L1-MOS

L1 MOS

EDITOR
Reference Manual

olivetti

PREFACE

This manual is a guide to the use of the full-screen EDITOR of the L1 MOS system.

It is addressed to the user engaged in application programming development, or in word processing environment.

For a better understanding of the manual the user is advised to read first "Introduction to MOS" manual.

SUMMARY

The manual described the EDITOR operational characteristics and comnmands. It gives a description of the video, keyboard and command types.

This is followed by a description of each command; the commands are grouped according to their function.

REFERENCES

Read first ...

Introduction to MOS
Code 4002130 G (Vol. 2)

For further information, read ...

Glossary/Glossario
Code 4002140 H (Vol. 1)

MOS Operating Guide
Code 3983040 M

First Edition:
January, 1983 - Release 1.0

Second Edition:
November, 1983 - Release 2.0

Third Edition:
September, 1984 - Releases 3.0/4.0

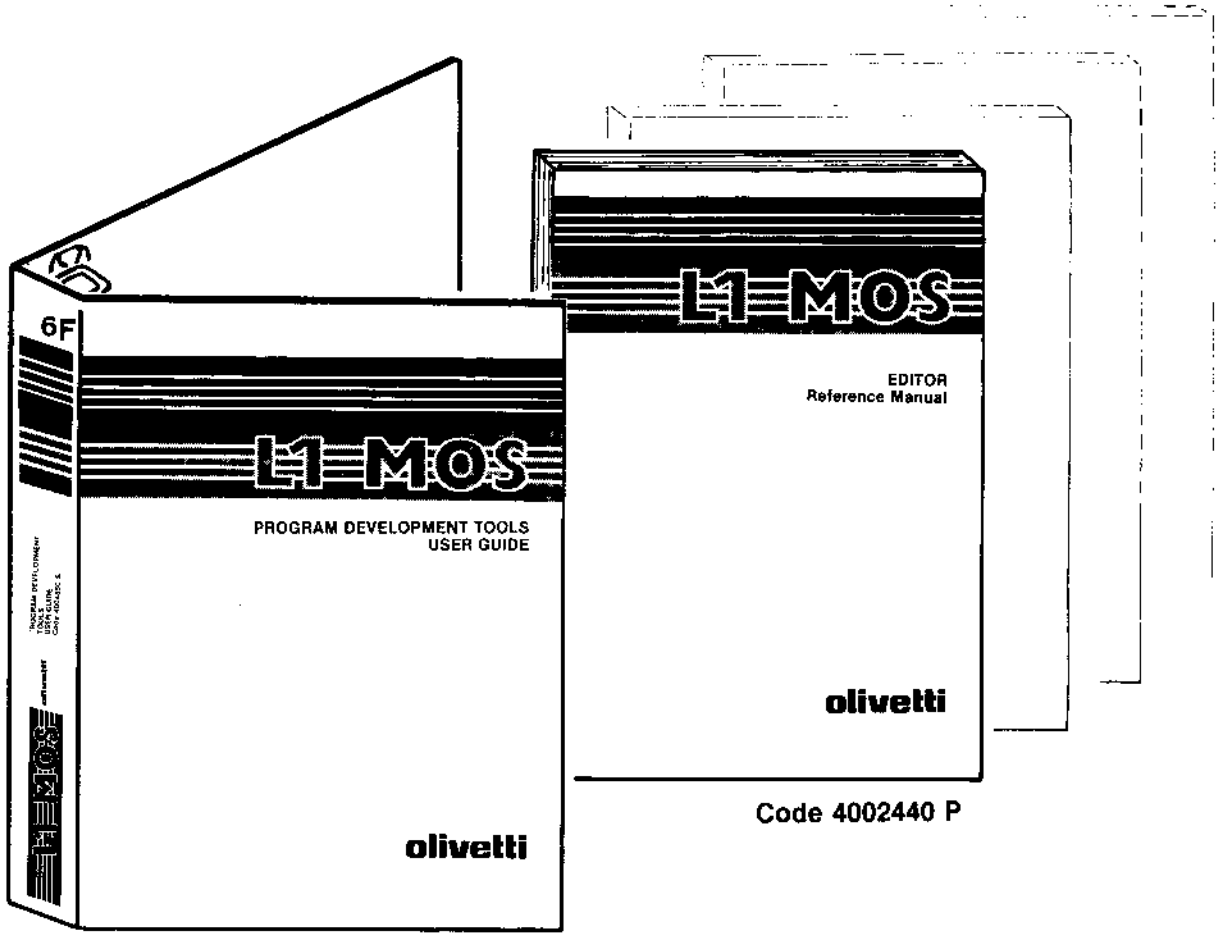
Update:
June, 1985 - Release 5.0

Fourth Edition:
September, 1986 - Release 5

PUBLICATION ISSUED BY:

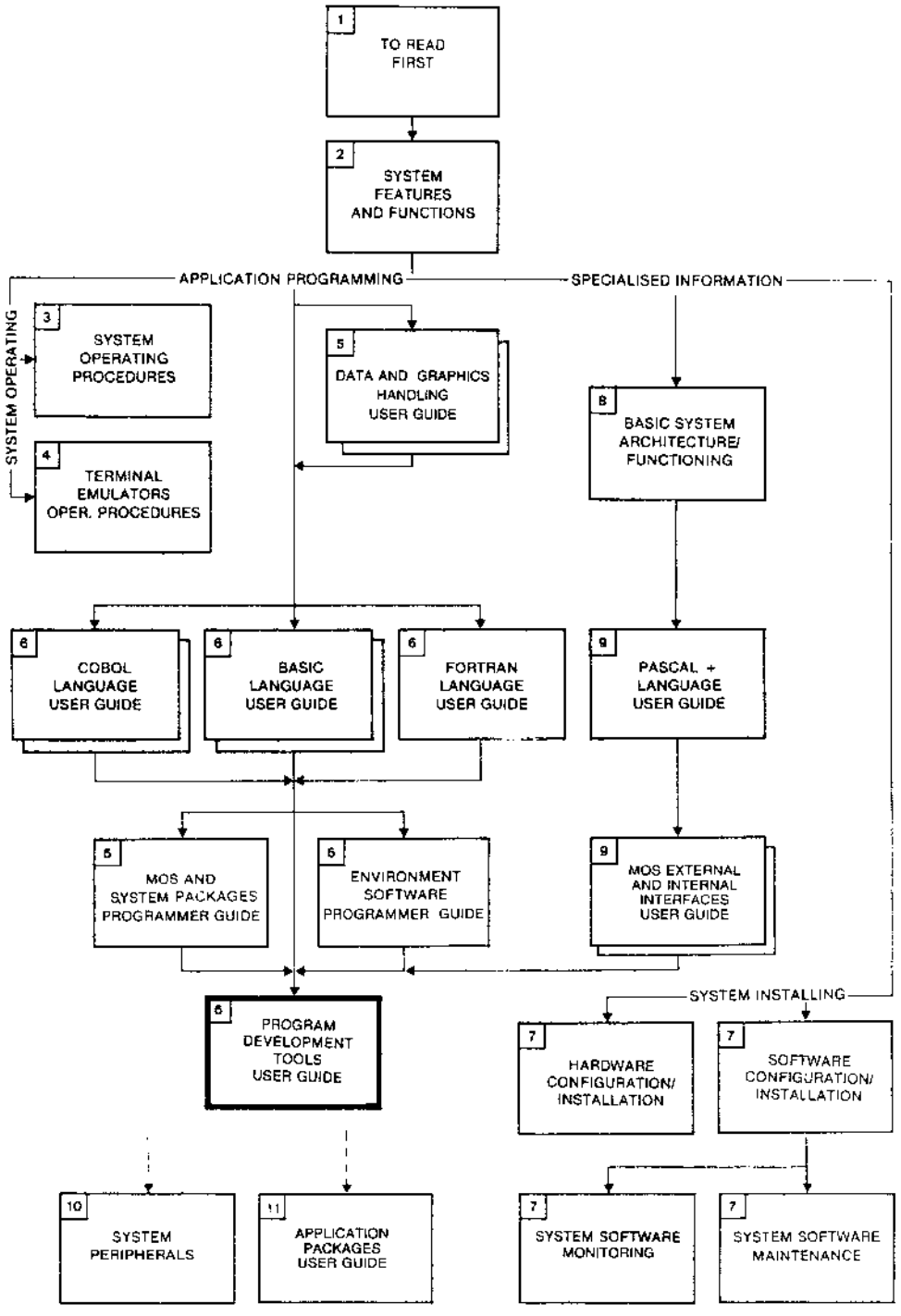
Ing. C. Olivetti & C., S.p.A.
Direzione Documentazione
77, Via Jervis - 10015 IVREA (Italy)

4002440 P



Code 4004850 S

Code 4002440 P


1
TO READ FIRST
2
SYSTEM FEATURES AND FUNCTIONS
APPLICATION PROGRAMMING
SPECIALISED INFORMATION
SYSTEM OPERATING
3
SYSTEM OPERATING PROCEDURES
4
TERMINAL EMULATORS OPER. PROCEDURES
5
DATA AND GRAPHICS HANDLING USER GUIDE
8
BASIC SYSTEM ARCHITECTURE/ FUNCTIONING
6
COBOL LANGUAGE USER GUIDE
6
BASIC LANGUAGE USER GUIDE
6
FORTRAN LANGUAGE USER GUIDE
9
PASCAL + LANGUAGE USER GUIDE
5
MOS AND SYSTEM PACKAGES PROGRAMMER GUIDE
6
ENVIRONMENT SOFTWARE PROGRAMMER GUIDE
9
MOS EXTERNAL AND INTERNAL INTERFACES USER GUIDE
6
PROGRAM DEVELOPMENT TOOLS USER GUIDE
SYSTEM INSTALLING
7
HARDWARE CONFIGURATION/ INSTALLATION
7
SOFTWARE CONFIGURATION/ INSTALLATION
10
SYSTEM PERIPHERALS
11
APPLICATION PACKAGES USER GUIDE
7
SYSTEM SOFTWARE MONITORING
7
SYSTEM SOFTWARE MAINTENANCE

INTRODUCTION

This manual is a guide to the use of the Editor. The operating system component which permits the editing of files containing programs or text.

It is structured as follows:

- Chapter 1 describes the use of the Editor.
- Chapter 2 describes the editing environment: the screen and keyboard, the types of commands and the files which may be edited.
- Chapter 3 describes all the commands. These are grouped by function.
- Appendix A lists all messages which may be displayed in the Editor environment together with their meanings and the reply which the user may enter.

1. USE OF THE EDITOR

The Editor allows a new file to be edited or an existing file to be modified. An 'Editor session' comprises all operations effected between the entry into and exit from the Editor environment. The Editor session may be subdivided into three phases:

- Opening
- Editing
- Closing

In order to operate, the Editor automatically creates the following auxilliary files:

- work-file
- save-file

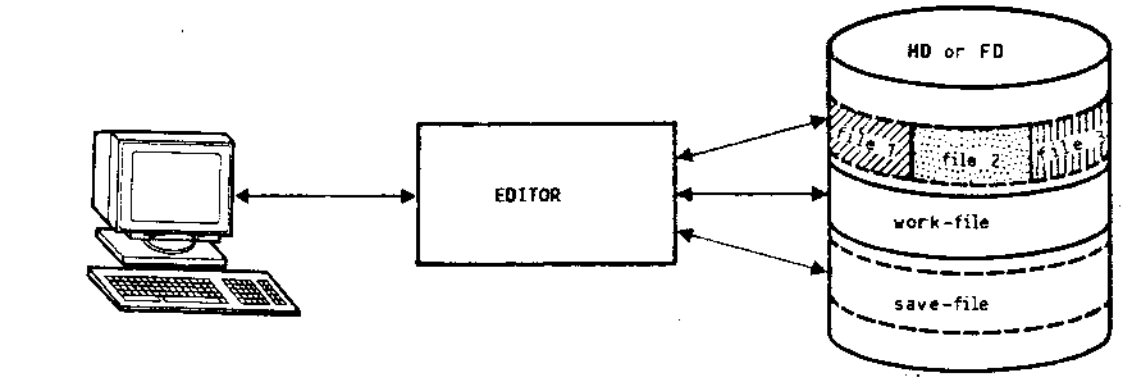


Fig. 1. 1 - Working with the Editor

work-file When the memory area allocated to a file is full and can no longer manage updates a 'work-file' is automatically created under the current working directory by the Editor. All modifications made to the original file or the new file are written into this 'work-file'. 40K are required for the work-file. If there is not sufficient space on the disk an 'OUT OF DISK SPACE' error occurs when trying to save even small files in the working directory.

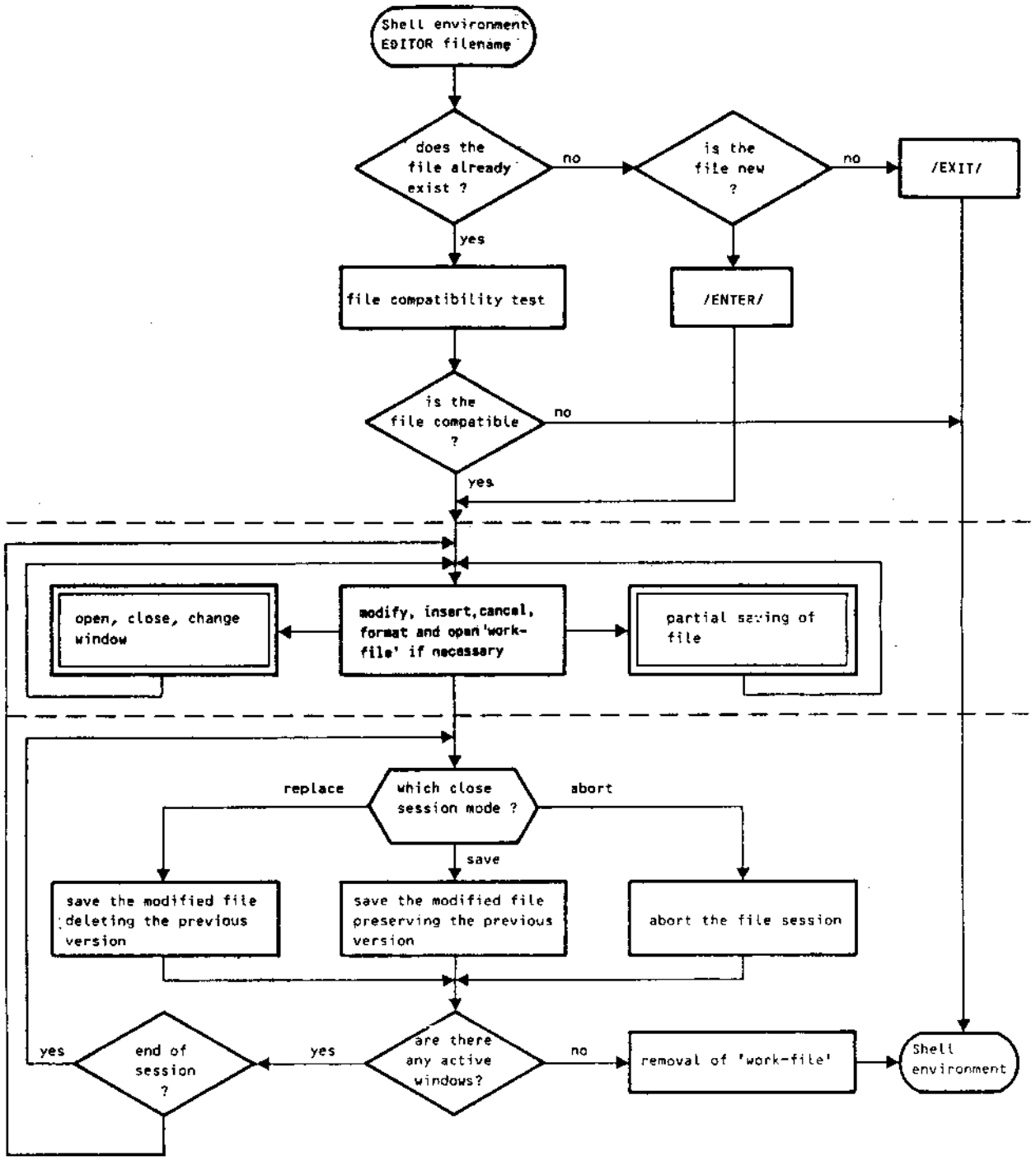


Fig. 1. 2 - Phases of an Editor Session

save-file — When a partial save or a save is requested at the closure of a session a temporary save-file is created. The Editor uses this file to create the updated copy of the original file which assumes the name of the original file, and if necessary to create the back-up version of this file.

The 'save-file' will be removed only on termination of these operations.

If the file to be edited is a new file this is created only at the closure of an Editor session or when a partial save is executed during a session.

The Editor allows the editing of local or remote byte-stream files.

The block diagrams in Figures 1.2 and 1.3 give a general idea of the way the Editor works.

Phase I: opening — The Editor is accessed from the Shell environment by entering the command "Editor filename" where 'filename' is the name or the pathname of the file to be entered.

If the name of the file that has been entered does not exist, the Editor asks if this is the name of a new file or if an error has occured when entering the pathname. If it is a new file the Editor session starts, whereas in the latter case, the session will not be opened.

If, on the other hand, the file already exists, Editor carries out a compatibility test to verify that the file can be handled by the Editor.

Phase II: editing — The editing session, now starts. A window covering most of the screen area is displayed. If the 'filename' used to open the session is new, the window is empty otherwise the text to be edited is displayed in the window. Modification, insertion, deletion and justification operations may be carried out on it.
When the memory area allocated to the file is full and can no longer manage updates the 'work-file' is automatically opened.

Characters, strings and lines in the text may be edited, scrolling the window whenever necessary. It is possible to move through and edit texts by using predefined tabulation stops, and to have the text line numbers displayed on screen when editing.
It is also possible to recover lines which have just

Phase III: closing

Each file opened may be closed in one of three modes: aborting the whole session, storing the modified file while deleting the previous version or storing the modified file in a new file while saving the previous version.
The Editor session is closed when there are no more open files. In this case the system switches back to the Shell Environment.

The following figures illustrate two important Editor Session blocks.

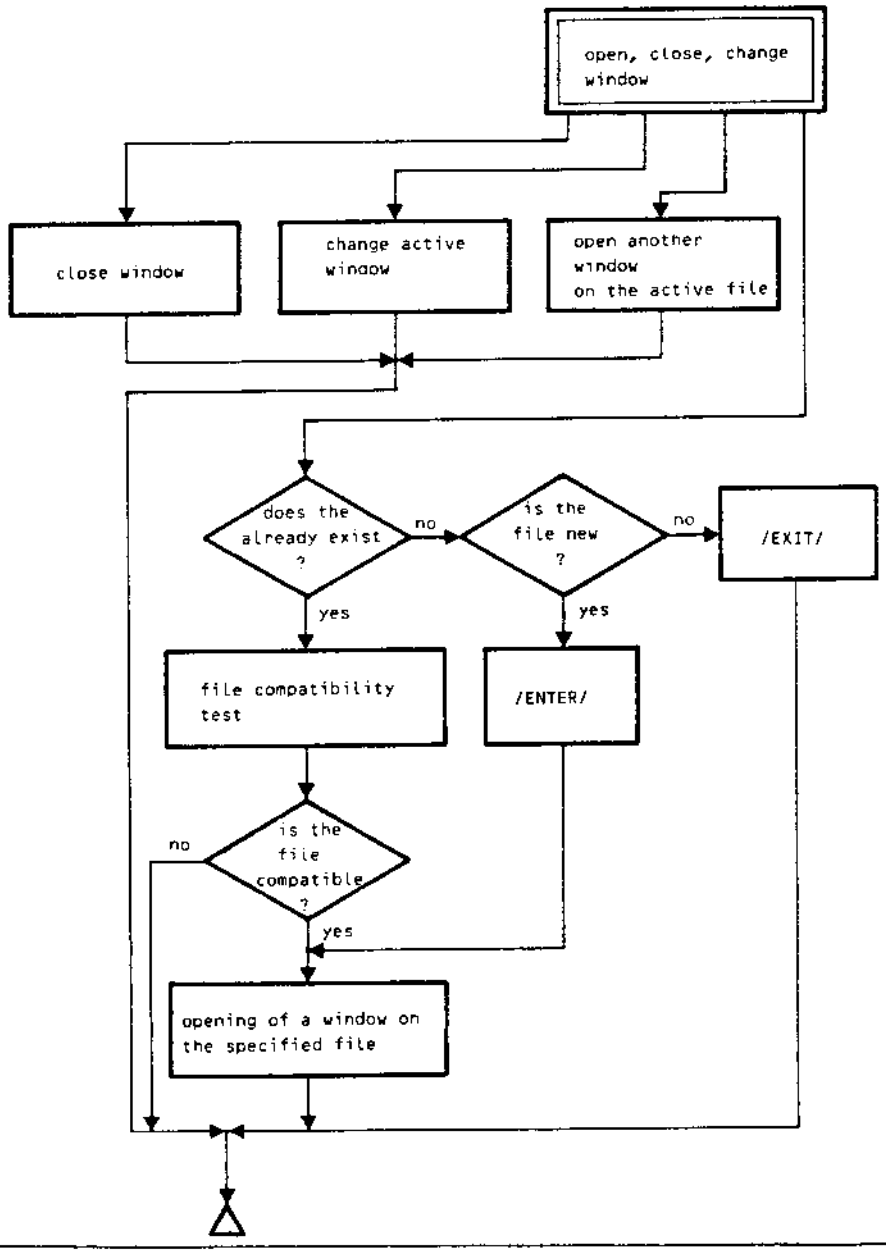


Fig. 1. 3 - Opening a Window

opening, closing and changing windows

A total of three horizontal and/or vertical windows can be opened on the same file or on different ones, allowing more than one file to be edited at a time.

When opening a window on a new file, the Editor asks whether you have entered the name of a new file or whether you have made an error when entering the pathname of the file. If it is a new file a window is opened on the file otherwise the command is aborted. Before opening a window on an existing file a compatibility test is carried out to verify whether the file may be handled by the Editor.

When changing the active window the order in which the windows were created is respected.

The second and/or third windows (in order of creation) opened on the same file may be closed. The user must however close the Editor session if he wishes to close the first window opened on the file.

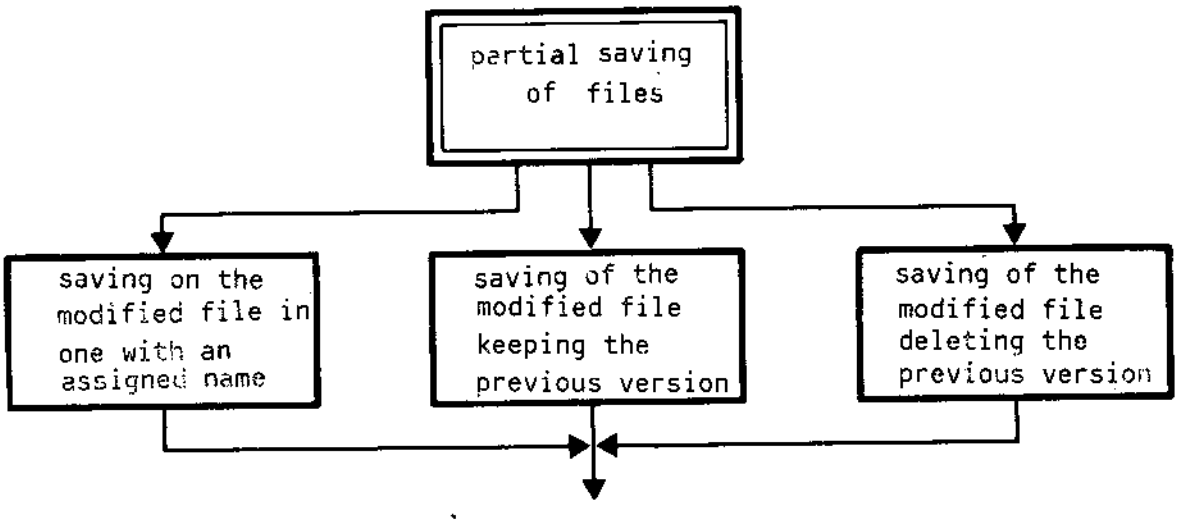


Fig. 1. 4 - Saving a File

saving files

The user may save a file without exiting from the editing session, thus avoiding loss of text in the case of power failure or system breakdown.
Files may be partially saved in one of three ways:

- by storing the modified file under another name thus obtaining more than one version of the same file;
- by storing the modified file keeping the previous version;
- by storing the modified file and deleting the previous version.

2. THE EDITING ENVIRONMENT

This chapter describes the screen and keyboard organisation and the Editor command types.

The characteristics of files which may be edited, the basic terms used and the general functions of the Editor are also described here.

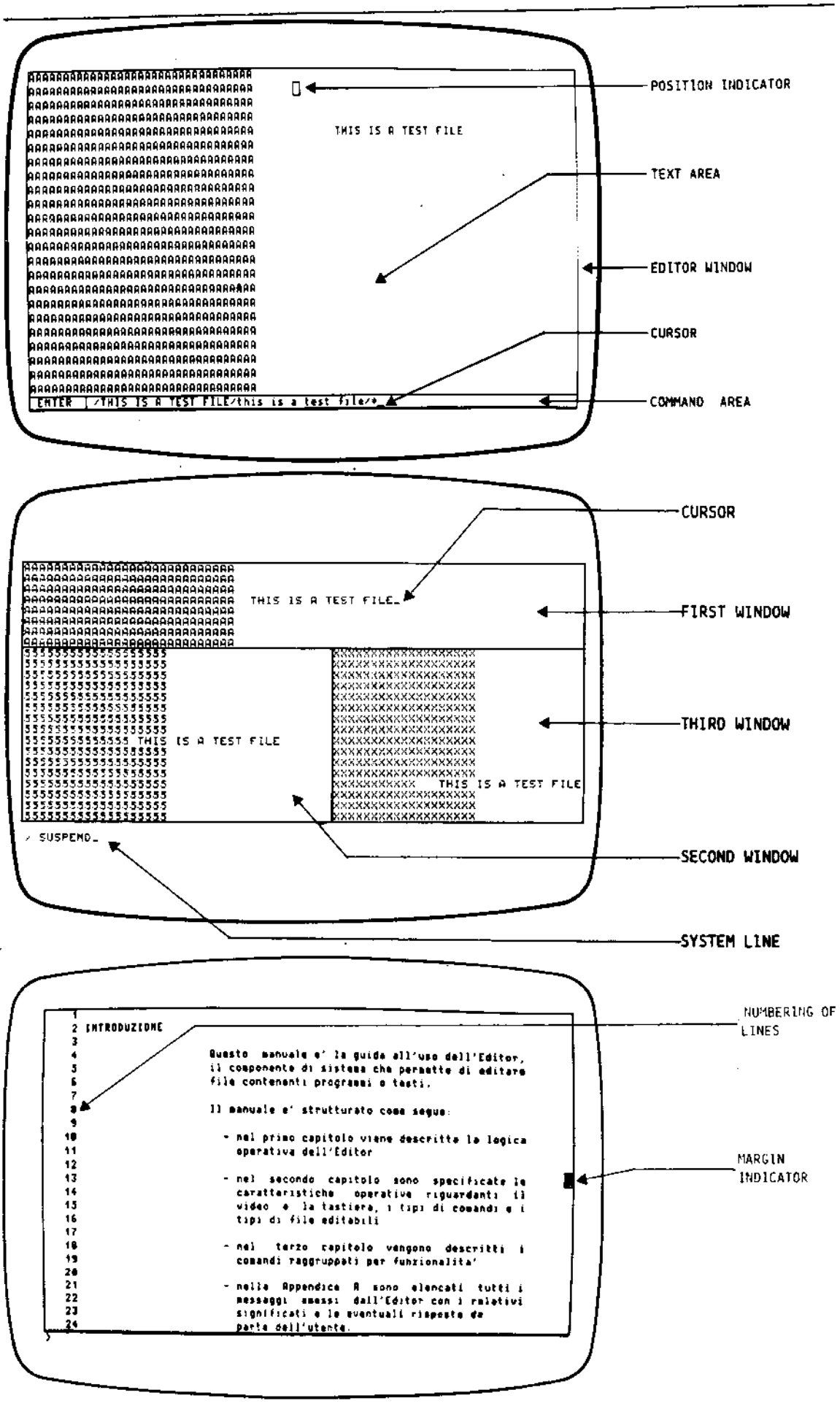


Fig. 2. 1 - Editor Screen

SCREEN ORGANISATION

As soon as the editing environment is entered, a rectangle (window), is displayed. This does not occupy any character positions. (See Figure 2.1).

System Line

If the system line is present, it is the bottom line on the screen. It is reserved for system messages and MCL commands.

Text Area

The text area is inside the window and is reserved for the text to be edited. This area can be divided into (max.3) windows, by means of horizontal lines that do not take up screen space and/or vertical lines that do take up screen space; these windows can be opened in the same or different files.
The size of the window is not predefined and must be determined by the user when the windows are opened.

The absence of the left and/or right margins which frame the window, indicate that the text displayed continues to the left and/or right of the window.

Command Area

The command area is a means of communication between Editor and user. It is not always present and only appears when necessary. When it appears it covers the last line of text on the screen. The command area appears when:

- commands which make use of parameters are entered (maximum length of 70 characters).

- there are error messages or prompts.

When the appropriate command is entered, the name of the file that is being edited and the absolute position of the cursor are displayed.

One of the function keys /RETURN/,/ENTER/or/EXIT/ must be pressed to remove the command area. The text line which was covered by the command area is displayed again.

Cursor	The cursor marks the position in which the Editor is active within an active window and can be shifted everywhere inside it.
Position Indicator	The position indicator is a small rectangle that stands in for the cursor when the cursor has been called to the command area by an /ENTER/ or /EXIT/ command.
Numbering of Lines	Numbering of file lines is optional. When activated, all lines are shifted to the right by 6 characters and the line number is inserted in the first 5. The maximum length of the lines and of the file itself remains unchanged.
Margin Indicator	When the cursor is positioned on the eightieth character and more alphanumeric characters are typed in, these will not be displayed. The eightieth character will be displayed in reverse and an acoustic signal is emitted. Any command entered (other than a write command) will remove the 'reverse' display of the last character. Writing on that same line may continue only if the window is shifted to the left.

The following figure shows the Scientific, Data Processing and Multifunctional keyboard layouts. Only the keys that correspond to Editor commands are shown.
See the MOS, Operating Guide manual for the correspondence between L1 MOS keys and M24 keys (with emulation).
Above the numeric and functional area, there is a template for the top line of the function keys as well as four LEDs, only three of which are handled by the Editor.

READY Indicates that the Editor is ready to receive input from the keyboard. Typing may continue even when this LED is switched off because the keyboard has a buffer of 32 characters. When this buffer is full an acoustic signal is emitted after which any characters typed in will be stored until the buffer has been partially or totally emptied.

L1 indicates that the Editor is in INSERT MODE

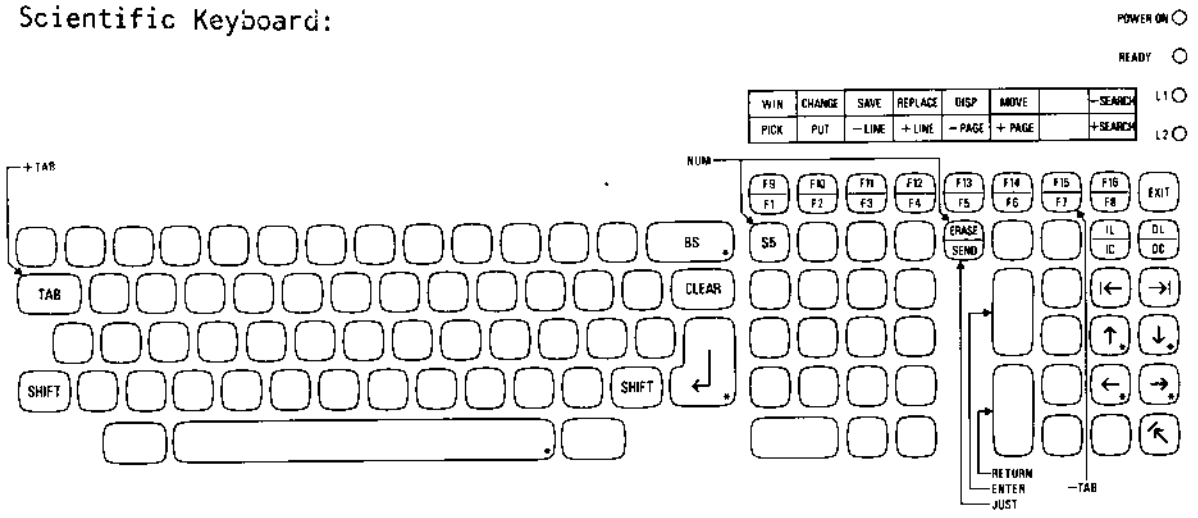
L2 indicates that the Editor is effecting SAVE operations on disk

The second function of each key is obtained by pressing the /CONTROL/ key on DP/BC and Multifunctional keyboards, or the /SHIFT/ key on Scientific keyboard.

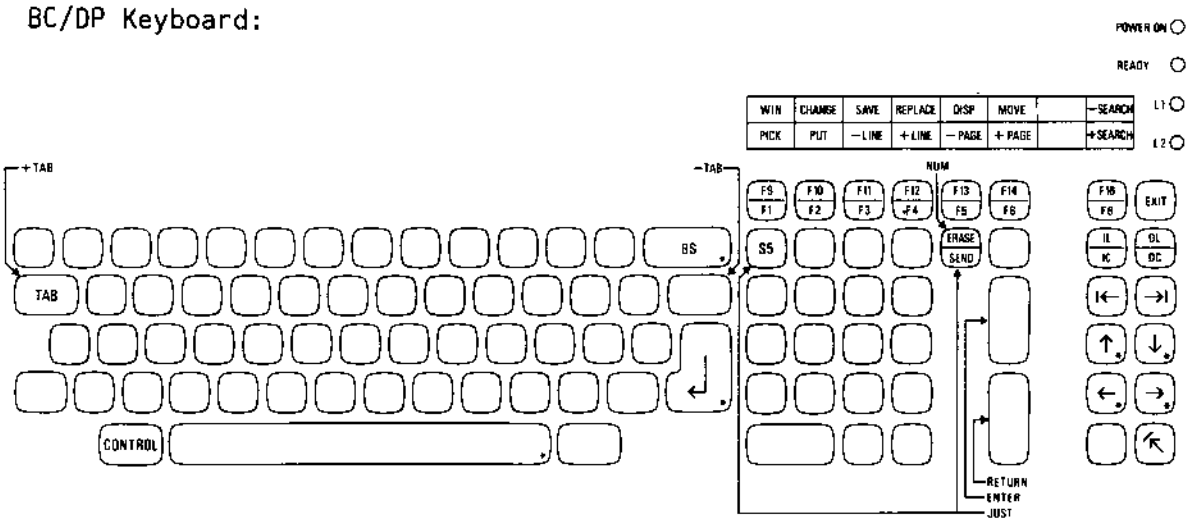
The keys marked with asterisks are equipped with automatic repeat.

The various function keys are summarized in the following tables.

Scientific Keyboard:



BC/DP Keyboard:



Multifunctional keyboard:

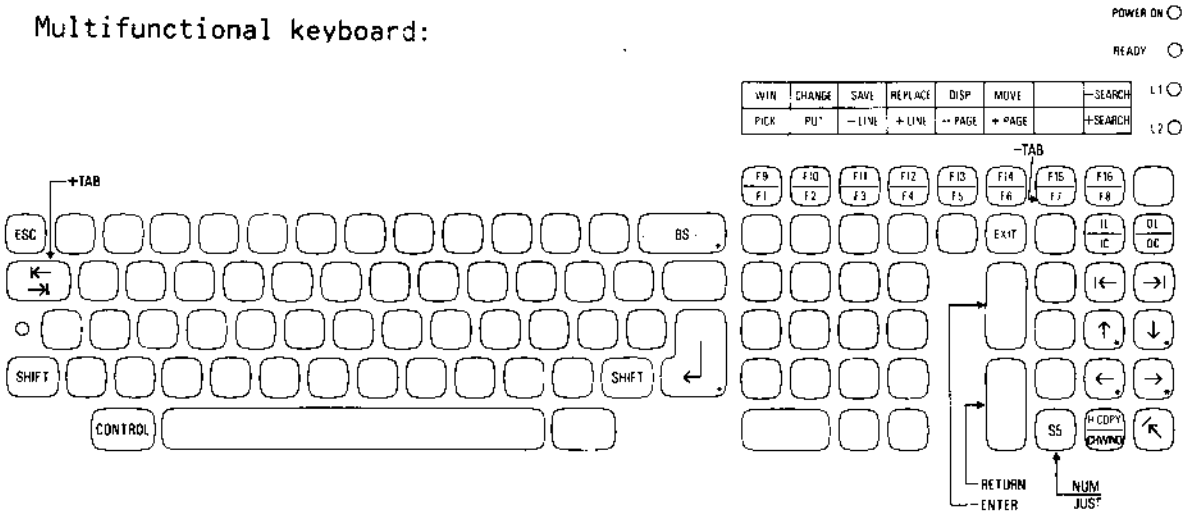


Fig. 2. 2 - Editor Keyboard Types

FUNCTION NAME	KEY	GENERAL FUNCTION
Backspace	BS	Moves the cursor to the left by one character position, deleting the character found there replacing it with a blank. In INSERT MODE, the line is packed to the left, instead of a blank being left. If the character is the last character in a line then it is deleted.
+TAB	TAB or ⇤⇥ (1)	Moves the cursor to the right of the set tabulation point.
-TAB	F7(2)	Moves the cursor to the left of the set tabulation point.
Carriage Return	↵	Moves the cursor to the beginning of the next line. Removes the command area from the screen.
SHIFT	CONTROL SHIFT(3)	Allows the selection of the second function of function area keys on DP/BC keyboard.
WIN	F9(4)	Allows operations on the window.
PICK	F1	Allows line position storage operations
CHANGE	F10(4)	Strings in the text can be changed into preset ones.
PUT	F2	Inserts a line that has been stored temporarily
SAVE	F11(4)	Permits file saving operations.
-LINE	F3	Moves the window down a few lines.
REPLACE	F12(4)	Replaces the old version of the file with the modified one.
+LINE	F4	Moves the window up a few lines.
DISP	F13(4)	Permits execution of compound commands defined by the cursor movement. Once the command has been entered, the line on which the cursor is positioned will be displayed in boldface.
-PAGE	F5	Moves the window down a page in the text.

MOVE	F14(4)	Moves a line from one place to another in the text.
+PAGE	F6	Moves the window up a page in the text.
-SEARCH	F16(4)	Allows searching for a sample string in the part of the text which is above the cursor.
+SEARCH	F8	Allows searching for a sample string in the part of the text which is below the cursor.
ABORT	EXIT(4)	Used as a compound command with /EXIT/ to abort an editor session.
EXIT	EXIT	Used as a compound command, followed by other commands to close an editor session.
NUM	ERASE(4) S5 (5)	Compound command using /ENTER/or/EXIT/ activates or disactivates the numbering of file lines.
JUST	SEND S5(6)	Allows justification operations.
Insert Line	IL(4)	Inserts a blank line above the cursor.
Insert Character	IC	Changes the state of the Editor to that of INSERT MODE and vice-versa. When in INSERT MODE the characters entered are inserted to the left of the cursor position shifting the rest of the string to the right.
Delete Line	DL(4)	Deletes the line where the cursor is. The text moves up automatically to eliminate the space.
Delete Character	DC	Cancels the character indicated by the cursor. The string is moved to the left automatically to eliminate the space.
Left Pan	|←	Moves the window to the left of the text by 1/4 of the screen width.
Right Pan	→|	Moves the window to the right of the text by 1/4 of the screen width. For the movement to be effected there must be at least one character in the last vertical quarter of the screen.
Cursor Up	↑	Moves the cursor up one line in the same column

Cursor Down	↓	Moves the cursor down one line in the same column.
Begin Line	|← (4)	Moves the cursor onto the first character of the current line.
End Line	→| (4)	Moves the cursor to the last character of the current line.
Cursor Left	←	Moves the cursor one character to the left.
Cursor Right	→	Moves the cursor one character to the right.
Home	↖	Moves the cursor to the beginning of the window
ENTER	See fig. 2.2	Permits compound commands to be entered and displays the command area.
RETURN	See fig. 2.2	Same as Carriage Return.

Tab. 2. 3 - Editor Keys

Notes

(1) For the +TAB function on the Scientific and DP/ BC keyboards, use the /TAB/ key and on the Multifunctional keyboard use the / ⇤⇥ / key.

(2) For the -TAB function on the DP/BC keyboard see Fig. 2.2.

(3) For the SHIFT function on DP/BC and Multifunctional keyboards use /CONTROL/, for the Scientific keyboard use /SHIFT/.

(4) These keys must be pressed with either /CONTROL/ or /SHIFT/ depending on (3).

(5) The NUM function for each of the keyboards: for DP/BC use /CONTROL/ /S5/ or /CONTROL/ /ERASE/, for Scientific use /SHIFT/ /ERASE/ and for Multifunctional use /CONTROL/ /S5/.
In the rest of the manual the name of the function (NUM) is referred to, rather than that of the key.

(6) The JUST function for each of the keyboards: for DP/BC use /S5/ or /SEND/ for Scientific use /SEND/ and for Multifunctional use /S5/.
In the rest of the manual, the name of the fuction (JUST) is referred to, rather than that of the key.

TYPES OF COMMANDS

From a functional point of view, the commands can be divided into two groups:

- commands for movement within the text
- commands for text modification

The movement commands do not change the text at all, but permit cursor movement anywhere within it. Commands for addressing within the text and commands for window movement are included in this group.

Modification commands permit insertion, deletion, substitution and searching operations.

From an operational point of view the commands may be considered to be:

- intrinsic
- simple
- compound

Intrinsic Commands

These commands are implicit; no key is required for their execution. The operator actions recognised as intrinsic commands are:

- appending characters to a line;
- replacing a character;
- appending a line to the text.

To execute these, the cursor must be set correctly and the required characters typed-in, appending and replacement are carried out automatically.

Simple Commands

To execute simple commands, position cursor correctly and press the key enabled to execute the required command.

Compound Commands

To execute compound commands, position cursor, enter arguments when necessary and press keys corresponding to the commands selected.

Arguments may be both strings and numbers and are written in the command area. The arguments for some commands are compulsory, as for a search command; the search string must be written in the command area. There are two types of compound commands:

- normal
- cursor-defined

normal

To execute normal commands:

- press /ENTER/ or /EXIT/: the command area is displayed and the cursor is activated within it
- enter the arguments in the command area if necessary
- press the key that corresponds to the required command.

cursor-defined

The commands executed on a screen area defined by the cursor are commands which are executed on a specified number of text lines starting from the initial position of the cursor and its final position which is arrived at by vertical displacement of the cursor.

To execute these commands:

- position the cursor on the first line of the area to be defined
- enter the /DISP/ key. This will highlight the line on which the cursor is currently found
- displace the cursor vertically by the number of lines required for the execution of a specific operation
- enter the key which has been enabled to execute the requested command.

Once the /DISP/ key has been entered, the active windows may not be changed until the required function key is entered, otherwise the effect of /DISP/ is annulled.

The following are definitions of the basic terms most commonly used in this manual:

character A character is a single ASCII character (including non-printable characters), but excluding 0A which is interpreted by the Editor as a Line Feed.

string A string is a sequence of ASCII characters.

line A line is a sequence of characters (≤ 253 characters) terminated by a "Line Feed".

window A window is a section of the screen that frames lines of text to be edited.

page Is the amount of text framed by the window.

tabulation-stop is the column on which the cursor is automatically positioned when /±TAB/ is pressed.

character processing

Characters may be:

- appended
- inserted
- replaced
- deleted

operations on strings

Strings may be:

- searched for
- searched for and replaced

The string to be searched for is stored in a special buffer by Editor and searching may take place at different points of the current editing session.

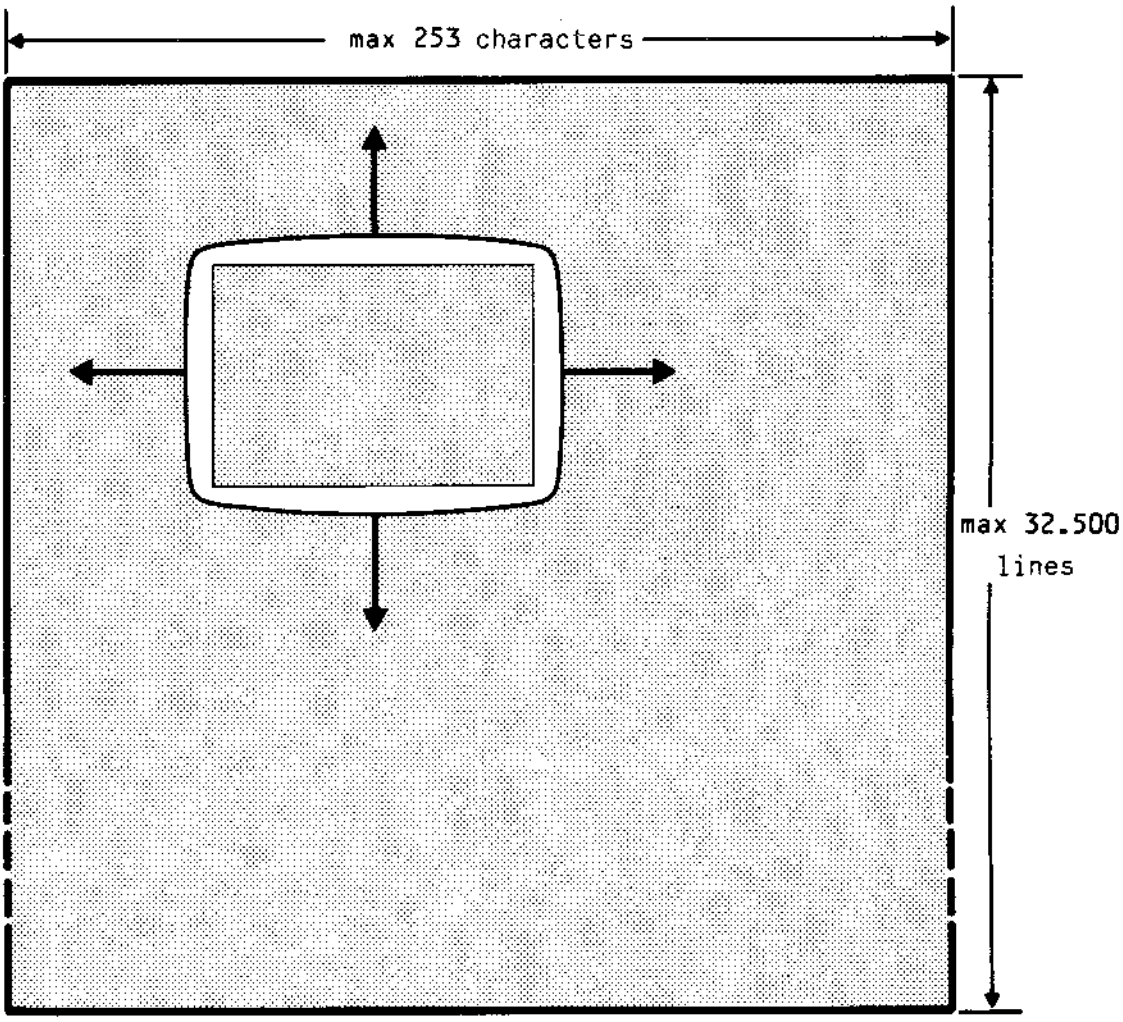


Fig. 2. 6 - Shifting the Window

line processing

Operations that can be carried out on lines are:

- appending
- insertion
- joining the next line to the current line
- splitting the current line
- deletion
- storing position and transferring.
- numbering of file lines (for display purposes only

The deleted lines are stored by Editor in a buffer.

	The last line, or groups of lines to be deleted can be reinserted in the text.
window processing	Up to three windows can be opened on the screen. Permitted operations on these are: - opening and closing - selecting the active window - vertical and horizontal scrolling (see Fig. 2.6)
tabulation	Permits text to be edited according to predefined tabulation stops.
justification	Allows text to be aligned to two or more predefined margins. This alignment is obtained by transferring words from one line to the previous or next line and by varying the number of spaces between the words.

COMPATIBILITY OF FILES WITH EDITOR

Whereas the files which are created and edited by the Editor are obviously compatible with the Editor, problems of compatibility may occur with files created and/or edited in environments other than that of the Editor.

Characteristics of Files which May Be Edited

The Editor can only edit byte-stream files.
A byte-stream file which may be edited may have up to 32,500 lines of a maximum of 253 characters each. The Editor treats the file as a sequence of lines separated by a Line Feed ('0A' hexadecimal configuration which is interpreted as Line Feed.) If the last line does not end with a 'Line Feed', this will be automatically added by the Editor.

If the file contains a sequence of characters which does not contain a 'Line Feed' and which is large with respect to the size of the file, only the part of the file until the last carriage return before this sequence will be considered as valid. If the session is continued this latter part of the file is lost.

An Editor session may also be opened on a file having lines which are longer than 253 characters. The execution of Editing commands on such files will cause certain characters to be substituted by a "Line Feed". Substitution is not regular, and will vary depending on the sequence of commands effected.

When a session is opened on a file which is not considered to be compatible by the Editor, the Editor will send out a message indicating the reason why the file opened cannot be operated on in the normal manner.
The user may then choose not to continue with the session leaving the file unaltered.

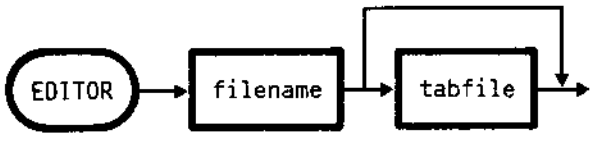
3. EDITOR COMMANDS

This chapter describes the Editor commands.

The syntax used in the description of commands is explained in Chapter 3 of the "Guide to the Literature" manual.

OPENING OF AN EDITOR SESSION: EDITOR

The command that opens an Editor session must be entered in the Shell environment.



where:

filename is the name or the pathname of the file to be edited.

tabfile is the name or pathname of the file which contains the default tabulation stops. This file is referred to as the tabulation file.

See the section "TABULATION" in this chapter.

If the tabulation file is called, it must have been created using this command.

If the command is accepted, the window containing the first part of the file appears on the screen and the cursor moves to the first character in the text.

If the specified file is a new one or its name is not under the specified directory, the command area appears with the following message in it:

** NEW FILE ? **

The user may:

- press /ENTER/, and the editing session begins.
- press /EXIT/ to exit from the editing session without creating a new file. See note.

If "tabfile" is the name or pathname of a non-existent file or a file that does not exist under the directory specified, the following message is displayed in the command area:

** ERROR IN OPEN TABFILE **

The user should now press /EXIT/ to leave the command.

The file name may have a maximum of 12 characters. The whole pathname may have a maximum of 60 characters.

Note

Next to some messages the user is given a choice of answers, e.g.:

EDITOR PLUTO

this message appears:

** NEW FILE ** Press ENTER or EXIT

Only the first part of the message is referred in the following.

Examples

1. EDITOR /USR/FRANK/Comp/SATURN

   Opens an Editor session on the file SATURN where '/USR/FRANK/Comp/SATURN' represents the complete "Saturn" file pathname.

2. EDITOR Jupiter

   Opens an Editor session on the file "Jupiter" under the current working directory.

3. EDITOR Jupiter Saturn

   Opens an Editor session in the file "Jupiter" in which the tabulation stops defined in the file "Saturn"can be used. Both these files are under the current working directory.

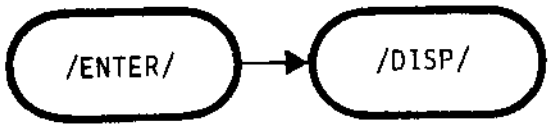
4. It is possible to define an MCL procedure named E which has the following structure:

```
READ 'SPECIFY THE NAME OR PATHNAME FILE:'
EDITOR %FILE
```

Once the procedure has been activated, the name of the file on which the Editor session is to start must be typed in, in reply to the prompt:

SPECIFY THE NAME OR PATHNAME OF THE FILE:

DISPLAYING THE FILE NAME AND THE CURSOR POSITION: /ENTER/,/DISP/



The command area appears and displays the name or the pathname of the active file and the absolute position of the cursor within it.

If the pathname is longer than the maximum number of characters permitted in the command area, it is truncated starting from its root, this will be substitued by the characters '??' .

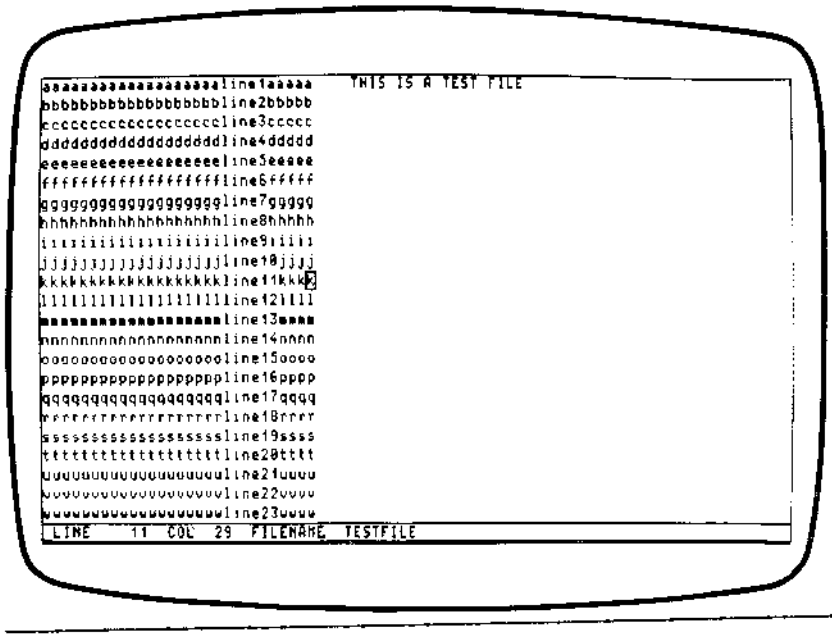


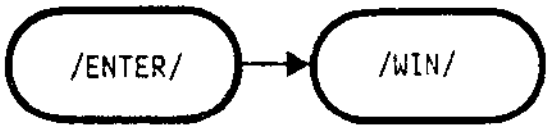
Fig. 3. 1 - Example of the Compound Command /ENTER//DISP/

OPENING, CHANGING AND CLOSING WINDOWS: /EXIT/,/ENTER/,/WIN/

Changing the Active Window

This command changes the active window. The cursor becomes active within the window which was next created, and appears in the same position it was in before it was disactivated unless there have been text modifications within another window opened on the same file. If there is only one window open this command has no effect.

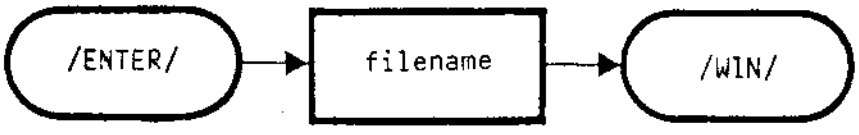
Opening Another Window on the Active File



A new window is opened on the active file starting from the line after the current cursor position if

the window is horizontal, or from the column after the current cursor position if the window is vertical (note 1).
To open a vertical window the cursor must be positioned on the first line of the current window. The new window ends where the next one opens (or above the system line) and the first part of the file is displayed in it (note 2).

Opening a Window on the Specified File



where:

filename is the name or pathname of the file on which the user wishes to open a new window.

A new window is opened on the specified file starting from the line after the current cursor position if the window is horizontal, or from the column after the current cursor position if the window is vertical (note 1).
To open a vertical window the cursor must be positioned on the first line of the current window. The new window ends where the next one opens (or above the system line) and the first part of the file is displayed in it (note 2).

If the name or pathname specified, does not correspond to that of an existing file, the following message is displayed in the command area:

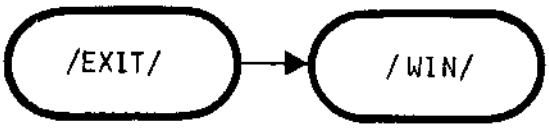
** NEW FILE ? **

The user may:

- press /ENTER/: this will create a new file with the specified name or pathname; a window is opened on this file.

- press /EXIT/: to abort the command

Closing a Window on a File



Closes the active window. This command is accepted only if more than one window is open on the same file and the window being closed is not the first one opened. If the above conditions are not satisfied the following message appears:

** INVALID OPERATION **

One of the commands for closing a session must be entered to close a single window open on a file (see paragraph Saving and Closing a Session).

Once a window has been closed the active window automatically becomes the one which was opened first.

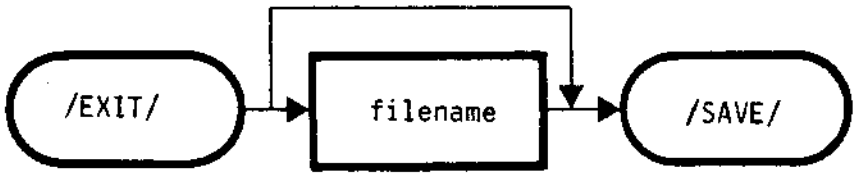
Notes

1. When a new window is opened this automatically becomes the active window.

2. It is not possible to open a new window if the cursor is positioned on the last line of one of the existing windows.

SAVING AND CLOSING THE SESSION: /EXIT/,/SAVE/,/REPLACE/,/ABORT/

Closing and
Saving of File
Keeping the
Previous Versions



where:

filename is the name or the pathname under which the active file is to be saved and closed (note 1).

If the 'filename' is specified, the active file is closed and saved with that name
If 'filename' already exists, the following message appears in the command area:

** FILE ALREADY EXISTENT **

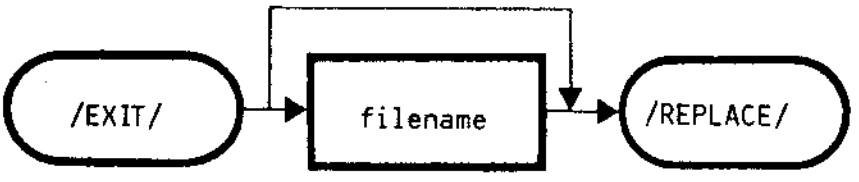
The user may:

- Press /ENTER/. The save operation is executed and the previous contents of 'filename' lost.

- Press /EXIT/. The save operation is aborted.

If 'filename' is not specified, the active file is closed and saved with its original name.
The original name with the suffix .B is given to the previous version of the active file (note 2), and, if this file already exists, its previous contents are lost (notes 3,4,5).

Closing and
Replacing of the
Previous Version



where:

filename is the name or the pathname under which the file is to be closed (note 1).

If 'filename' is specified, the active file is closed and saved with that name replacing the previous version of the file. If 'filename' already exists, the following message appears in the command area:

** FILE ALREADY EXISTENT **

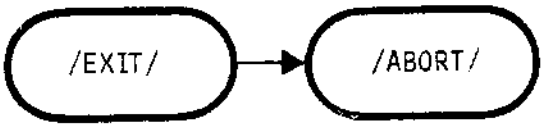
The user may:

- Press the /ENTER/ key. The save operation is performed and the previous contents of 'filename' lost.

- Press the /EXIT/ key. The save operation is aborted.

If 'filename' is not specified, the active file is closed and saved with its original name, covering the previous version (note 3,4,5).

Closing and Aborting the Session



If no changes have been made to the file the whole session is aborted and the current file is closed in the same state as it was when it was last saved. If the file has been modified the following is displayed:

** FILE MODIFIED **

and the user can:

- press /ENTER/ to confirm the abort.
- press /EXIT/ to remain in the Editor session.

See notes 3 and 4.

Notes

1 'filename' must differ from the name or pathname of any file which is still open in the Editor session. If this condition is not verified then the following error message appears:

** INVALID OPERATION **

2 If the file name is longer than 10 characters, this is truncated at the 10th character and suffix .B is added.
If there are two files both having names which are 11 or 12 characters long, distinguished only by the 11th and/or 12th characters, truncation from the 10th character and the addition of the suffix .B will render the two names identical.

For example after truncating the files CHRISTINE1F7 and CHRISTINE1G1 both become CHRISTINE1.B. In the above example the .B version of the last file saved

replaces the .B version of the other file.

To avoid this problem it is advisable to have file names which are distinguished from each other by characters other than the 11th or 12th.

3 All the windows opened on the same file are closed and if there are no other windows opened on other files, the Editor session is closed and the system switches to the Shell environment.

4 If a close and save operation is effected on a file which has not been modified since the last save operation, the following message is displayed.

** NO MODIFICATIONS **

The user may:

- Press /ENTER/ to execute the compound command.

- Press /EXIT/ to ignore the compound command and close the Editor session on the file. In practice the command is interpreted as the /EXIT//ABORT command.

5 Led L2 lights up during the execution of the command.

6 On an alias file the close and save operation modifies this file into a byte-stream file which may no longer be used as an alias.

Examples

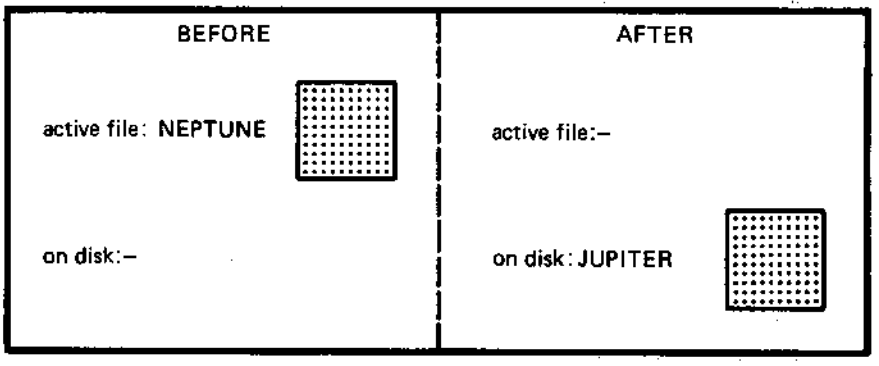
The strings enclosed by inverted commas must be entered in the command area.
All file names correspond to the working directory.

1. The file "NEPTUNE" is created and edited without carrying out any save operation before the execution of the compound command:

/EXIT/ The command area is displayed and the cursor appears within it.

'JUPITER' This is the name used to save the "NEPTUNE" file. This file does not already exist on disk.

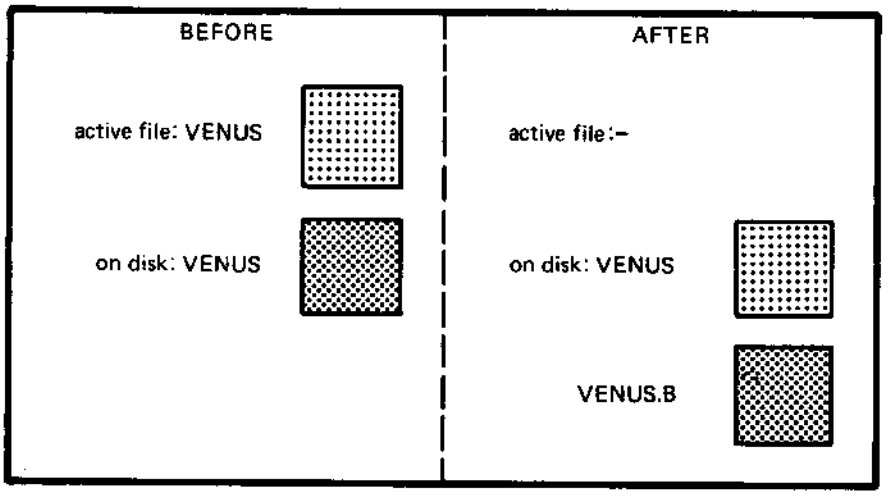
/SAVE/ The "NEPTUNE" file is closed and stored under the name "JUPITER" (see figure below).



2. The file "VENUS" is opened and modified before entering the compound command:

/EXIT/ The command area is displayed and the cursor appears in it.

/SAVE/ The file "VENUS" is closed and stored under its own name; the previous or back version is saved in the "VENUS.B" file (see figure below).

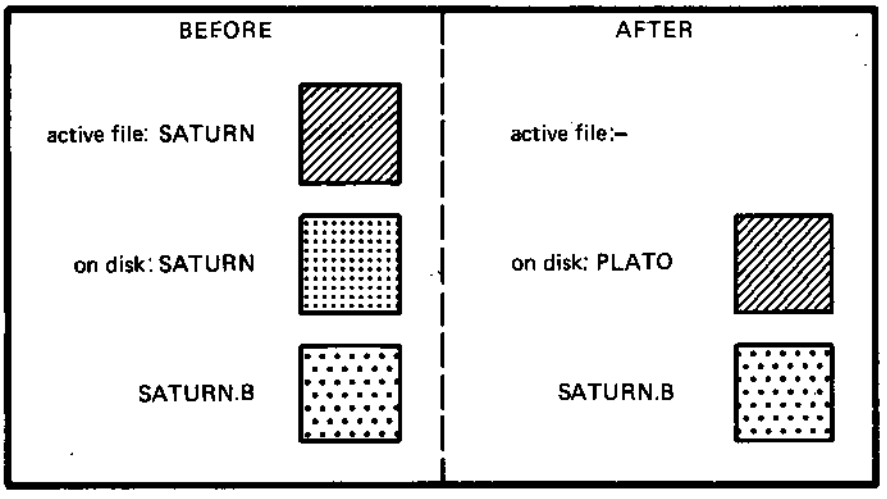


3. The file "SATURN", already exists on disk with a copy "SATURN.B", is modified before entering the compound command:

/EXIT/ The command area is displayed and the cursor appears in it.

'PLATO' The name of a file which does not exist on disk.

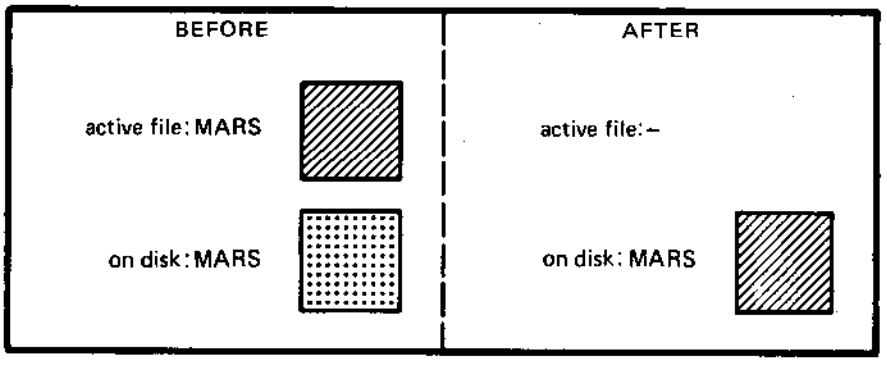
/REPLACE/ The file "SATURN" is closed and stored under the name "PLATO" replacing the previous version of "SATURN" (see figure below).



4. The file "MARS" is opened and modified before entering the compound command.

/EXIT/ The command area is displayed and the cursor appears within it.

/REPLACE/ The file "MARS" is closed and stored on disk replacing its previous version (see figure below).



5. The file "URANUS" is opened and modified before entering the commands:

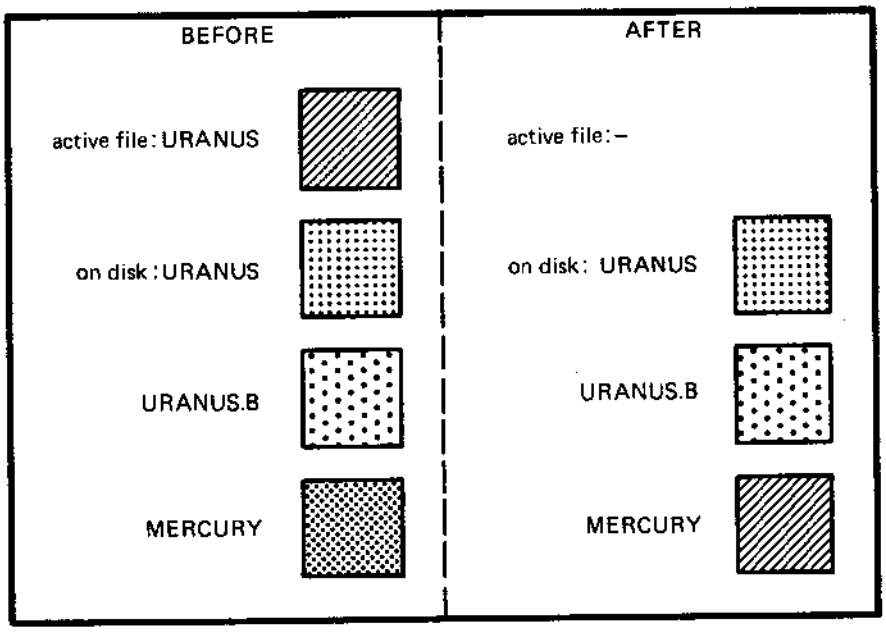
/EXIT/ The command area is displayed and the cursor appears within it.

'MERCURY' This is the name used to save the file "URANUS". The file "MERCURY" already exists on disk.

/SAVE/ The following message is displayed in the command area:

** FILE ALREADY EXISTENT **

/ENTER/ The file "URANUS" is closed and stored under the name "MERCURY" overwriting the previous contents.



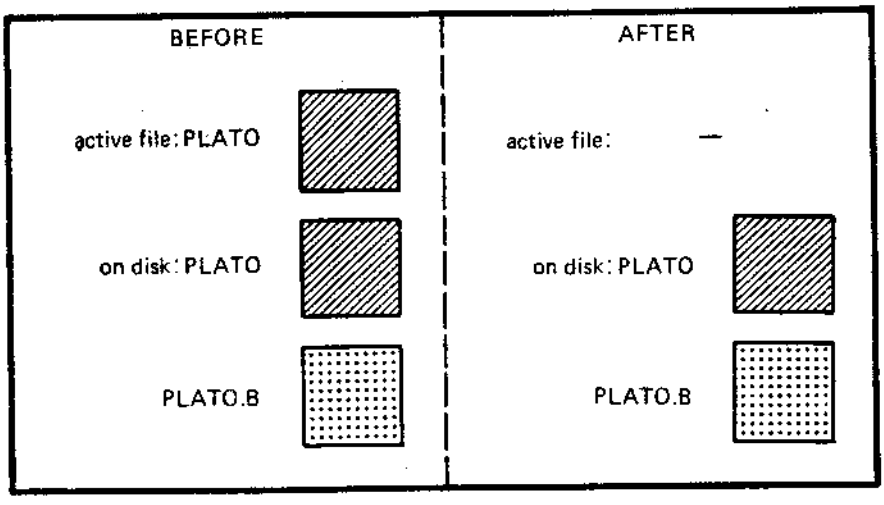
6. The file "PLATO" is opened but is not modified before entering the commands:

/EXIT/ The command area is displayed and the cursor becomes active in it.

/SAVE/ The following message appears in the command area:

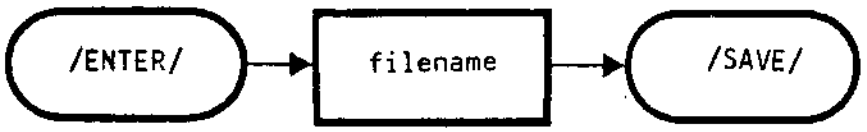
** NO MODIFICATION **

/EXIT/ The command area is removed and the compound command is ignored.



SAVING WITHOUT CLOSING THE SESSION: /ENTER/,/SAVE/,/REPLACE/

Saving of File
with Another Name
Keeping the
Previous Versions



where:

filename is the name or the pathname under which the file is to be saved (note 1).

The active file is saved in the file with the specified name, leaving the original version unchanged (notes 2,4,5).
If 'file name' already exists, the following message appears in the command area:

** FILE ALREADY EXISTENT **

The user may:

- Press /ENTER/. The save operation is executed and the previous contents of 'file name' lost.

- Press /EXIT/. The save operation is aborted.

Saving of the File with the Same Name

The active file is saved with the original name (note 2) and the the previous version is given a name made up of the original one plus the suffix .B. If a version of this file existed before, it is lost (notes 3,4,5).

Saving the Active File and Changing its Name

where:

filename — is the name or the pathname under which the file is to be saved (note 1).

The active file is saved with the specified filename, overwriting what was previously in it. From now on the active file has the specified name (notes 2,5).
Use this command to change the name of an active file.

Replacing of the Previous Version of the File

The active file is saved with the original name, overwriting the previous version (notes 2,4,5).

Notes

1 'file name' must differ from the name or the pathname of the active file, otherwise the following message is displayed:

** INVALID OPERATION **

2 The active window and the cursor position are not modified.

3 If the file name is longer than 10 characters, this is truncated at the 10th character and suffix .B is added.
If there are two files both having names which are 11 or 12 characters long, distinguished only by the 11th and/or 12th characters, truncation from the 11th character and the addition of the suffix .B will render the two names identical.
For example after truncating the files CHRISTINE1F7 and CHRISTINE1G1 both became CHRISTINE1.B. In the above example the .B version of the last file saved replaces the .B version of the other file.
To avoid this problem it is advisable to have file names which are distinguished from each other by characters other than the 11th or 12th.

4 If a save operation is executed on a file which has not been modified since the last save operation, the following message is displayed:

** NO MODIFICATION **

In response to this the user may:

- press /ENTER/ for the command to be executed
- press /EXIT/ for the command to be aborted

5 Led L2 automatically switches on during command execution.

Examples

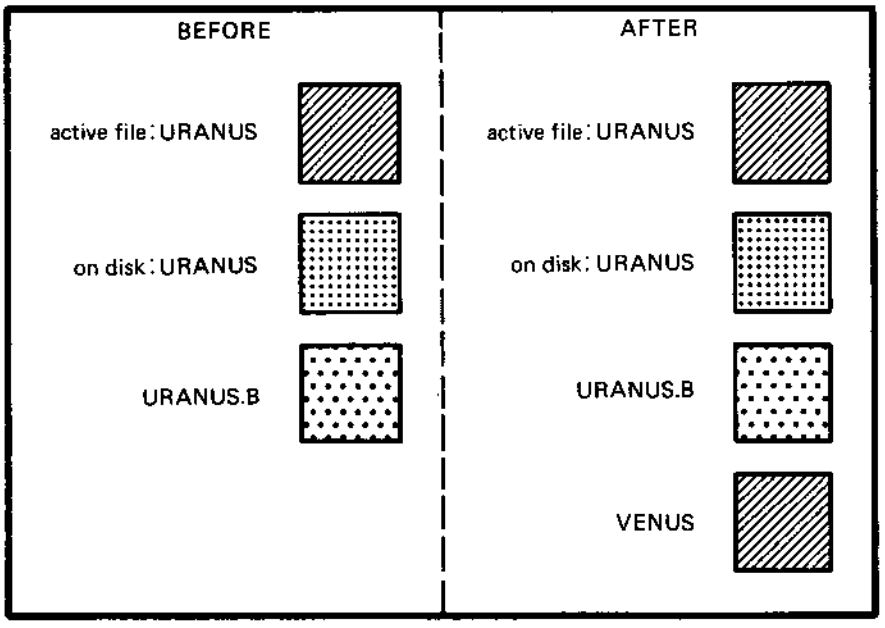
The strings enclosed in inverted commas must be entered in the command area.

1 The file "URANUS" is opened and modified before entering the compound command:

/ENTER/ The command area is displayed and the cursor appears within it.

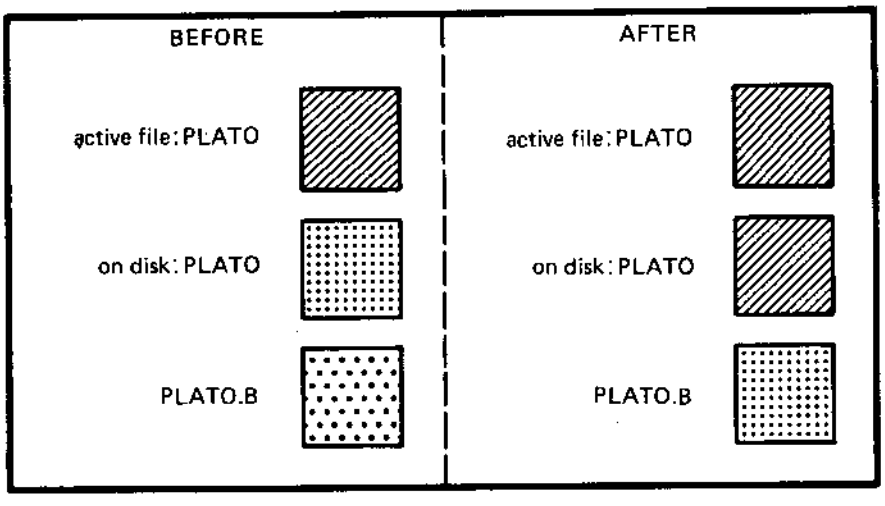
'VENUS' Name under which the file "URANUS" is saved. It does not exist on disk.

/SAVE/ The "URANUS" file is stored on disk under the name "VENUS".



2 The file "PLATO" is opened and modified before entering the command:

/SAVE/ The modified file "PLATO" is saved under its original name. The previous version is saved in the file "PLATO.B" . The file previously contained in "PLATO.B" is consequently lost.

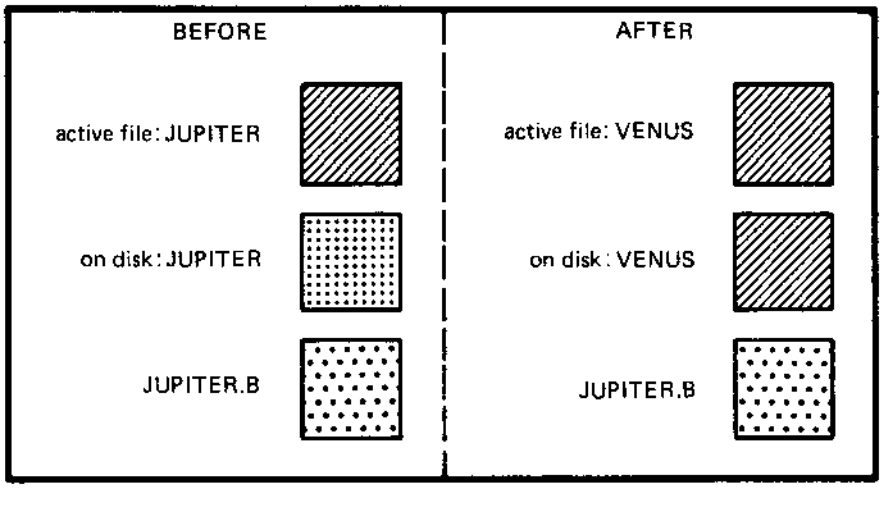


3 The file "JUPITER" which already exists on disk with a JUPITER .B version, is opened and modified before entering the command:

/ENTER/ The command area is displayed and the cursor appears within it.

'VENUS' The name under which the file "JUPITER" is saved. The file "VENUS" does not exist on disk.

/REPLACE/ The file "JUPITER" is stored under the name "VENUS" overwriting its previous version. The name of the active file .VENUS becomes the active file.



MOVING THE CURSOR: /↑ /,/ ↓ /,/→ /,/← /,/↵ /,/↖ /, /±TAB/

↑ moves the cursor up one line in the same column.

↓ moves the cursor down one line in the same column.

→ moves the cursor one character to the right on the same line.

← moves the cursor one character to the left on the same line.

All these movements are circular: when the cursor gets to the edge of the window, it appears again on the opposite side of the same line and column.

/CTRL/ ← Begin Line. Moves the cursor onto the first character of the current line, without affecting the window contents.

/CTRL/ → End Line. Moves the cursor onto the last character of the current line without affecting the window contents.

↵ Carriage Return. The cursor is moved to the beginning of the following line. Movement is not circular: if the cursor is at the last line in the window the text is scrolled upwards by one line.

↖ Home. The cursor is moved to the top left corner of the active window. If the cursor is there already, the command will not be executed.

/+TAB/ The cursor is moved to the right of the current line by one tabulation stop.

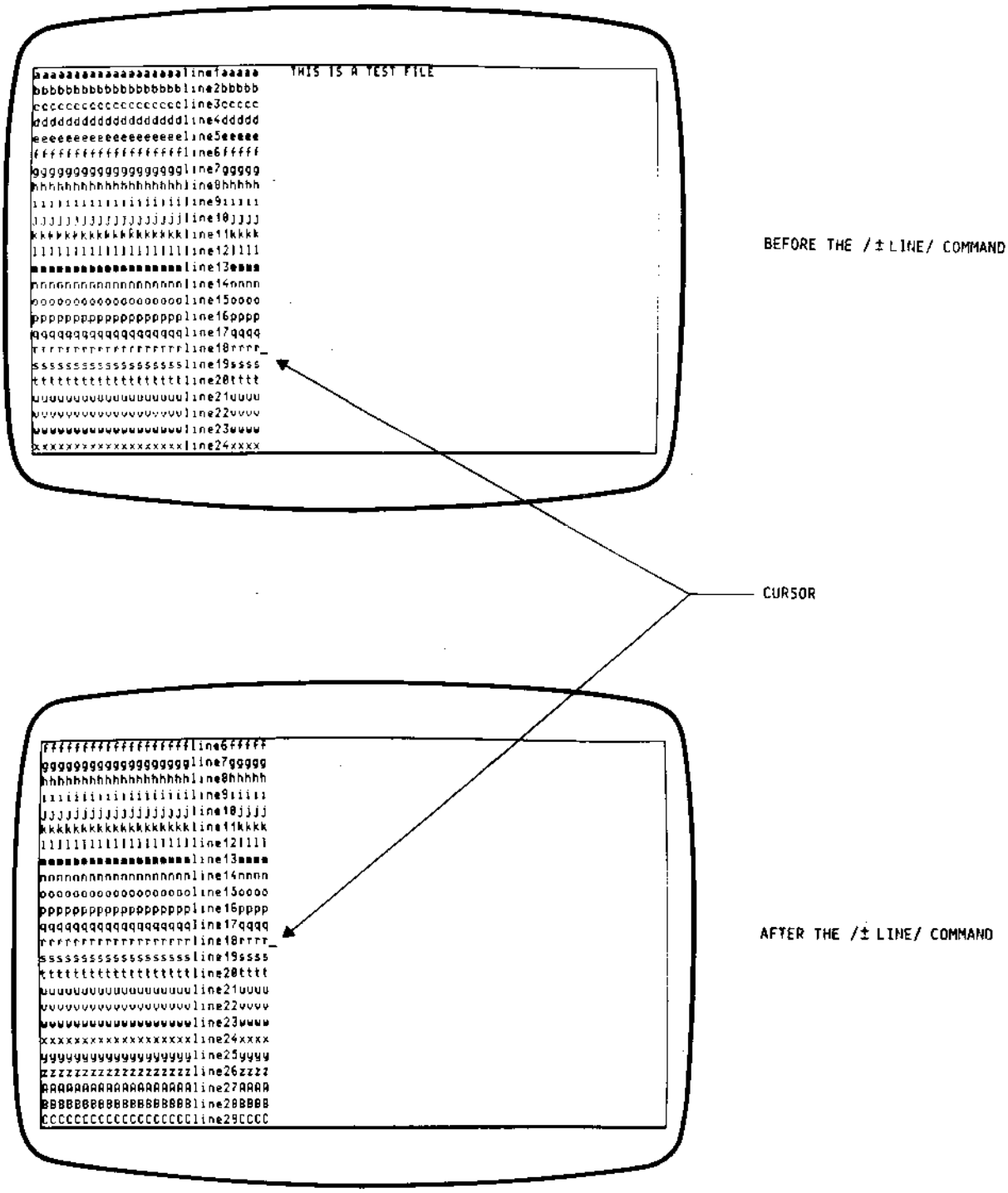


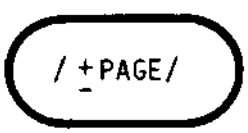
Fig. 3. 2 - Movement of Lines in a Window

MOVING WINDOWS VERTICALLY: /ENTER/,/±LINE/,/±PAGE/

Movement of Lines in a Window

Moves the window backwards or forwards more than one line in the file depending on whether /+LINE/ or /-LINE/ has been entered. Conventionally the exact number of lines is equal to a fifth of the lines present in the window rounded up to the next integer (note 1).

Movement of a Page



Moves the window backwards or forwards by a page in the file according to whether /+PAGE/ or /-PAGE/ has been entered (note 2).

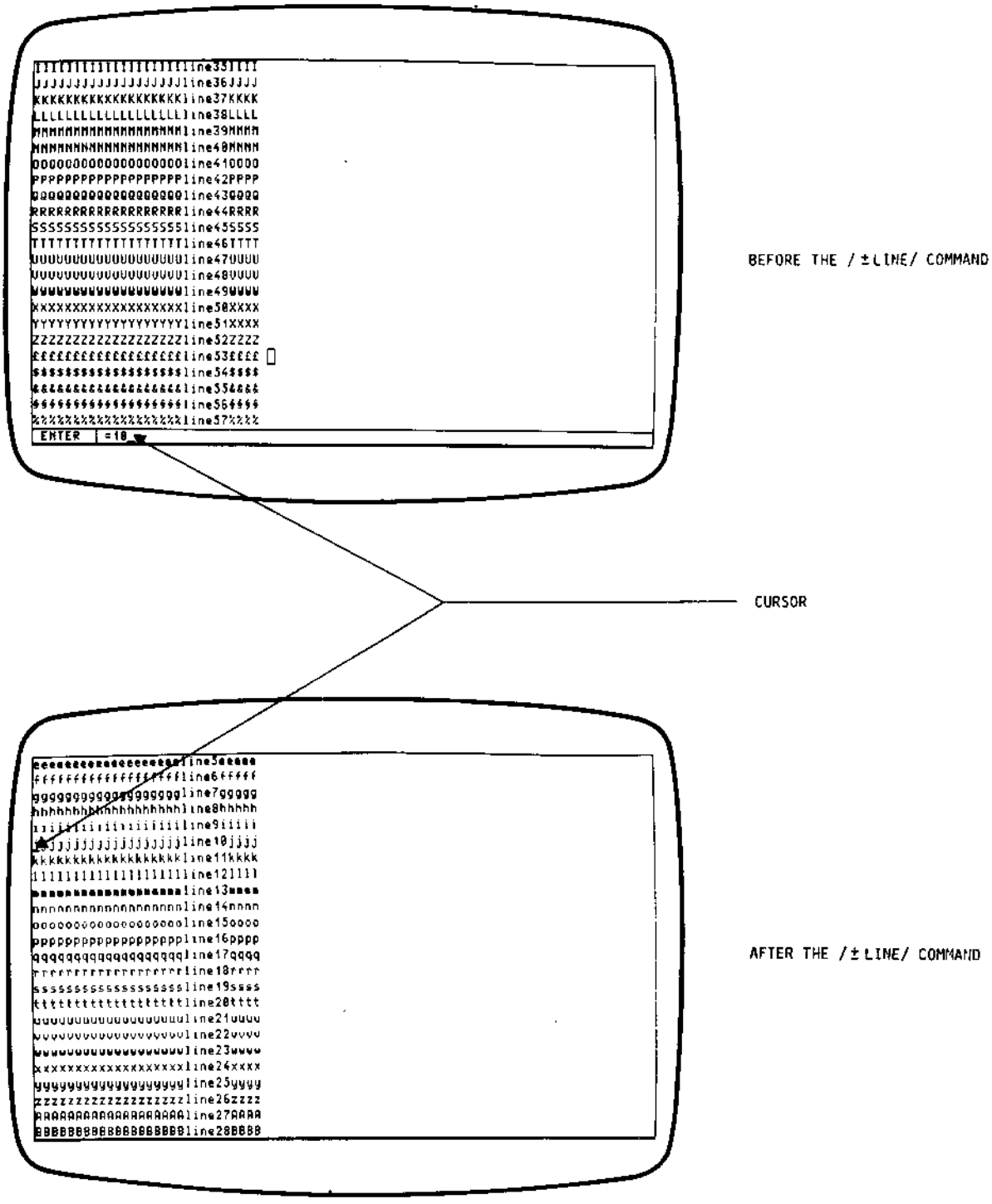


Fig. 3. 3 - Movement of the Cursor to the 'nth' Line

Movement of 'n'
Lines in a Window

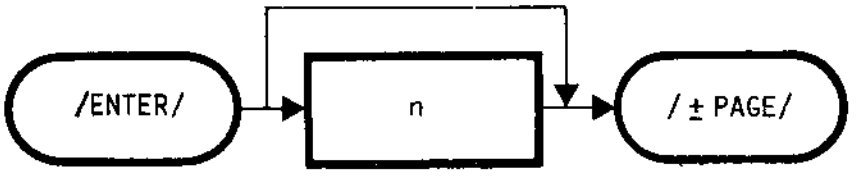
where:

n is a whole number.

If only 'n' is specified, this is the equivalent to entering /+LINE/ or /-LINE/ 'n' times. The window scrolls down or up according to whether /+LINE/ or /-LINE/ is entered.

When '=' is entered before 'n', the window will automatically move to line 'n', displayed in the window. Otherwise it will automatically be scrolled upwards or downwards, positioning line 'n' one fifth of the way down the window (note 1).

Movement of 'n'
Pages



where:

n is a whole number.

Scrolls the window backwards or forwards by 'n' pages depending on whether /+PAGE/ or /-PAGE/ is entered (note 2).

If 'n' is not specified, the beginning or the end of the file is displayed in the window depending on whether /-PAGE/ or /+PAGE/ has been entered and the cursor is placed on the first column of the first or last line respectively.

Notes

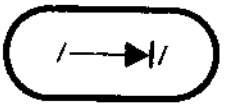
1 If the line containing the cursor is still in the window after movement, the cursor moves with the line, staying in the same position in relation to the text.

However, if an attempt to position the cursor outside the boundaries of the file has been made this will be positioned at the first column of the first or last line, depending on whether the command was /-LINE/ or /+LINE/.

2 The cursor remains stationary except in the case where the command entered indicates scrolling beyond the limits of the file. In such a case the cursor will move to the first character of the first or last line depending on whether /-PAGE/ or /+PAGE/ was entered.

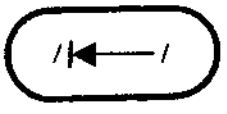
MOVING WINDOWS HORIZONTALLY: /ENTER/, /|←—/, /—→|/

Moving Windows to the Right



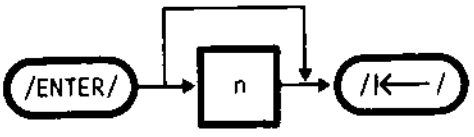
Moves the window horizontally to the right by 1/4 of the screen width (note 1). This movement will only take place if there is at least one character in the last quarter of the screen.

Moving Windows to the Left



Moves the window horizontally to the left by 1/4 of the screen width (note 1).

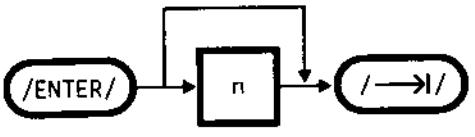
Moving the window to the nth column



where:

n is an integer.

Moves the window to the left until the nth column of the file is the first column of the window. The default value for n is 1 (note).



where:

n is an integer.

Moves the window to the right until the nth column of the file is the first column of the window. The default value for n is (253 - window width) (note).

Note

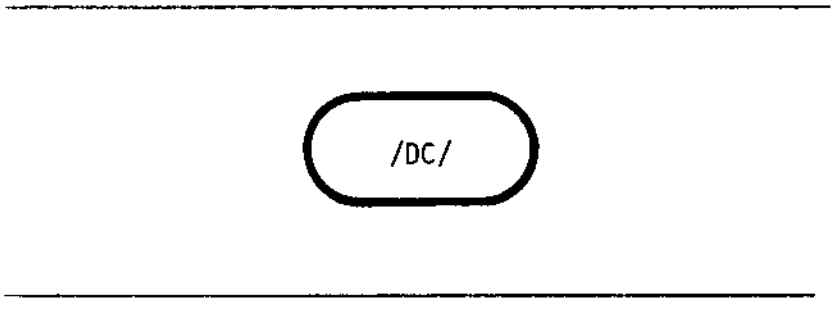
If the column containing the cursor is still in the window after the movement, the cursor moves with the column remaining in the same position with respect to the text.
Otherwise the cursor remains in the same position with respect to the window.

OPERATING ON CHARACTERS: /DC/,/IC/

Appending

This is an intrinsic Editor command.
All the user has to do is position the cursor as required and type in the characters. the characters.

Deletion

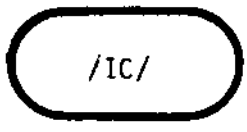


The character on which the cursor is positioned, is deleted and the gap is closed automatically by the characters to the right of it. The cursor does not change its position.

If the whole line is cancelled, there will of course be no gap to close.

Replacement

This is an intrinsic Editor command.
The user has to position the cursor on the character to be replaced and enter the correct key.
After replacement, the cursor moves one position to the right.

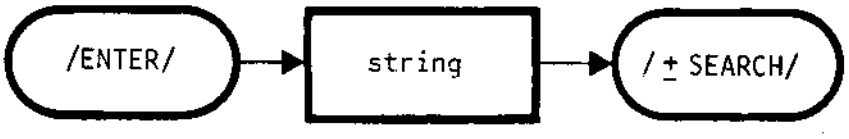


This command brings the Editor into INSERT MODE. The Led L1 above the numeric area and function keys lights up to signal this.

INSERT MODE affects only the writing of text, all other commands may be entered and executed while in insert mode. Characters are inserted to the left of the cursor. The cursor may be shifted to any character space within the window. The part of the line to the right of the character that has been inserted, is moved to the right.

The Editor will remain in INSERT MODE even if the active window is changed.
To Exit INSERT MODE press /IC/.

Search for Strings



where:

string is the character string to be searched for.

The sample string is stored in the search buffer, and deletes the old contents. (notes 1,3). Searching will take place above or below the cursor position depending on whether /+SEARCH/ or /-SEARCH/ is entered.

If the sample string is found in the part of the text that is not displayed, the window will scroll up or down until the string appears one fifth down and 1/4 to the right of the window.

If it is already displayed, the window does not change its position.

If the sample string is not found, the text within the window and the cursor position remain unchanged and the following message appears in the command area:

** PATTERN NOT FOUND **

In either caase the cursor will be positioned on the first character of the string.

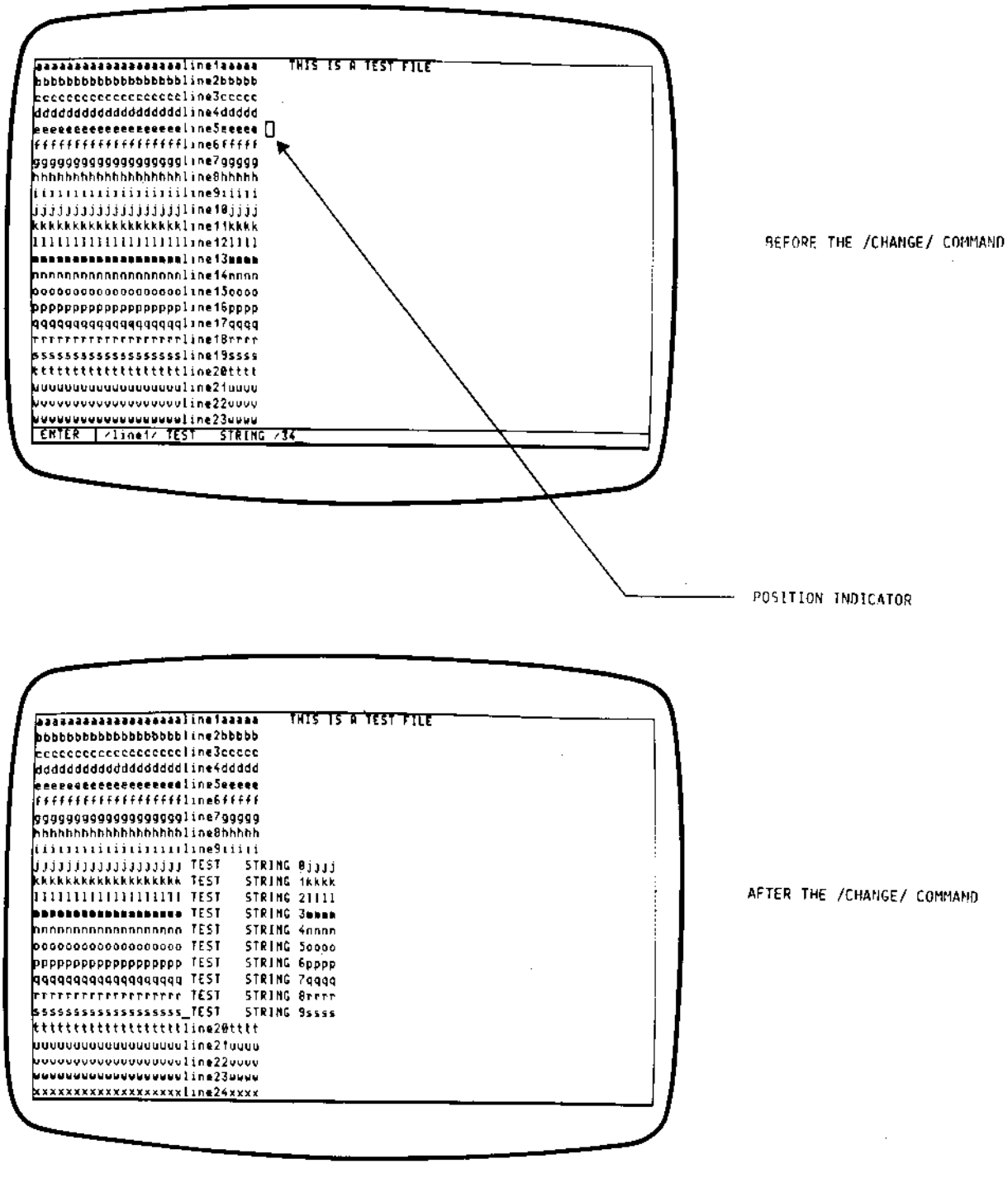
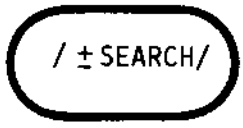


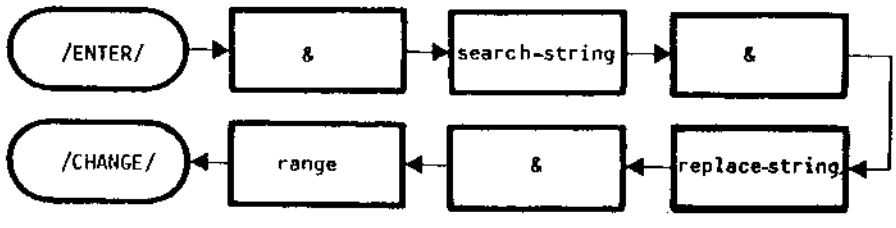
Fig. 3. 4 - Automatic Search and Replacement

Continuation of Search



The command function is the same as the previous one, except that the sample string is taken directly from the search buffer. This command permits searching to continue when the string found by the previous command is not the one being searched for (note 3).

Automatic Search and Replacement



where:

& is a character that does not appear either in 'search-string' or in 'replace-string'. It is used as a delimiter.

search-string is the string to be searched for.

replace-string is the string which replaces the one that has been found.

range is the number of lines which are to be searched. If the character '*' is specified, the file is searched from the current cursor position to the end of the file.

The 'search-string' is stored in the search buffer and the 'replace-string' is stored in the replacement buffer (notes 1,3).

Each time the 'search-string' occurs in the number of lines fixed by the parameter 'range', starting from the current line, it is replaced by the 'replace-string' (note 2).

If 'replace-string' is not specified, each 'search-string' is replaced by a null-string.

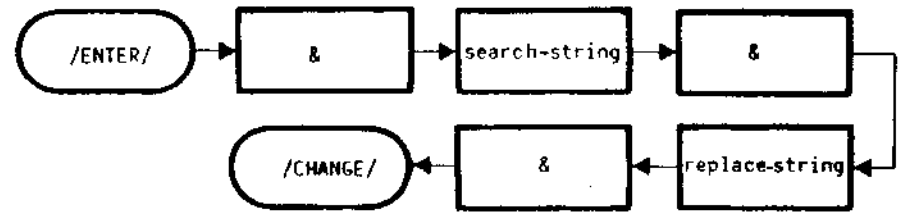
If the last 'search-string' found is not present on-screen, the window is shifted so as to display the string one fifth down and 1/4 across the window. If the 'search string' is already displayed on screen the window is not shifted.

The cursor is moved to the first character of the last replacement that has been made.

If the 'search-string' is not found, the following message is displayed in the system area:

** PATTERN NOT FOUND **

The window and cursor positions are not modified.



The parameters have the same meaning as in the previous command.

This command is used for direct replacement of the string (notes 1,3).

If the cursor is placed on the first character of a 'search-string', then the string is replaced by a 'replace-string'; otherwise sample string is searched for, but no replacement is made.
When another 'search-string' is found, the cursor moves to the first character in it.

If 'replace-string' has not been specified, 'search-string is substituted by a null-string.

If the 'search string' found is not displayed on the screen, the window is shifted so that the string is displayed one fifth down and one quarter across the window. If this string is already displayed no movement of the window is effected.

The user may:

- enter /CHANGE/, and to effect substitution with the 'replace string' (note 2).

- enter /±SEARCH/, to continue the search without making replacements.

If no occurrence of 'search string' is found, the following message is displayed in the command area:

** PATTERN NOT FOUND **

in which case the window and cursor positions remain unchanged.

Notes

1. The command area has 70 character spaces, so the group of parameters cannot be longer than this.

2. If the string searched for is substituted by a longer string this will automatically extend to the right of the screen line.

   If the line is longer than 253 characters after a replacement has been made, the following message is displayed:

   ** LINE TOO LONG **

   and searching is suspended. The cursor moves to the string in question after the command area has been removed by entering /RETURN/. In the case of substitution by a shorter string, it will be compacted to the left.

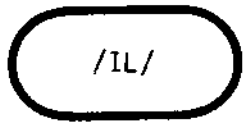
3. The search and replacement buffer is not modified when the active window is changed. This allows search and replacement operations to be effected between different files or different parts of the same file displayed on more than one window.

OPERATING ON LINES: /ENTER/,/DISP/,/IL/,/DL/,/PICK/,/PUT/,/MOVE/,/NUM/

Appending of Lines

This is an intrinsic Editor command. The user must move the cursor to the correct position and type in the required characters.

Insertion of a Line



Inserts a blank line above the cursor position (note 1).

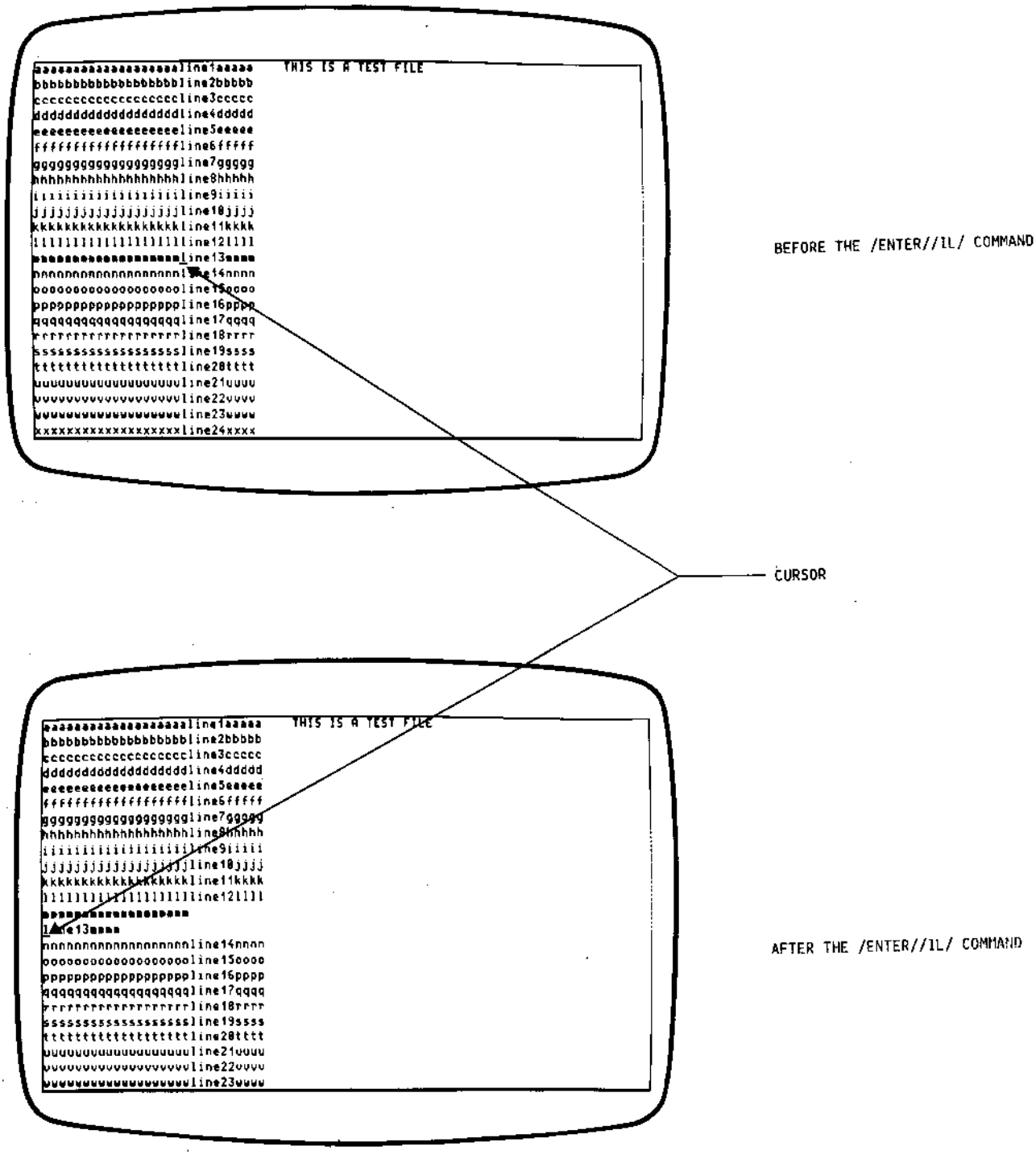
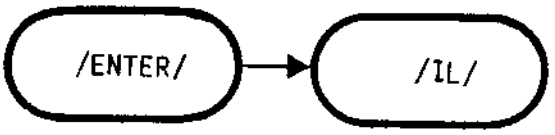


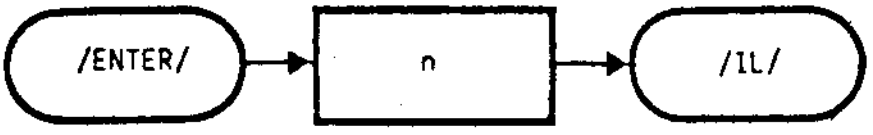
Fig. 3. 5 - Splitting a Line

Splitting a Line



The current line is divided in two at the point where the cursor is currently positioned. The right portion is inserted below the current line as a line in itself.

Insertion of 'n' Lines



where:

n is a whole number.

Inserts 'n' blank lines starting from the cursor position, scrolling the text down.
The cursor is positioned on the first line inserted (note 1).

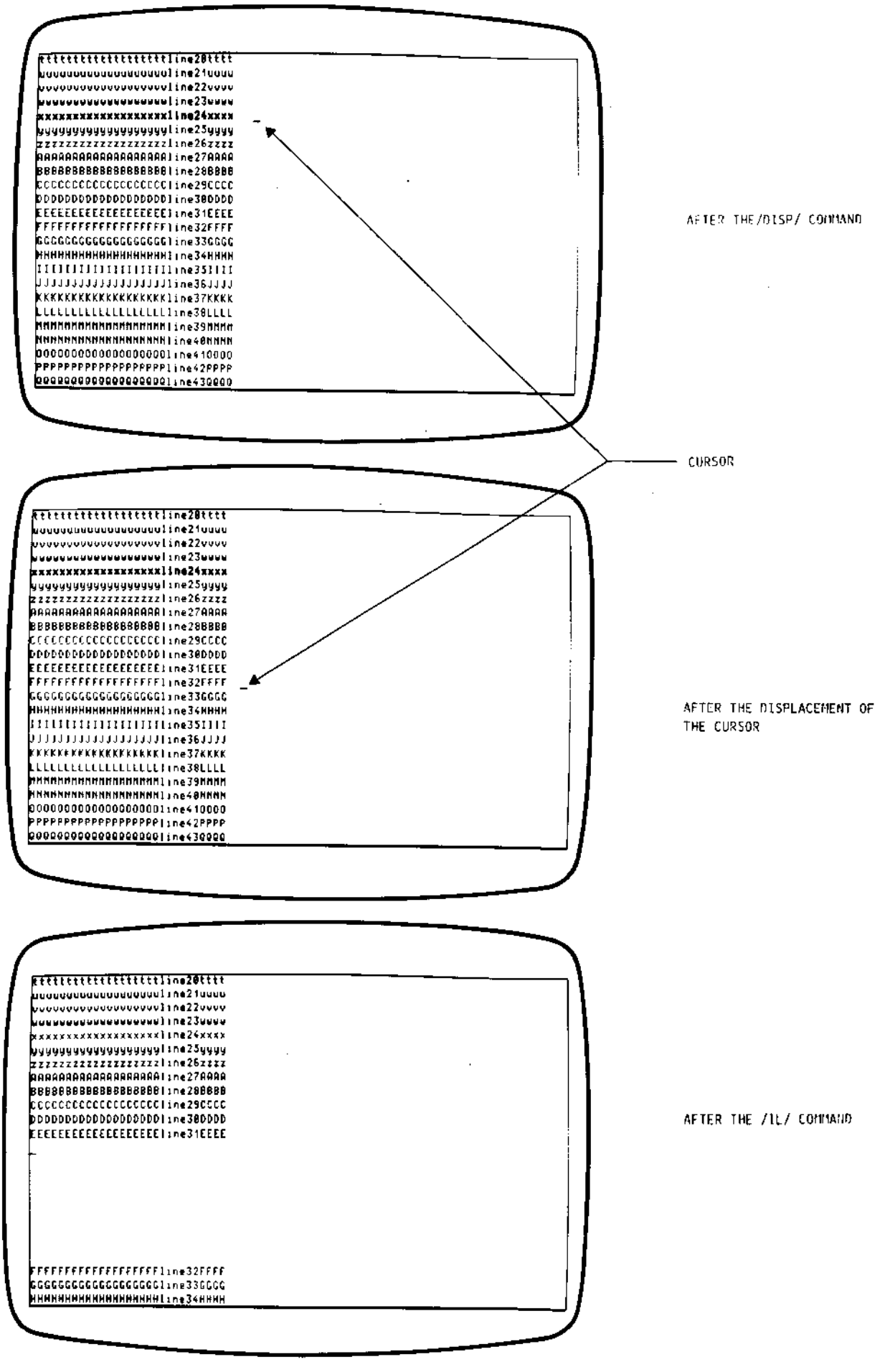
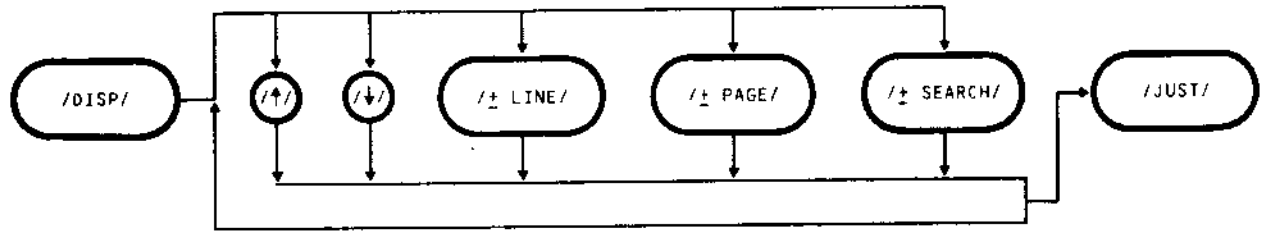


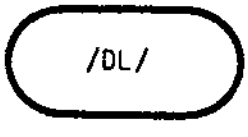
Fig. 3. 6 - Insertion of Lines as Indicated by the Cursor

Insertion of Lines as Indicated by the Cursor



Inserts the number of blank lines which exist between the first line (i.e. at the line on which the /DISP/ key is entered) and the last line (i.e. the line on which the /IL/ key is entered) both lines included. These lines are inserted above the last line, and the cursor is positioned on the first one, upward scrolling of the existing text takes place automatically.

Deletion of a Line



The current line is deleted (note 2).

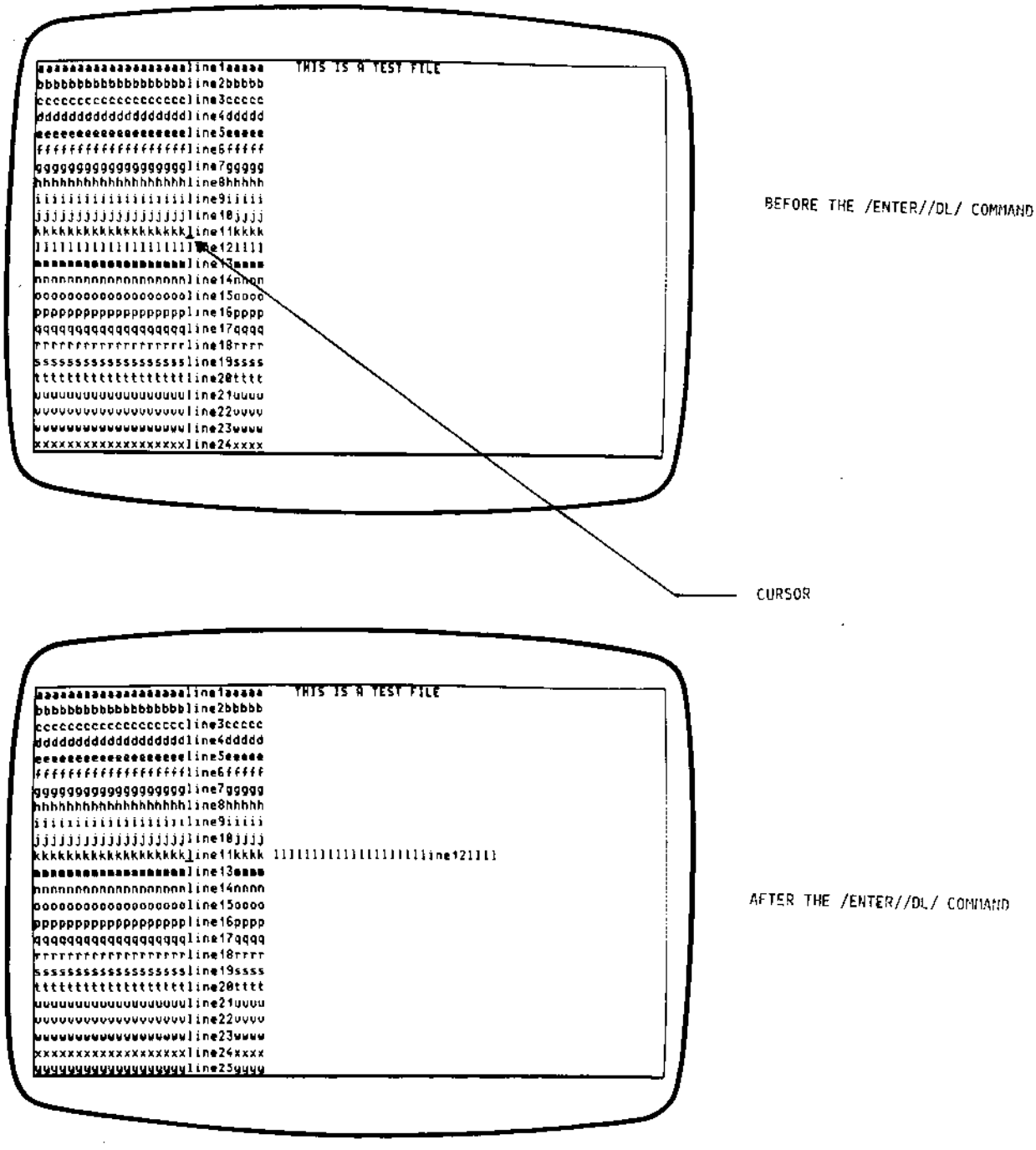
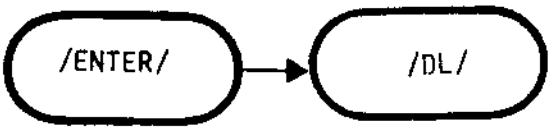


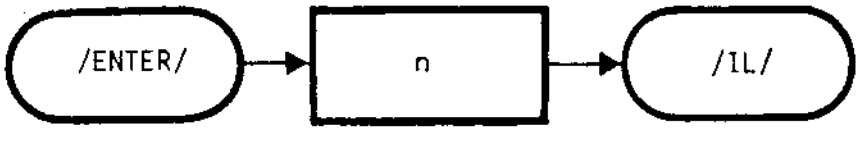
Fig. 3. 7 - Joining two Consecutive Lines

Joining two Consecutive Lines



The current line and the one following it, both of which must be visible in the window, are joined in one line, with a blank character inserted between the two, provided that the sum of the two lines is not greater than 253 characters.

Deletion of 'n' Lines



where:

n is a whole number.

'n' lines are deleted starting from the current line. If the specified number of lines is greater than the number of lines to the end of the file, only the lines in the text are deleted (note 2).

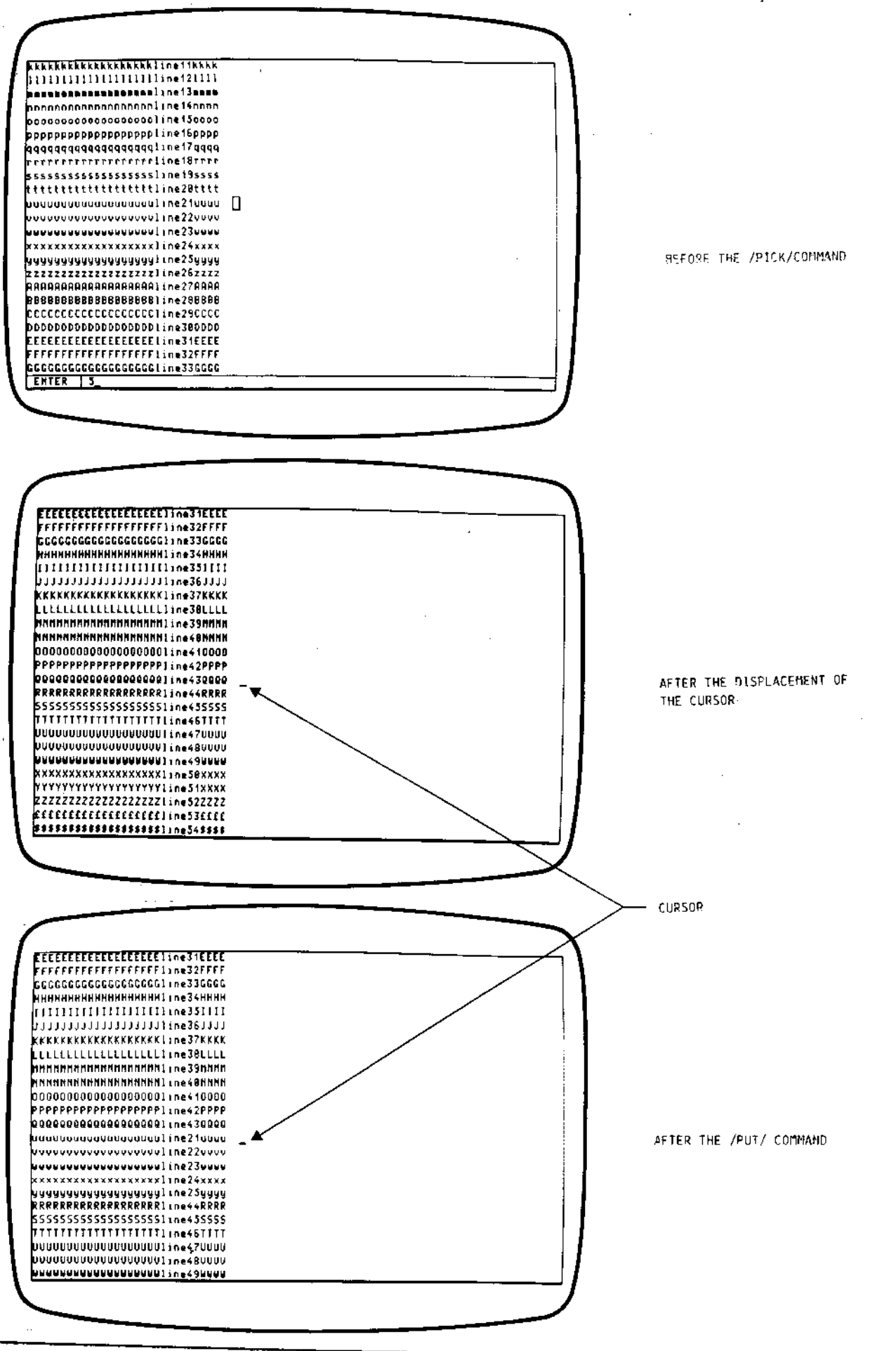
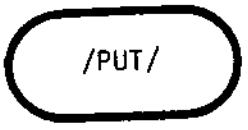


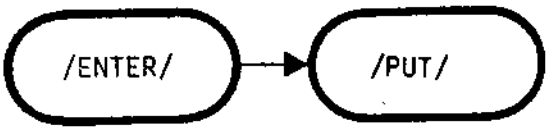
Fig. 3. 8 - Recovery of Stored Lines

Recovery of
Stored Lines



All the lines whose positions have been stored by a /PICK/ command or by an equivalent compound command are inserted after the current line (notes 3,5).

Recovery of
Deleted Lines

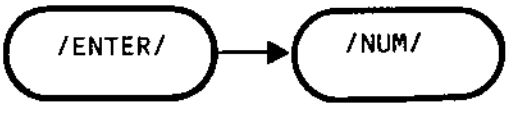


All the lines that have been cancelled by a /DL/ command or by an equivalent compound command are inserted in their original place.
The lines are unrecoverable ilf after the deletion a /SAVE/ command has been executed, another file opened, or the window has been changed (note 3).

Recovery with Deletion

All lines whose position has been stored via the /PICK/ command or an equivalent compound command, are inserted after the current line. The original lines are deleted (notes 3,5).

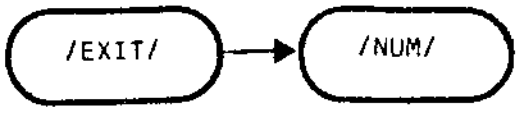
Numbering of Lines



Displays the line number of each line currently displayed in the window. The line number appears in the first 6 character spaces (5 characters and a blank) and the text is automatically shifted to the right by the same number of character spaces.

Numbering is merely visual and does not cause any modification of text. It is inserted only in the left margin of the file and is thus visible only if the first column is displayed.

Deleting line numbers



All line numbers are deleted. The text is automatically shifted to the left by six characters.

Notes

1 The cursor is positioned on the first character of the first line of the text which has been inserted.

2 After each deletion, the cursor stays in the same position in relation to the window and the space in the text is closed automatically.
The lines inserted are saved and, may be re-inserted.
The capacity is 32,500 lines that is a whole file.

3 The cursor is moved to the first line re-inserted. If no line positions are stored or lines cancelled, the following message is displayed:

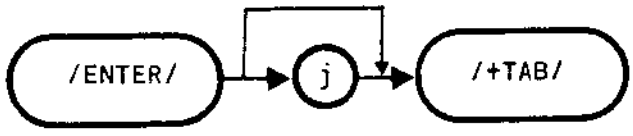
** NO LINES DELETED/PICKED **

4. The position of lines previously stored by another /PICK/ or equivalent compound command are lost. The position of the lines stored by a /PICK/ or equivalent compound command are deleted if the text is modified before a /PUT/ or /MOVE/ command is entered.

5. The position of the lines stored by a /PICK/ command or its equivalent remains unaltered if the current window is changed. This allows the transfer of lines from one file to another or between different parts of the same file displayed in more than one window.

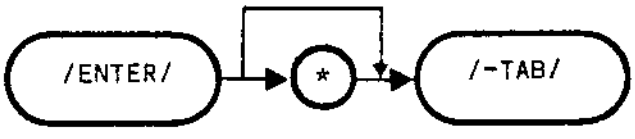
TABULATION: /ENTER/,/±TAB/

Defining a tabulation stop



Inserts a tabulation stop on the column in which the cursor is positioned. If "j" is specified, the space between the tabulation stop and the previous one is defined as a "reserved interval" (note). It has a special meaning only in JUSTIFY MODE (see the paragraph JUSTIFICATION) and is otherwise considered as a normal tabulation stop.

Removing a Tabulation Stop



Removes the tabulation stop on the column where the cursor is currently positioned. If '*' is specified, all tabulation stops are removed (note).

Note 1 All characters other than "j" or "*" are ignored.

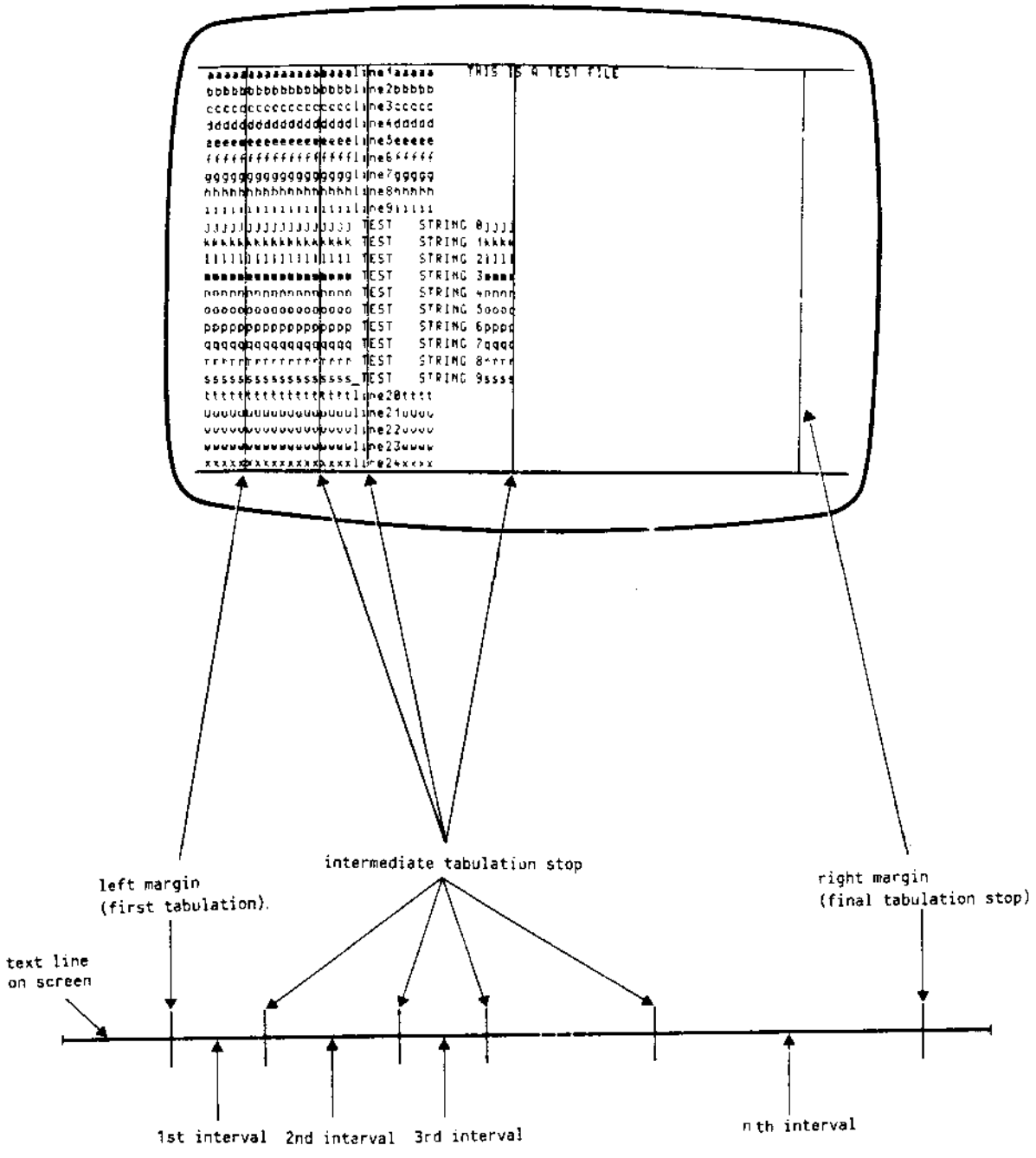


Fig. 3. 9 - JUSTIFY MODE: The Screen

JUSTIFYING TEXT: /ENTER/,/EXIT/,/DISP/,/JUST/

These commands allow text to be aligned with two or more defined points. Text is justified by varying the number of spaces between words and by transferring words from one line to the next.

The Editor is able to justify text when in "JUSTIFY MODE" a state which is entered and exited by executing specific compound commands which are described later in the chapter. When this state is entered the two vertical lines which fix the boundaries of the window are cancelled. This happens to avoid any confusion with the visual representation of the reference points.
JUSTIFY MODE may be entered only when there is one window on screen. All the editor commands except for those dealing with windows may be called and executed when the Editor is in JUSTIFY MODE.

The reference points for aligning the text are identified by tabulation stops (see the paragraph TABULATION) and are usually represented by vertical lines. These lines are placed to the left of the position of the defined tabulation stop and do not occupy any screen space (see Fig. 3.9).

The set of tabulation stops divides the screen into intervals: if there are no intermediate tabulation stops, there is one interval which is defined by the two tabulation stops set by the user or assumed by default at column 8 and 72 (see Fig. 3.9).
The first and last tabulation stops are known as the 'left margin' and the 'right margin' respectively.

Reserved Intervals

It is possible to define reserved intervals (see the paragraph 'TABULATION'). The words which start at this type of interval and which are contained within it are displayed in bold type.
The position of these words remains unchanged when justifying text: text is justified ignoring the presence of any text within the reserved intervals.
The first and/or last tabulation stops may not be used to define a reserved interval.

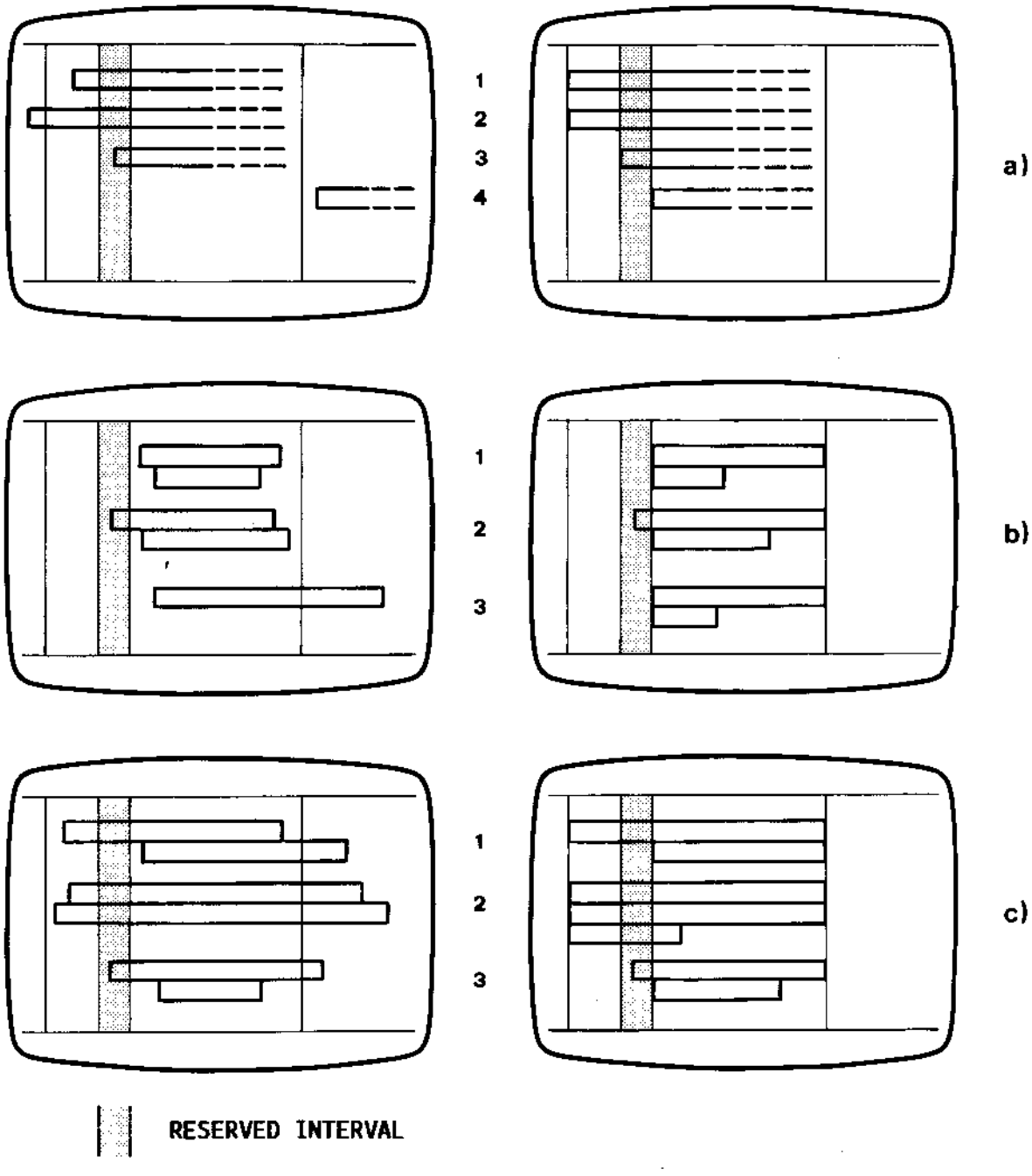


Fig. 3. 10 - Right and Left Justification and Transferring Words

Justifying text may be considered as the set of three operations:

- alignment to the right
- alignment to the left
- transfer of words

Left Justification

The lines which start in a given interval are aligned to the left at the start of the interval itself. (See a1 and a3 in Fig. 3.10).

Those lines which start to the left of the left margin are aligned to the left margin (see Fig. 3.10 a2).

Those lines which start beyond the right margin are aligned to the left at the start of the first interval.

Right Justification

Each line is aligned at the right margin if it satisfies at least one of the following conditions

- it starts in an interval which is not reserved and the next line starts in the same interval.
- it starts in a reserved interval and is followed by a line which starts in the next interval (see Fig. 3.10 b2).
- it goes beyond the right margin (see fig. 3.10 b3).

If a word is longer than the number of character spaces which exist between the left and right margins, it will be aligned to the left.

Transferring Words

Transferring words between two consecutive lines A and B is possible if:

- line B does not start in a reserved interval (see Fig. 3.10, c1, c2, c3).
- line A starts in an interval which is not reserved and line B starts in the same interval (see Fig. 3.10, c2).
- line A starts in a reserved interval and line B starts in the next interval which is not reserved (see Fig. 3.10, c3).

The transfer of words between consecutive lines may cause the creation of a new line which is aligned with the same tabulation stop as the preceding lines.

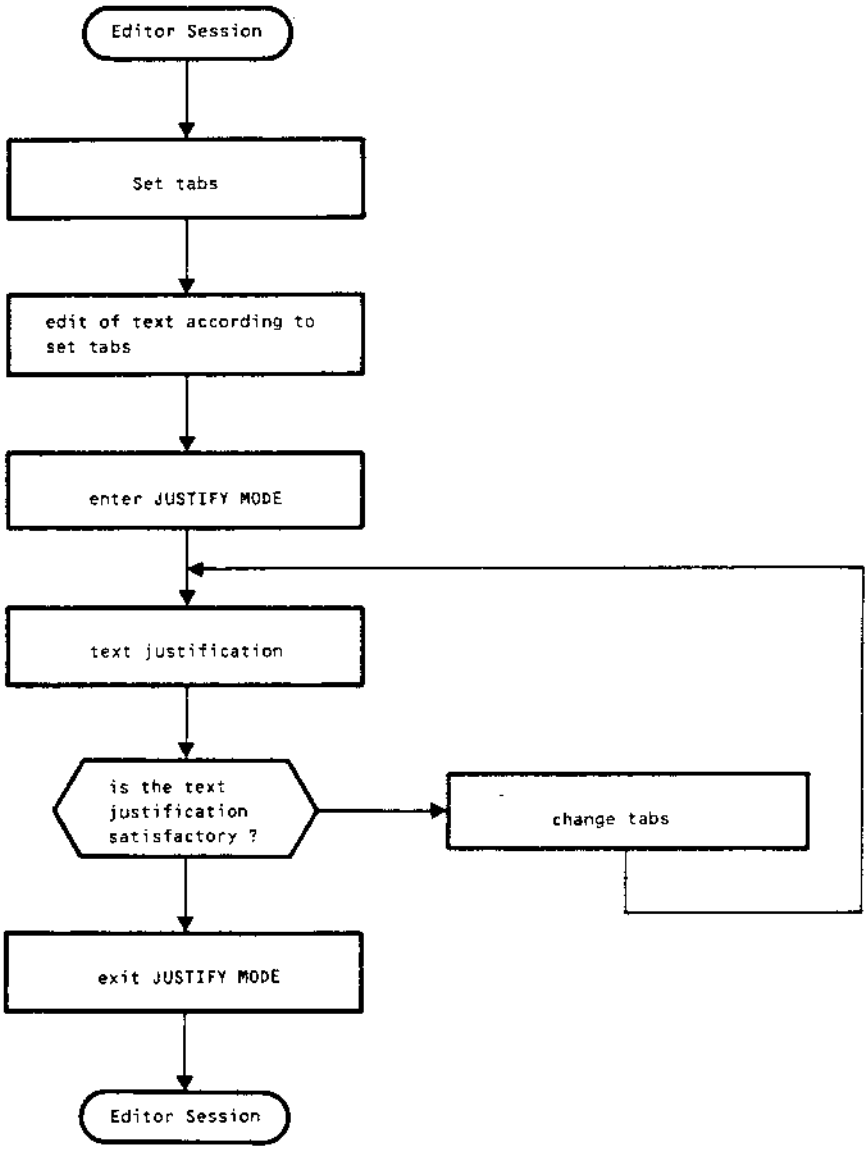


Fig. 3. 11 - Using JUSTIFY MODE

Using JUSTIFY MODE

Figure 3.11 gives a flow chart which illustrate the correct use of this function.

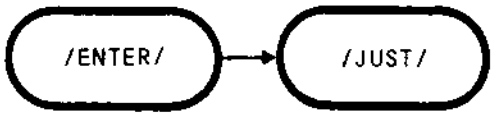
The use of tabulation is retained useful when editing a new text.

The text may be justified only after entering JUSTIFY MODE. If the text justification is not satisfactory, the tabulation may be reset and the text justified again.

The tabulation may be reset in various parts of the text (see the example illustrated in Figs. 3.13, 3.14). The right text margin may thus vary throughout the text.

Text justification may also be used to render programs, written in languages which allow the free use of blanks (Pascal,MCL...) easy to read.

Entering JUSTIFY MODE



This command will cause the Editor to switch to JUSTIFY MODE. The two vertical lines defining the window are cancelled and the tabulation stops which have been set are displayed in the form of vertical lines (see Defining a Tabulation Stop).

All editor commands except for those dealing with windows may be called and executed.

If more than one window is open on a file or different files at the time the command is entered, or one of the window commands is entered when already in JUSTIFY MODE, the following error message is displayed:

** INVALID OPERATION **

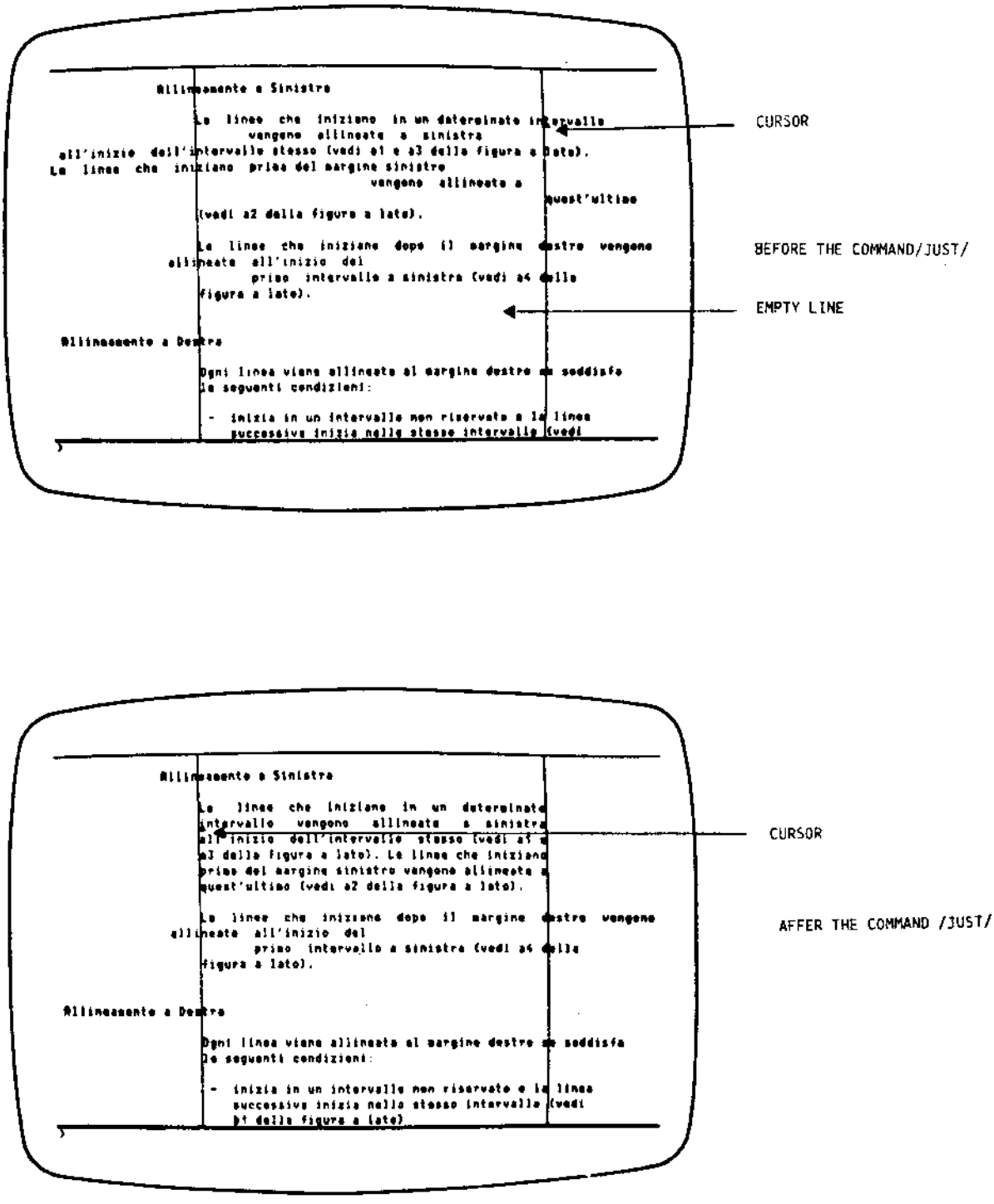
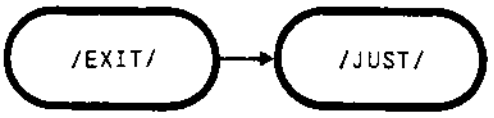


Fig. 3. 12 - Justifying Lines

Exit JUSTIFY MODE

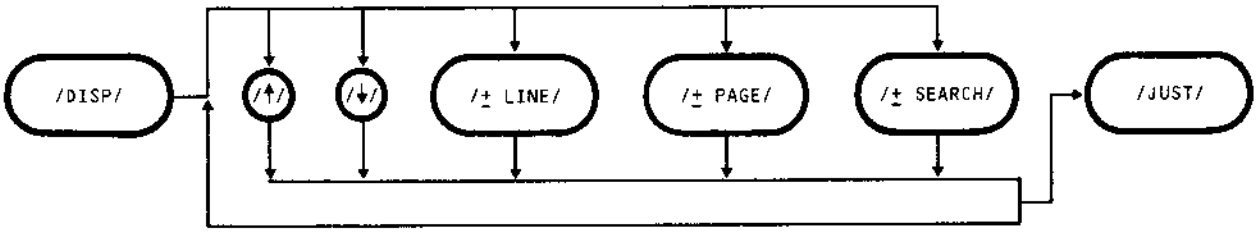


This command must be entered to Exit JUSTIFY MODE. All vertical lines corresponding to tabulation stops are cancelled and the two vertical lines which fix the window boundaries are displayed.

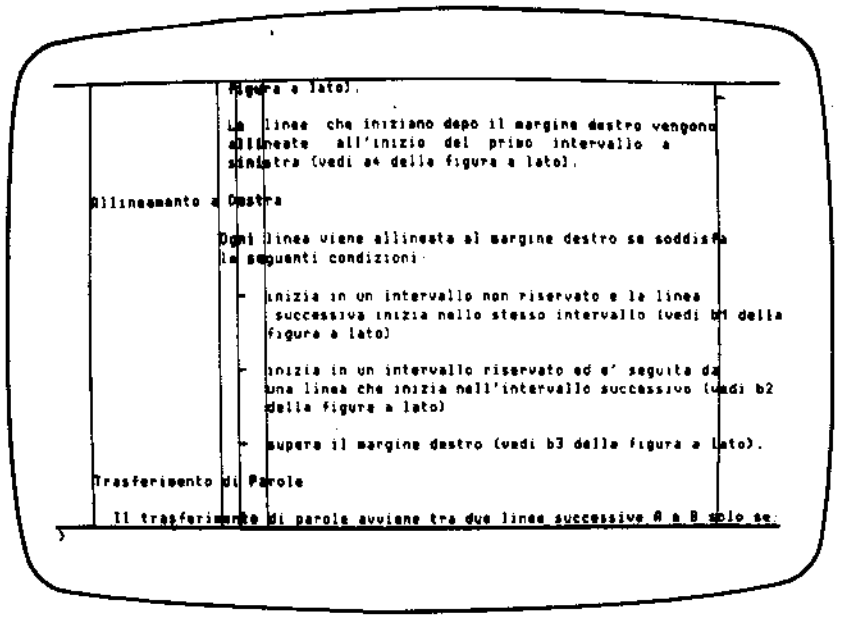
Justifying Lines

This command justifies all text lines starting from the current line until the first blank line or the end of the file (see notes 1 and 2).

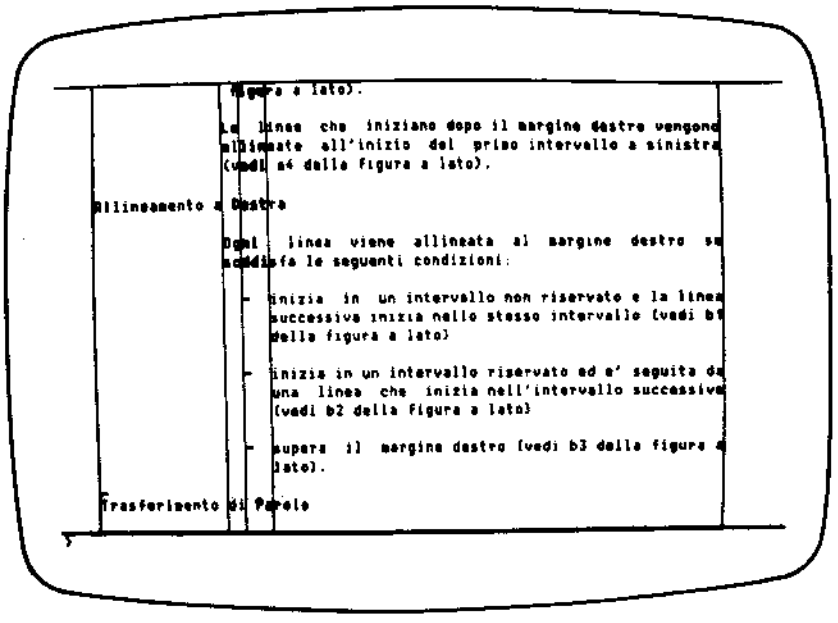
Justifying Lines as Indicated by the Cursor



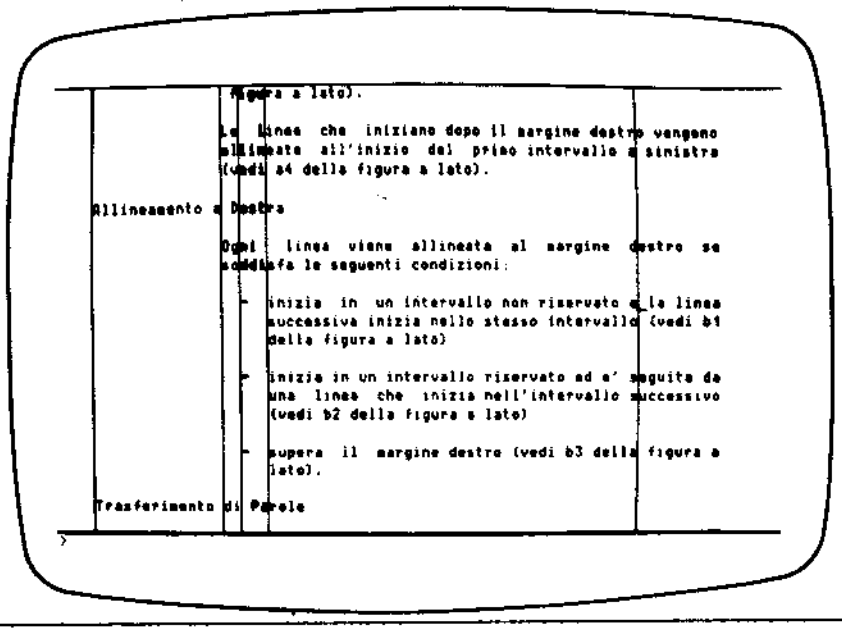
This command justifies all the text lines between the first line (i.e. the line at which /DISP/ was entered) and the last line (ie the line at which /JUST/ was entered) both lines included (see notes 1 and 2).



a)



b)



c)

Fig. 3. 13 - Example

Notes

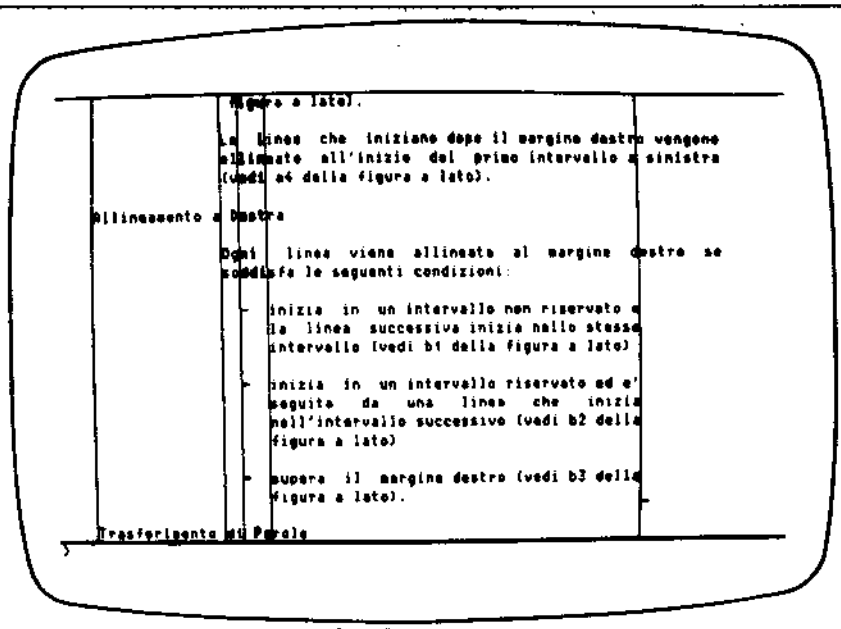
1. If the tabulation stops have not been set columns 8 and 72 will be assumed by default.

2. This command may only be executed in JUSTIFY MODE. If otherwise used the following error message is displayed:

   ** INVALID OPERATION **

Example — Assuming that a text has been edited and JUSTIFY MODE entered, text is justified as shown in Fig. 3.13(b) using the tabulation shown in Fig. 3.13(a), justification has been modified as follows:

cursor movement	the cursor is shifted to the right margin
/ENTER//-TAB/	the right hand margin is cancelled.
cursor movement	the cursor is shifted to column 65.
/ENTER//+TAB/	the new right hand margin is fixed at column 65 as shown in c).
/DISP/	the current line is displayed in boldface.
cursor movement	the cursor is shifted 9 line below
/JUST/	the text is justified as shown in the following figure.



d)

Fig. 3. 14 - Example (continued)

APPENDIX A. EDITOR MESSAGES

EDITOR : INTERRUPT REQUESTED

A request to abort the Editor session has been made due to an error or via a KILL command. Before effecting the operation the Editor removes the "work-file". Following this, the situation of the files on disk remains the same as it was after the last partial SAVE effected.

*** DEVICE NOT READY ***

The device specified in the command is not used correctly.

If the device used is the hard disk, it is possible that:

- the hard disk drive is switched off (valid only for a self-standing hard disk)
- the hard disk drive is disconnected (valid for integral and self-standing hard disks)

Whereas, if the device is a unit supported by a floppy disk the following may be verified:

- For a self-standing or integrated floppy disk unit.
  - . the floppy disk has been inserted in the drive with the wrong side up
  - . the door of the drive has not been closed
- For self-standing floppy disks

- the floppy disk unit is switched off
- the floppy disk unit is disconnected

The user must manually effect the necessary controls for the correct use of the device, operating on the device itself.
After these controls have been made the command may be repeated.

/ENTER/ The command which has had this message returned is repeated.

/EXIT/ The command which caused the printing of this message will not be executed. The Editor takes over control and will decide either to return to the condition prior to the entry of this command, or, to abort the entire Editor session.

** ERROR IN OPEN TAB FILE **

The tabulation file does not exist or is not in the directory specified.

/EXIT/ The command which causes the display of this message will not be executed.

*** ERROR SYSTEM ***

Editor system errors.

/ENTER/ The command which has had this message returned is repeated.

/EXIT/ The command which provoked the display of this message will not be executed.
The Editor takes over control and will decide either to return to the condition prior to the entry of this command, or, to abort the entire Editor session.

*** FILE ALREADY OPEN ***

The file on which you are trying to open a new Editor session or the file on which you are trying to save the active file has already been opened by another user, using the "lock option".

/ENTER/ The command which has had this message returned is repeated.

/EXIT/ The Editor returns to the condition it was in prior to when the command which caused the transmission of this message was entered.

** FILE INCOMPLETE **

This appears at the opening of a file to warn the user that there is a long sequence of characters with respect to the size of the file which does not contain the hexadecimal configuration '0A' (Line Feed).
If the user carries on with the session, the section of the file between the line feed and the end of the file is lost.

/RETURN/ The command area is removed and the system remains in the Editor environment.

/EXIT//ABORT/ This closes the active file. The original file remains unchanged.

** FILE MODIFIED **

An abort request has been made after the file was modified.

/ENTER/ The session is aborted.

/EXIT/ The command which causes the display of this message will not be executed.

** FILE NOT BYTE-STREAM **

The file you are trying to open is not a byte-stream file and cannot therefore be edited.

/ENTER/ The command which has had this message returned is repeated.

/EXIT/ Closes the active file. The original file remains unchanged.

** FILE TOO LARGE **

The file specified at the opening of an Editor session or of a window on another file has more than 32,500 lines or too many characters. It is not possible to edit a file which is so large.

/RETURN/ The command area is removed and the Editor restores the situation existing prior to the command being entered.

*** HARDWARE ERROR ***

Hardware error.

/ENTER/ The command which has had this message returned is repeated.

/EXIT/ The command which provoked this message will not be executed. The Editor takes over control and will decide either to return to the condition existing prior to the entry of this command, or, to abort the entire Editor session.

** INITIALISATION FAILED **

A system Editor has occurred during an attempt to open an Editor session; initialisation is aborted.

You may try to reopen the session.

** INVALID OPERATION **

You have entered an illegal compound command.

This message also appears when the name or pathnane specified in the close, save or partial save command is the name of one of the files still open in the Editor session.

/RETURN/ The command area is removed and the situation, existing prior to the entry of the command which caused the error, is restored.

** INVALID PARAMETERS **

When entering the command to open the Editor session you have specified a number of invalid parameters. You are still in the Shell environment.

Enter the command correctly.

*** INVALID PATHNAME ***

The pathname that you have specified in the compound command is incorrect:

- the structure of the pathname given is incorrect for one or more of these reasons:
  - . its length is greater than 60 characters
  - . the length of its intermediate components is greater than 14 characters
  - . the length of its last component is greater than 12 characters
- a directory or a volume name which does not exist appears in the pathname.

/ENTER/ The command which has had this message returned is repeated.

/EXIT/ The command which caused this message to be emitted will not be executed. The Editor takes over control and returns to the condition prior to the entry of this command.

** INVALID SYNTAX **

A syntax error has been made when entering the arguments of a compound command.

/RETURN/ The command area is removed and the Editor returns to the condition it was in before the command that caused the error was entered.

** LINE aa COL bb FILENAME cc..c **

Appears after entering the compound command /ENTER/ /DISP/, showing the file name or the pathname 'cc..c', the line 'aa' and the column 'bb' where the cursor is positioned.
If the pathname is longer than 47 characters, it will be truncated starting from its root, which will be substituted by the characters '??' .

/RETURN/ The command area is removed and the Editor returns to the previous state.

** LINE TOO LONG **

The string resulting from the execution of the search and replace, or replace, or linking of two lines compound command, is greater than 80 characters.

/RETURN/ The command area is removed and the system switches back to the condition in which it was prior to the entry of the command which caused this error. The cursor is positioned at the start of the string which caused the error.

This message may also appear when opening an Editor session to warn you that the lines of the file you have opened have more than 80 characters.
You are advised to exit out the Editor session without executing any commands other than save or close.
Entering commands in this session will cause the substitution of certain characters in the file by the hexadecimal configuration '0A' (Line Feed).

/RETURN/ The command area is removed.

/EXIT//ABORT/ Closes the active file aborting the entire Editor session.
The original file remains unchanged.

*** NAME ALREADY EXISTENT ***

You are trying to save the active file with a name which already exists on disk.

/ENTER/ The active file replaces the existing file.

/EXIT/ The command is aborted.

** NEW FILE ? ** The name of the file, specified in the command which opens the Editor session or that which opens a new window, does not exist. This means that:

- you want to edit a new file
- you have entered the file name or pathname incorrectly.

/ENTER/ When this is a new file. The Editor session opens on the new file.

/EXIT/ When an error has occurred. The command is aborted.

** NO LINES DELETED/PICKED **

This appears after a /PUT/, /ENTER/ /PUT/, /MOVE/ command, in all cases where there are no deleted lines or lines which have been stored through the /PICK/ command.

/RETURN/ The command area is removed and the Editor returns to the condition it was in prior to when the command that caused the error was entered.

** NO MODIFICATION **

The file you are trying to save has not been modified since the last save operation.

It is thus useless to carry out this operation since the active file is identifical to the previous version stored on disk.

In reponse to this message the user may press:

/ENTER/ to execute the command which provoked the display of the message.

/EXIT/ to abort the command which provoked the display of the message. If this command is:

- partial save, the Editor returns to the state it was in prior to the entry of the command
- close session, the Editor closes the session without modifying the file versions on disk. In this case the command entered has the same effect as /EXIT//ABORT/.

*** OUT OF DISK SPACE ***

This may appear after:

- the request to execute a storage command
- attempting to open the 'work-file' to enter into the Editor session
- attempting to open the 'save-file' to save the active file.

The user may operate on the system lines to free a memory area on disk by removing directories or files which are nc longer needed.

To operate from the system lines you must enter:

- /DEL/ The system line is activated.
- /CH.WIN/ The cursor is displaced from the video

to the system line and MCL commands may be entered.

The MCL commands which remove a file or directory are REMOVE and CLEARDIR (see the MOS COMMANDS - Reference Manual).

After having done this you may repeat the command.

/ENTER/ The command which has caused this message to be sent is repeated.

/EXIT/ The command which caused the printing of this message will not be executed. The Editor will return to the state it was in before the command was entered.

** PATTERN NOT FOUND **

The string being searched for is not found in the specified part of the text.

/RETURN/ The command area is removed and the Editor returns to the condition it was in prior to the entry of the command which caused the error.

*** SYSTEM ERROR ***

System error.

/ENTER/ The command which has had this message returned is repeated.

/EXIT/ The command which caused the display of this message will not be executed. The Editor takes over control and will decide either to return to the condition existing prior to the entry of this command, or, to abort the entire Editor session.

** TOO MANY LINES IN FILE **

Resulting from the execution of any command which adds on new lines, the total number of lines in the file is more than 32,500 which is the maximum number of lines allowed in a file.

/RETURN/ The command area is removed and the Editor returns to the state it was in before the command that caused the error was entered.

*** TOO MANY FILES ***

There are too many files opened in the system and it is not possible to open any more. This appears:

- at the opening of a session when the Editor tries to open the file to be edited
- when the Editor tries to open the "work-file"
- during a save operation when the Editor tries to open a "save-file".

You may ask other users to close some of their files and then repeat the command.

/ENTER/ The command which has had this message returned is repeated.

/RETURN/ The command is aborted and the Editor returns to the state it was in before the command which caused the transmission of this message was entered.

** YOU MUST SAVE THE FILE **

The file must be saved by exiting from the session, because the Editor has to organise its "work-file". This appears during the Editor session, when the session has gone on for too long.

/RETURN/ The command area is removed and the Editor will only accept the closing commands of the session itself.

/EXIT//SAVE/ The Editor session is closed and the file is saved under its own name.

Instead of /EXIT//SAVE/ any of the other closing commands of the Editor session may be entered as required.

Code 4002440 P (4)

Printed in Italy