono quelle comprese tra i due comandi .FN BEGIN e .FN END
  Tutti i comandi di formattamento del testo presenti tra le linee salvate vengono cancellati alla fine dell'inserimento e vengono riassunti i valori precedenti.
  I comandi di formattamento presistenti si applicano anche alle linee salvate, se non esplicitamente annullati.
  All'interno di una nota viene forzato il comando .SS e vengono ignorati i comandi di salto pagina.
- non crea una interruzione nelle linee tra le quali e' inserito.

.RM n{NOSAVE / SAVE / DELETE} (remote)

- consente di salvare temporaneamente una o piu' linee di input e inserirle automaticamente in un punto specifico della pagina corrente o delle seguenti.
  n e' un numero che indica la linea della pagina, prima della quale devono essere inserite le linee interessate (n al di fuori dei campi definiti da .TM e .BM). Se n e' posto uguale a * viene assunto uguale al valore corrente di .TM +1, cosi' che le linee vengono inserite in testa alla pagina interessata.
  Se e' specificato NOSAVE, le linee salvate sono cancellate automaticamente dopo il primo uso.
  Se e' specificato SAVE le linee salvate vengono ripetute in ogni pagina seguente, finche' non viene incontrato lo stesso comando con l'opzione DELETE.
  Le linee salvate sono quelle comprese tra il primo comando .RM e il secondo .RM trovati nel testo
  Tutti i comandi di formattamento del testo presenti tra le linee salvate vengono cancellati alla fine dell'inserimento e vengono riassunti i valori precedenti.
  I comandi di formattamento preesistenti si applicano anche alle linee salvate, se non eplicitamente

annullati.
- se n e' omesso viene assunto uguale a *; se il secondo argomento e' omesso viene assunto uguale a NOSAVE; se entrambi gli argomenti sono omessi, viene assunto .RM * NOSAVE
- non crea una interruzione nelle linee tra le quali e' inserito.

Esempi relativi al paragrafo 8f) :

```
Un file e' caratterizzato da nome, tipo e modo.
.FN BEGIN
Nella maggior parte dei comandi il modo
di un file puo' essere omesso.
.FN END
Il CMS fornisce i comandi che permettono
di elaborare i files.
```

Le righe precedenti danno la seguente stampa:

```
Un file e' caratterizzato da nome, tipo e modo.
Il CMS fornisce i comandi che permettono di elaborare i
files.

----------

Nella maggior parte dei comandi il modo di un file puo'
essere omesso.
```

8g) Comandi per il collegamento tra files diversi.

.IM nome n1 n2 (imbed)

- serve per inserire nel posto desiderato un altro file "nome" SCRIPT che si trova sul disco di lavoro della propria macchina virtuale. Al momento in cui il comando .IM viene incontrato, in fase di stampa del file corrente, il controllo passa al file "nome" SCRIPT, che viene stampato di seguito al primo; alla fine il controllo ritorna al file corrente e la stampa continua dalla riga successiva al comando.
  I comandi di formattamento presenti continuano a

valere anche per il nuovo file, se non esplicitamente annullati.
I valori di n1 e n2 indicano le linee del file "nome" che devono essere inserite nel file corrente durante la stampa
- se n2 e' omesso si intende da n1 alla fine del file
- se n1 e n2 sono omessi si intende tutto il file.

.AP nome n1 n2 (append)

- ha lo stesso scopo del comando .IM, solo che si applica soltanto per aggiungere files alla fine di quello corrente, non in mezzo alle istruzioni. Al momento in cui il comando viene incontrato in fase di stampa del file corrente, il controllo passa al file "nome" SCRIPT che viene stampato di seguito al primo; alla fine la stampa si interrompe senza ritornare al file corrente.

.RD n (read terminal)

- causa la sospensione momentanea della stampa a terminale e permette l'inserimento di n linee di input, dopo di che il processo di stampa riprende regolarmente.
Se la stampa avviene sulla stampatrice off-line, in corrispondenza del comando si ha la stampa di n righe bianche.
- se l'operando e' omesso viene assunto .RD 1
- crea una interruzione nelle linee tra le quali e' inserito.

.TY linea (type on terminal)
oppure .PR linea (print)

- causa la stampa dalla linea indicata sulla console di controllo dello NSCRIPT.
Se tale console e' anche l'unita' di uscita della stampa formattata, tale comando viene ignorato, eccetto che durante il primo passo di una esecuzione in due tempi (opzione 2PASS del comando di stampa SCRIPT) quando il comando causa regolarmente la stampa della informazione voluta.
- non crea una interruzione nelle linee tra le quali e' inserito.

Esempi relativi al paragrafo 8g) :

```
Conformemente a quanto precedentemente stabilito si comunica
che:
.SP .IM LETTERA 12
Facendo poi riferimanto agli avvenimenti dell'ultima
settimana, verranno presi i seguenti provvedimenti:
.SP
.AP PRIMO
.AP SECONDO

Al momento della stampa, le ultime linee del file LETTERA, a
partire dalla dodicesima, vengono inserite nel testo
corrente, alla fine del quale vengono poi stampati
nell'ordine i files PRIMO e SECONDO.


.RA YES
Spett. Ditta
.RD 3
.SP2
Si trasmette una breve illustrazione

Al momento della stampa, dopo aver scritto sul margine
destro del foglio la dicitura Spett. Ditta, il terminale si
arresta, in attesa che vengano battute tre righe contenenti
il nome e l'indirizzo della ditta. Poi la stampa riprende
regolarmente. In questo modo si possono ottenere copie del
file (una lettera) con indirizzi diversi su ogni copia.
```

8h) Comandi per il cambiamento di simboli.

.FN SET char (footnote)

Il carattere char indicato nell'operando SET viene stampato in una linea come separatore tra il testo effettivo e la nota in oggetto.

- se l'operando SET e' omesso, viene assunto .FN SET ␣ (dove ␣ sta per spazio bianco)
- se il comando .FN SET char e' omesso, viene assunto .FN SET -
- non crea una interruzione nelle linee tra le quali e' inserito.

.CW char (control word separator)

- lo NSCRIPT permette di codificare piu' linee logiche di input sulla stessa linea fisica. Tali linee logiche sono tra loro separate da un particolare carattere delimitatore.

Il comando .CW permette di ridefinire tale delimitatore
- se l'operando e' omesso, non sono permesse piu' linee logiche in una stessa linea fisica
- se il comando e' omesso viene assunto .CW ;
- non crea una interruzione nelle linee tra le quali e' inserito.

.CR cw1 cw2 (control word replacement)

- permette di variare i comandi che identificano una certa funzione dello NSCRIPT
  cw1 e' la parola di controllo originale (2 caratteri)
  cw2 e' la nuova parola di controllo (2 caratteri)
  che sostituira' la precedente
  Se cw2 e' anch'essa una parola di controllo gia' esistente nello NSCRIPT questa perdera' il suo primitivo significato per acquistare il nuovo.
  cw1 deve sempre essere una parola originale dello NSCRIPT e' non puo' essere una parola di controllo ridefinita col comando .CR
- se cw2 e' omesso, la parola di controllo cw1 e quella (cw2) associata precedentemente ad essa sono riportate ai significati primitivi
- se entrambi gli operandi sono omessi, tutte le parole di controllo sono portate ai loro valori primitivi
- non crea una interruzione nelle linee tra le quali e' inserito.

.LI { n | char } (literal)

- permette di variare o ignorare l'identificatore dei comandi dello NSCRIPT.
  Se l'argomento e' un carattere non numerico, esso sostituisce l'identificatore dei comandi di NSCRIPT in colonna 1 della linea di input.
  Se l'argomento e' un carattere numerico esso indica il numero di linee seguenti per le quali dovra' essere ignorato l'identificatore in colonna 1 (Esso sara' cioe' trattato come un normale carattere di input e stampato di conseguenza in uscita).
- se l'operando e' omesso viene assunto .LI 1
- se il comando e' omesso viene assunto .LI .
- non crea una interruzione nelle linee tra le quali e' inserito.

.PS char (page number symbol)

- permette di variare il simbolo di numerazione delle pagine.
  La numerazione delle pagine viene evidenziata all'esterno dello NSCRIPT mediante la sostituzione del numero corrente della pagina al simbolo definito da tale comando ad ogni occorrenza di tale simbolo nelle intestazioni di inizio o fine pagina.
- se l'operando e' omesso, viene assunto .PS %
- se il comando e' omesso, viene assunto .PS %
- non crea una interruzione nelle linee tra le quali e' inserito.

.TR s1 t1 s2 t2....sn tn (translate)

- permette la conversione automatica dei caratteri sn indicati con i corrispondenti caratteri tn nella fase di uscita del documento.
  sn e tn possono avere la forma di singoli caratteri o numeri esadecimali di due cifre.
  L'opzione e' valida fino a che non viene incontrato il comando .TR senza argomenti.
  La conversione non si applica alle linee che contengono comandi Script
- non crea una interruzione nelle linee tra le quali e' inserito.

.TI s1 t1 s2 t2....sn tn (translate on input)
.TI SET char

- permette la conversione automatica nei caratteri tn, dei soli caratteri sn che sono preceduti da uno speciale carattere definito con la opzione SET.
  sn e tn possono avere la forma di singoli caratteri o numeri esadecimali di due cifre
  L'opzione e' valida fino a che non viene incontrato il comando .TI senza argomenti
- se viene omesso l'operando SET, il comando equivale a .TR
- non crea una interruzione nelle linee tra le quali e' inserito.

Esempi relativi al paragrafo 8h) :

.CW ?

Ridefinisce il carattere ? quale delimitatore delle linee logiche:

```
Questa linea ? viene interpetrata ? .br ? come 4 linee

equivale a

Questa linea
viene interpetrata
.br
come 4 linee
```

```
.CR SP SK

Il comando .SP (space) perde  il suo significato e viene
sostituito dal comando .SK (skip)
```

```
.LI 2

Permette di ignorare l'identificatore dei comandi Script
in colonna 1 nelle due linee che seguono il comando .IT,
permettendo  di  interpetrarle  come  normali  linee  da
stampare in uscita

I comandi
.SP
.LI 2
.PA
.IN

diviene in uscita

I comandi

.PA
.IN
```

8i) Comandi per l'uso di nomi di riferimento simbolici.

Lo NSCRIPT permette l'uso di nomi (variabili e costanti) di riferimento simbolici, composti da una sequenza da 1 a 10 caratteri non speciali, che possono avere un determinato valore ed essere cosi' opportunamente utilizzati come vedremo negli esempi alla fine del paragrafo.

.SR nome { 'stringa' / espressione } (set reference)

- permette di assegnare il valore risultante dal secondo operando (stringa di caratteri o espressione numerica), al nome simbolico definito dal primo operando.
  L'eventuale stringa di caratteri al secondo operando non deve superare i 14 caratteri di lunghezza ne'

avere spazi bianchi in mezzo; non e' necessario raddoppiare gli apici che compaiono all'interno della stringa.
L'eventuale espressione aritmetica e' una sequenza di operatori (+, -, *, /) e termini (numeri interi o caratteri particolari dello NSCRIPT) che viene eseguita da sinistra a destra per ottenere il risultato da assegnare al nome simbolico.
I seguenti caratteri hanno il significato particolare riportato:
% o & o il simbolo definito nel comando .PS = Numero pagina corrente dall'inizio del file
<* = Numero linea corrente dall'inizio del file
<1 = Numero linea corrente dall'inizio della pagina
- non crea una interruzione nelle linee tra le quali e' inserita.

.SE stringa con inclusi nomi simbolici

- permette di assegnare un valore (numerico o stringa di caratteri) a un nome simbolico, facendo uso di altri nomi simbolici inclusi nella stringa e preceduti dal carattere &.
La stringa operando viene cioe' ricreata sostituendo a tutti i nomi simbolici trovati preceduti dal carattere & i loro valori attuali al momento della esecuzione del comando e successivamente interpetrata come se fosse l'operando di un comando .SR
- non crea una interruzione nelle linee tra le quali e' inserita.

.SU { n / ON / OFF } (substitute symbol)

- permette di sostituire, nelle n linee indicate o nelle linee comprese tra i comandi .SU ON e .SU OFF, tutti i nomi simbolici preceduti dal carattere & con i loro valori correnti, prima dell'esecuzione delle linee medesime
- se l'operando e' omesso, viene assunto .SU 1
- se il comando e' omesso, viene assunto .SU OFF
- non crea una interruzione nelle linee tra le quali e' inserito.

.UR linea (use reference)

- permette di sostituire nella linea indicata come operando, tutti i nomi simbolici preceduti dal carattere & con i loro valori correnti, prima

dell'esecuzione della linea medesima
- non crea una interruzione nelle linee tra le quali e' inserito.

.IF { 'stringa1' op 'stringa2' / numero1 op numero2 } (conditional)

- permette di eseguire o meno la riga successiva al comando.
  Se la condizione indicata e' verificata la linea viene eseguita, altrimenti viene ignorata.
  Il confronto puo' avvenire tra due stringhe di caratteri (non contenenti spazi bianchi) nella normale sequenza EBCDIC o tra due numeri interi con segno.
  L'operazione puo' essere una delle seguenti:

=	o	EQ	:	uguale
¬=	o	NE	:	non uguale
<	o	LT	:	minore
>	o	GT	:	maggiore
<=	o	LE	:	minore o uguale
>=	o	GE	:	maggiore o uguale

- non crea una interruzione nelle linee tra le quali e' inserito.

Esempi relativi al paragrafo 8i) :

Creazione di un "Indice del Contenuto"

a) Dopo il titolo di ciascun capitolo occorre inserire un comando

.SR riferimento mnemonico %

con il quale si assegna il valore % (numero della pagina corrente al riferimento mnemonico indicato.
Esempio:

Capitolo 1° = Significato dello NSCRIPT
.SR SIGNIF %

b) Ciascuna linea dell'indice deve essere formata da comandi del tipo:

.UR titolo &riferimento mnemonico

Cioe' nel nostro caso

.UR Significato dello NSCRIPT.....&SIGNIF

Il comando .UR che precede la linea consentira' la sostituzione automatica del riferimento mnemonico SIGNIF col valore % della pagina nella quale si trova il relativo capitolo.

c) Se l'indice cosi' costituito viene posto alla fine del manuale, non vi sono altre iniziative da prendere.
Infatti nel momento in cui viene stampata la pagina contenente il Capitolo 1°, viene assegnato al riferimento SIGNIF il valore della pagina.
Alla fine del manuale, quando viene stampato l'indice, i riferimenti simbolici usati (SIGNIF ecc) hanno ciascuno il valore della pagina relativa e le linee dell'indice possono essere direttamente stampate. Se invece vogliamo che l'indice stia in testa al manuale, occorre stamparlo "in due passi" (vedi operando 2PASS nel comando di stampa SCRIPT): il primo passo serve per poter assegnare a tutti i nomi di riferimento simbolici i valori delle pagine volute, ma senza che venga effettuata la stampa su carta.
Terminata la prima fase, lo NSCRIPT ricomincera' dall'inizio, passando questa volta alla stampa effettiva: in questa seconda fase i riferimenti simbolici trovati posseggono gia' un valore e pertanto l'indice, anche in testa al manuale, viene stampato regolarmente con la numerazione delle pagine volute.

9) STAMPA DI UN FILE SCRIPT.

Abbiamo visto che un file Script si compone delle normali linee di stampa del testo e di particolari comandi (.XX) che servono a gestire in modo particolare la stampa del file.
Vi sono due modi di ottenere la stampa di un file Script.

stampa senza formattamento : si ha la stampa del file cosi' come e' stato battuto a terminale, compresi tutti i comandi Script; e' cioe' una lista pura e semplice di quanto battuto prima a terminale ed equivale al comando TYPE del CMS.

stampa con formattamento : i comandi Script non vengono listati esplicitamente, ma vengono interpretati per ottenere il formattamento della stampa.

Una volta che abbiamo memorizzato il nostro file su disco, il comando di stampa sotto CMS e' il seguente:

SCRIPT nome (opzioni)

nome: e' il nome del file che si vuole stampare.
opzioni: un insieme di parole chiave, separate tra loro da uno o piu' spazi bianchi. Esse possono essere scelte tra le seguenti:

NO (nowait)

fa partire immediatamente la stampa del file su terminale.
Se l'opzione NO e' omessa, appare invece il messaggio:
LOAD PAPER; HIT RETURN.
e l'esecuzione si interrompe fino a che non viene premuto il tasto RETURN, permettendo eventualmente di aggiustare la carta.

ST (stop)

interrompe la stampa del file su terminale alla fine di ogni pagina, permettendo ad esempio di cambiare carta. Per far ripartire la stampa con la pagina successiva, basta premere il tasto RETURN.
Se l'opzione ST e' omessa, non c'e' pausa fino a quando tutto il file non sia stato stampato.

PA nnn (page)

fa stampare il file a partire dalla pagina nnn.
Tale opzione e' attiva anche con la specifica .PN OFF, mentre non e' attiva con la specifica .PN OFFNO.
Se l'opzione e' omessa, la stampa comincia dall'inizio del file.

SI (single)

fa stampare una sola pagina selezionata eventualmente da una opzione PA nnn.
Se l'opzione SI e' omessa, la stampa prosegue fino alla fine del file.

UN (unformatted)

fa stampare il testo non formattato dallo SCRIPT.
Se l'opzione UN e' omessa, il file viene formattato in fase di stampa.

TR (translate)

esegue una conversione di tutti i caratteri da minuscoli a maiuscoli, prima di inviare il file in stampa.
Se l'opzione TR e' omessa, viene eseguita la stampa cosi' come il file e' stato creato.

OF (offline) oppure
PR (printer)

la stampa viene effettuata sulla stampante centrale del calcolatore, invece che sul proprio terminale.
Se tale stampante non e' equipaggiata con una particolare catena, comprendente ad esempio i caratteri minuscoli, si deve anche effettuare una conversione con la opzione TR.

ON (online) oppure
TERM (terminal)

la stampa viene effettuata sul proprio terminale on-line. Se TERM e' omesso viene assunto default il valore CF e la stampa viene effettuata sulla stampatrice centrale del calcolatore.

AD (adjust)

la stampa sulla stampante centrale viene centrata sulla carta spostata di 30 colonne rispetto al margine sinistro del foglio.

Se l'opzione AD e' omessa, la stampa e' allineata a sinistra.

CE n (center)

la stampa sulla stampatrice centrale viene spostata di n colonne (n compreso tra 0 e 50) rispetto al margine sinistro del foglio. Se n e' omesso viene assunto uguale a 0.

CO (continue)

permette di continuare automaticamente la stampa formattata anche dopo un errore. L'errore trovato viene evidenziato con un opportuno messaggio.

2PASS (twopass)

permette di eseguire la stampa di un file in due fasi.
In una prima fase viene effettuato tutto il dovuto formattamento, vengono inizializzati e assegnati i valori ai nomi simbolici, ma non viene prodotta nessuna uscita.
La seconda fase serve invece alla produzione fisica dell'uscita su carta o su terminale.

Esempi:

a) SCRIPT CASA (TERM

il calcolatore risponde a terminale:
LOAD PAPER; HIT RETURN.
e si ferma, dando cosi' la possibilita' di cambiare carta o di aggiustarla. Per iniziare si preme il tasto RETURN. Il calcolatore stampa il file CASA SCRIPT a terminale, con le opportune specifiche di formato, salti pagina ecc.
L'omissione del parametro TERM causa la stampa sulla stampatrice centrale del calcolatore invece che sul proprio terminale.

b) SCRIPT CASA (NO TERM

si ha la stampa come nell'esempio a), ma il calcolatore non richiede la sostituzione della carta (NO): appena premuto il tasto RETURN la stampa inizia immediatamente.

c) SCRIPT CASA (NO UN TERM

si ha a terminale la stampa non formattata del file CASA SCRIPT, cioe' una lista pura e semplice del file come e' stato precedentemente battuto a terminale.

d) SCRIPT CASA (NO PA 19 SI TERM

si ottiene a terminale la sola stampa, formattata, della pagina 19 del testo.

e) SCRIPT CASA (NO PA 19 ST TERM

si ottiene a terminale la stampa del file a partire da pagina 19 (PAGE019). Alla fine di ogni pagina il calcolatore si ferma (ST) e, per continuare, si deve premere il tasto RETURN.

f) SCRIPT CASA (TR OF

si ottiene la stampa formattata in caratteri maiuscoli (TR) del file sulla stampante centrale del calcolatore (OF).

APPENDICE A

Caratteri con significato particolare.

- Cancellazione di un carattere : @
  il segno @ scritto subito dopo un carattere di una linea, cancella tale carattere. Si usa generalmente per cancellare un carattere errato in fase di battitura. Se il segno @ e' ripetuto n volte di seguito, cancella gli ultimi n caratteri scritti. Logicamente e' come se tale comando ci riportasse indietro nella linea e ci permettesse di sovrapporre ai vecchi i nuovi caratteri battuti. Si tenga presente che lo spazio bianco (tasto SPACE) e' considerato dallo SCRIPT un "carattere" al pari degli altri: se si vogliono percio' eliminare degli spazi bianchi sovrabbondanti, si deve sempre fare uso del carattere @ e mai del tasto BACKSPACE, che come vedremo serve per altri scopi. Esempi:

```
SV@CRIPT        =  SCRIPT    (cancella la lettera V)
SVRI@@@CRIPT    =  SCRIPT    (cancella le lettere VRI)
SC  @@RIPT      =  SCRIPT    (cancella i due bianchi)
```

- Cancellazione di una linea : ¢
  il carattere ¢ permette di cancellare tutta la linea corrente scritta (prima di premere il tasto RETURN) e di riscrivere di seguito quella corretta. Esempio:

```
SVRI ¢SCRIPT    =  SCRIPT
```

  logicamente e' come se tale comando ci riportasse all'inizio della linea, cancellando quanto precedentemente scritto.

- Backspace logico : non esiste in maniera standard
Viene definito mediante il comando CMS:

```
SET INPUT char 16
```

dove char e' il carattere che si vuol utilizzare. Esempio:

```
SET INPUT < 16
```

permette di tornare indietro logicamente in una linea di tanti caratteri quanti sono i segni < scritti. A questo punto i nuovi caratteri non bianchi battuti di seguito vanno a sovrapporsi ai vecchi, mentre gli spazi bianchi fanno restare invariato il vecchio carattere della linea. Esempi:

```
ABCD<<H      =  ABHD
ABCD<<HP     =  ABHP
ABCD<<H P    =  ABHDP
```

Si noti la differenza col carattere di cancellazione @ che, quando ripetuto n volte, perde completamente il ricordo degli n caratteri precedenti, anche se successivamente vengono battuti caratteri bianchi. Esempi:

```
ABCD@@H      =  ABH
ABCD@@H P    =  ABH P
```

- Tasto di backspace fisico : BACKSPACE
durante la codifica di un testo in SCRIPT puo' essere utile sovrapporre due o piu' caratteri per ottenere dei segni particolari (ad esempio il segno ∄ indicante lo spazio bianco e' ottenuto dalla sovrapposizione dei due caratteri b e /) o, come nella maggior parte dei casi, per ottenere delle parole sottolineate. Ci si serve in tal caso del tasto BACKSPACE,il cui funzionamento e' il seguente: una volta battuta la parola da sottolineare (ad esempio) si prema il tasto BACKSPACE tante volte quanto necessario per riportare fisicamente la pallina di stampa sotto la prima lettera della parola e si batta il carattere di sottolineatura "_" tante volte quante necessarie; poi si riprenda la codifica normale.
C'e' solo da notare che mentre i caratteri @, ∄ e < non vengono inviati al calcolatore, in quanto agiscono subito sulla linea corrente, ed e' la linea gia' corretta che viene inviata al calcolatore (RETURN), il tasto BACKSPACE causa invece l'inserimento nella linea di certi caratteri particolari, che verranno poi interpretati in fase di stampa dallo SCRIPT. Esempio:

La parola SCRIPT battuta a terminale come:
SCRIPT(BACK)(BACK)(BACK)(BACK)(BACK)(BACK)_____
viene inviata al calcolatore come:
S(BS)_C(BS)_R(BS)_I(BS)_P(BS)_T(BS)_
dove con (BACK) abbiamo indicato l'uso del tasto BACKSPACE e con (BS) abbiamo indicato quel particolare carattere che lo SCRIPT interpretera' in seguito per ricreare la sottolineatura.
Pertanto e' da tener presente che le parole sottolineate occupano nel file il triplo della loro lunghezza.

- Carattere di fine linea logica : #
permette di scrivere su una linea fisica del terminale piu' linee logiche contenenti comandi CMS, separate tra loro dal segno #. Esempio:

```
LIST CASA SCRIPT # TYPE CASA SCRIPT          (Return)
```

equivale a

```
LIST  CASA SCRIPT     (Return)
TYPE  CASA SCRIPT     (Return)
```

- Carattere di escape: "

permette di poter stampare i caratteri con significato particolare descritti finora, senza che essi vengano interpretati e assumano quelle particolari funzioni ad essi regolarmente associate: per poter ottenere cio' basta far precedere il carattere particolare in oggetto dal segno ".
Esempio:

```
Il carattere /@/     =  Il carattere  /
                        (cancella la barra)
Il carattere /"@/    =  Il carattere /@/
                        (viene stampato)
```

Si noti che il segno " stesso e' un carattere particolare e quindi per ottenerne la stampa va a sua volta fatto precedere dal segno... "
Esempio:

```
Il carattere "       =  Il carattere
                        (scompare)
Il carattere ""      =  Il carattere "
                        (viene stampato)
```

Il comando TERMINAL (abbreviato TERM) consente di cambiare i

caratteri speciali sopra descritti:
il suc formato e' il seguente

```
        CHARDEL     ON
 TERM   LINEDEL     OFF
        LINEND      char
        ESCAPE
```

e permette di:
a) cambiare il carattere speciale con un nuovo carattere (operando char)
b) eliminare la funzione speciale di tutti i caratteri (operando OFF)
c) ripristinare i caratteri speciali standard del CMS (operando ON)

APPENDICE B

- Comandi CP e CMS (4-->13) (49-->52):

Comandi CP

```
LOGON   nome
(L)

IPL     CMS
(I)

LOGOFF
(LOG)
```

Comandi CMS

```
LISTFILE   nome   tipo
(L)          *      *

TYPE  nome tipo  n1 n2
(T)                *

PRINT nome tipo
(PR)

ERASE   nome tipo

COPYFILE nome1 tipo1 A1 nome2 tipo2 A1 ... nomeN tipoN A1 (FROM n FOR m <REP|AP>
(COPY)

RENAME  nome1 tipo1 A1 nome2 tipo2 A1
(R)

EDIT    nome tipo
(E)      oppure
TED     nome tipo

        INPUT  | definizione del file
          |    | comandi SCRIPT  (vedi seguito)
          |    |
(2 RET)   I
          |    |
          |    | SAVE
        EDIT   | FILE
               | comandi EDIT o TED  (vedi seguito)
```

caratteri particolari

```
@ = cancellazione carattere
¢ = cancellazione linea
# = fine linea logica
" = escape
```

- Comandi EDIT e TED (14-->20):

		EDIT	TED
Spostamenti	TOP	TOP	TO
	BOTTOM	B	BO
	UP n	U	U
	UP /stringa/	non esiste	U
	NEXT n	N	N
	NEXT /stringa/	non esiste	N
	LOCATE /stringa/	L	L
	FIND ▶...◀ stringa	F	F
	FUP ▶...◀ stringa	non esiste	FUP
	LINENO	non esiste	LI
	GOTO n	non esiste	G
Elaborazioni	CHANGE /stringa1/stringa2 n m	C	C
	REPLACE linea	R	R
	DELETE n	DEL	D
	DELETE /stringa/	non esiste	D
	OVERLAY ▶...◀ stringa	O	O
	INPUT linea	I	I
Varie	=	=	=
	AGAIN n	non esiste	A
	X comando e X n	X	X
	Y comando e Y n	Y	Y
	TYPE n	T	T
	ZONE n1 n2	Z	Z
	PUT n nome tipo	non esiste	PU
	PUT /stringa/ nome tipo	non esiste	PU
	GET nome tipo modo n m	non esiste	GE

- Comandi SCRIPT (21-->44) (45-->48):

a) formato della pagina

```
.IL n
.PL n
.TM n          .BM n
.HM n          .FM n
```

b) Intestazioni

```
.HE 'S1'S2'S3'  (.TT /S1/S2/S3/)     .FF 'S1'S2'S3'  (.BT /S1/S2/S3/)
.HD 'S1'S2'S3'  (.OT /S1/S2/S3/)     .FD 'S1'S2'S3'  (.OB /S1/S2/S3/)
.HV 'S1'S2'S3'  (.ET /S1/S2/S3/)     .FV 'S1'S2'S3'  (.EB /S1/S2/S3/)
```

c) Spaziatura e salto pagina

```
.SP n
.DS     .SS     .LS  <n|YES|NO>
.LE     <YES|NO>
.PA     <n|+n|-n>
.PD     (.OP)     .PV     (.EP)
.CP n
.PM     <YES|NO>
```

d) Numerazione

```
.PN     <ON|OFF|OFFNO>
.PN     <ARABIC|ROMAN>     .AR     .RO
```

e) Formato del testo

```
.CO     <YES|NO>     .NC     <YES|NO>
.JU     <YES|NO>     .NJ     <YES|NO>
.FO     <YES|NO>     .NF     <YES|NO>
.FP
.LA     <n|YES|NO>   .RA     <n|YES|NO>
.CE     <n|YES|NO>
.IN     <n|+n|-n>    .UN n
.CP     <n|+n|-n>
```

f) Inserimento di note

```
.FN     <BEGIN|END>
.RM     <n <NOSAVE|SAVE|DELETE>>
```

g) Cambiamento di simboli

```
.FN     SET char
.CW     char
.CR     cw1 cw2
.LI     <n|char>
.PS     char
.TR     s1 t1...sn tn
.TI     s1 t1...sn tn
.TI     SET char
```

h) Collegamento tra file diversi

```
.IM     nome n1 n2     .AP     nome n1 n2
.RD     n
.TY     linea          .PB     linea
```

i) Nomi di riferimento simbolici

```
.SR     nome  < 'stringa' | espressione >
.SE     stringa
.UR     linea
.SU     <n|ON|OFF>
.IF     < 'stringa1' op 'stringa2' | numero1 op numero2 >
```

APPENDICE C

Valori di default da assumere esistenti all'inizio di un file Script:

```
.IL 60
.PL 66
.TM 5
.BM 3
.HM 1
.FM 1
.HE '''PAGE %'
.FE ''''
.SS
.LE NO
.EM NO
.PN ON
.PN ARABIC
.FO YES
.IN 0
.FN SET -
.CW ;
.LI -
.PS %
```