TV EDIT

```
00000  00000   0000       //
  0    0       0         //
  0    000     0        //
  0    0       0       //
  0    00000   0000   //
```

```
0000     0    00000    0     0     0  00000  0000   000    0
 0  0   0 0     0     0 0    00   00  0       0  0   0    0 0
 0  0  0   0    0    0   0   0 0 0 0  000     0  0   0   0   0
 0  0  00000    0    00000   0  0  0  0       0  0   0   00000
0000   0   0    0    0   0   0     0  00000  0000   000  0   0
```

```
TTTTTTTT  VV      VV          EEEEEEE   DDDDD      IIII  TTTTTTTT
   TT     VV      VV          EE        DD   DD     II      TT
   TT     VV      VV          EE        DD    DD    II      TT
   TT      VV    VV   =====   EEEEE     DD    DD    II      TT
   TT       VV  VV            EE        DD    DD    II      TT
   TT        VVV              EE        DD   DD     II      TT
   TT         V               EEEEEEE   DDDDD      IIII     TT
```

A USER'S GUIDE TO TEC/DATAMEDIA TV-EDIT
****

by Pentti Kanerva

Institute for Mathematical Studies
in the Social Sciences

Stanford University
Stanford, California

July 1973

Revised October 1974
Last lesser editing 6-Oct-75

HINT: The following commands allow you to read this manual using TV-Edit ("$" stands for typing the 'ESC' or 'ALT MODE' before typing the command character(s)):

```
     $W    to read the next window,
     $-W   to read the previous window,
     $10G  to go to page 10, and
     $$F   to finish (notice, TWO ESCs and an F).
```

To suppress printing, type (CRLT)-O (useful on slow terminals).

CONTENTS
----

^L

0. BACKGROUND

TV-Edit is a page-oriented text editor. It was originally written for the IMSSS/Stanford AI Project PDP-1 computer and Philco displays by Brian Tolliver in 1965(?). Four of its "descendants", all written for the PDP-10, are currently in use at Stanford: two at the AI Project use the DATA-DISC display system; one at IMSSS (by John Prebus, since 1970) uses the IMLAC PDS-1 display computer; and the present one at IMSSS and SUMEX (by Pentti Kanerva, since 1972) with versions for TEC Series 400 and DATAMEDIA Elite 2500 displays.

TV-Edit provides a window into a user's text file, with a section of the file (23 lines on TECs and DATAMEDIAs) displayed at once. Commands exist for moving this window to different parts of the file and for altering text in the current window. When an editor command is typed, the window is immediately updated to show the effect of the command on the text.

While editing, the user need not to concern himself with writing the current window, or some internal text buffer, back on the physical records of the file. These manipulations are done automatically by the editor program. 'Finish' is the only command where physical file updating is explicitly specified by the user.

Particular attention has been given to the amount of typing necessary to perform an editing function. Many commands require the typing of just one character. Frequently used sequences of commands and/or text can be defined as strings (macros), and reproduced by typing only the string call character.

As a rule, commands are entered "in the dark", i.e., the command characters themselves are not displayed as they are typed in. However, once a command is completely entered, its effect on the file is immediately visible. This instant feedback is invaluable in learning to use the editor and in assuring the user that the right things are happening to his file.

TV-Edit Documentation

The present manual, <DOCUMENTATION>TVEDIT.MANUAL describes how to use TEC/DATAMEDIA TV-Edit. Chapters 1-3 give the basics of TV-editing, Chapter 4 deals with the more advanced features of the editor.

Recent changes are reported in TV News or Updates, available through option "*" of the editor, or in <DOCUMENTATION>TVEDIT.UPDATE No implementation manual for the editor exists.

^L

Assumptions of This Manual

     Basic knowledge of how to use the timesharing system is assumed. At the least you must know how to log in and have a general idea of what a file is. In addition, we assume that you want to do one of the following using the editor: (a) create a new file and write a few lines of text in it; (b) modify an old file (correct a mispeling, for instance); or (c) read an old file (the TV-Edit news file, for instance).

^L

1. STARTING AND FINISHING

Starting

To use the editor, you must log in on a TEC or DATAMEDIA terminal and call for the editor program. Calling and starting the editor is explained below by means of an example. We assume that you want to create a new file, give it the name "FILE1", and start writing in it. In the example computer dialog is indented, 'CR' stands for the (Carriage) Return key, 'ESC' for the Escape key (labeled "ALT MODE" or "ENTER" on some keyboards), and your replies are indicated by an underline. To start, type

```
            ^C
            @TVEDIT 'CR'
             -----------
```

i.e., you should type the word "TVEDIT" and then the RETURN key.

```
                       T V - E D I T

            ( ?  for help,  ^  to start over)

            (File name, *, or :) FILE1 'CR' [New File] 'CR'
                                 ----------            ----
```

You can enter the file name using the TENEX file name recognition feature. The file must be a disk file. New files are created in the connected directory only.

The second 'CR' confirms the file name, but it also lets you review the standard options. For a fast start with standard options, confirm with an 'ESC' instead.

In place of the file name, you can type "*" to read the latest News, ":" to read this manual, or "?" to get a help message.

When the editor is restarted and a file name appears, followed by two slashes, as in

```
            (File name, *, or :) FILE1.;1//,
```

you can either accept the file by typing 'CR' or 'ESC', or reject it by typing the name of another file.

PLACE LIST. The program asks for a list of places to edit:

```
            (Page.Line) 1// 'CR'
                            ----
```

Typing 'CR' means that you accept the list that appears to the left of the double slash "//". The editor program will take you to the first place in the list, and while editing, you can progress through ^L the list by using the "@" command. The place list has no significance when creating a new file.

A place is specified by a page number, or by a page number, dot, line number. Use spaces or commas between places in the list, 'CR' or 'ESC' to terminate the list. Example:

```
          (Page.Line) 1// 3 7.11 22.45,10.10 'CR'
                          -----------------------
```

.PL FILES. The place list can also be read from a file. For the file "FILE1" the name of the place list file must be "#FILE1.PL" and for "FILE1.A" it must be "#FILE1-A.PL". Such files are created by the "(.PL)" option of the SEARCH program. See: <DOCUMENTATION>SEARCH.MANUAL for a description of the SEARCH program). The editor reads the first page (up to first Form Feed character), but not more than 1000 characters, of the .PL file. To read the list from a file, type "@" at the "(Page.Line)" request.

Next comes a request for editing mode:

```
          (Mode:  L, U, or R) L// 'CR'
                                  ----
```

On DATAMEDIAs the usual choice is L for writing and modifying text. The distinction between L and U is important on uppercase-only terminals such as the TEC. Even though TEC displays everything in upper case, the text in your file can be in either upper or lower case. Here you are asked to specify the case that you MOSTLY use in entering text in the file. The "L" of the example means that you plan to type mainly in lower case. On a TEC terminal, occasional uppercase letters for capitalizing sentences and proper names, for instance, are entered by using the various case-shifting commands of the editor.

WRITE MODE. Options "L" and "U" imply that you want to write, i.e., that the file will be modified. However, if the file is not your own (not in connected directory), a file with the same name is created in your directory, leaving the original file intact.

READ-ONLY MODE. Option "R", for "Read-Only Mode", allows you to view the contents of the file, but protects against accidental modification. In addition to safety, read-only mode is considerably faster. Its use is recommended whenever you do not intend to write or modify a file.

^L

USE OF 'CR' AND 'ESC', DEFAULT VALUES. Some questions come with a default option. The default is displayed to the left of the double slash "//", and you can accept it by typing only 'CR' or 'ESC'. To reject it, simply type your answer. Terminating an answer with 'ESC' means to use the default answers for all remaining questions. A fast way to start is to confirm the file name with an 'ESC'.

USE OF "?" AND "^". Any question in the starting dialog can be answered by typing the question mark "?". The editor then prints a message explaining the available alternatives and repeats the question. Replying with "^" takes back to the first question allowing the review and changing of options.

Once the file name and options are accepted, the first 23 lines of the file "FILE1" appear on the screen. If FILE1 is a new file, it is perfectly safe to start typing and experimenting with the editor. Straight typing should raise no major questions. Changing text that is already on the screen, however, requires knowledge of editor commands. Most of this manual is devoted to describing these commands, but first we concern ourselves with what is shown on the screen and how the keyboard is laid out.

What Is Shown on the Screen

WINDOW. You can think of the screen as a window to your file. Up to 23 lines of the file are shown at once. To view successive windows, type 'ESC' followed by the letter "W".

MESSAGE LINE. The topmost line (Line 0) is not part of the file but is information about the file and the editor. It gives the page and line numbers of window line 1, current editing modes, and messages from the editor. Possible editing mode letters are

I	insert mode,
R	read-only mode,
L	lowercase input mode, and
U	uppercase input mode.

If the program was started according to the sample protocol above, the message line should contain the following (TECs show a BOX character in place of the "*"):

```
****P.1***L.1****    L *******
```

meaning that top of window is page 1 line 1, and keyboard input mode is lowercase mode.

TEXT LINES, SPECIAL CHARACTERS. All text is displayed in upper case on TECs. Characters that have no graphic representation (i.e., control characters) are indicated by an all-bright "BOX" character on TECs, "*" on DATAMEDIAs. The ESC is displayed as "$".
^L

MARGINS AND LONG LINES. At each end of a text line there is a colon ":" or an asterisk "*" indicating the edge of the window. An "*" means that the line continues past the edge of the window. To demonstrate, the line below has 94 characters:

```
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmno
```

On your listing it might appear as two lines, but on the display the right end of the line is hidden. Initially the window width is 72 (77 at SUMEX). You can set it to any value n from 1 to 78 (1 to 77 for DATAMEDIAs) by using the "$$nX" command.

PAGE MARKS. In addition to genuine text lines, a text file can contain page marks which are displayed as page mark lines. Notice that the "P" of a page mark line is aligned with the colons of the left margin. Page marks can be inserted and deleted. No other editing of page mark lines is possible.

INDICATOR LINES. Equals sign "=" at left and right margins indicates an indicator line that is produced by the "$$=" command (explained later on).

END OF FILE. Letter "E" at left margin indicates end of file.

CURSOR. There is one special underline called the cursor. Its function is to point. At any one time the cursor points at a character, a word, a line, and a page. These are called the CURRENT character/word/line/page, respectively.

TEC and DATAMEDIA Keyboards

The keyboards have a main board, a 15-key pad to the right of the main board, and one or two EDIT keys beside the space bar. DATAMEDIAs have additional keys above and to the left of the main board.

MAIN BOARD. The main board resembles the Model-33 Teletype keyboard. It contains all the keys that are necessary for editing with TV-Edit. The additional keys are used to make editing more convenient.

Most keys generate characters by themselves (letters, numbers, space, carriage return, etc.). Five keys don't, but are held depressed while other characters are typed. They are:

SHIFT	The two shift keys resemble typewriter shift keys. However, on uppercase-only terminals (TECs) they are NOT used to generate selectively upper- and lowercase letters.

CTRL	Control key (two on DATAMEDIAs). Causes non-printing control characters to be generated. Some control characters, e.g., (CTRL)-I for TAB, have predefined meanings, others can be given meanings by the user (see section on Command Strings).

^L

EDIT	One (orange key) on TECs, two on DATAMEDIAs. Used for generating editing commands.
REPEAT	For repeated generation of a character.

Several of the "regular" keys deserve mention:

ESC (ALT MODE) (ENTER)	Escape, referred to also as "ALT MODE" ("ENTER" on some keyboards). Represented in this manual by "$". It tells the editor program that an editing command is about to be typed.
CR (RETURN)	Carriage Return signifies end of line.
LF	Line Feed moves cursor down.
DEL (RUBOUT)	Delete key. Also called "RUBOUT". Its main function in the editor is backspacing.
BREAK	The Break key has no function in the editor.
TAB	On TECs you might be fooled into believing that TAB is (SHIFT)-I. Actually it is (CTRL)-I.--Both keyboards have also a dedicated TAB key.
_	On TECs the underline (left arrow in 1965 ASCII) is generated by (SHIFT)-O (the letter "O"). The key caps in no way indicate this fact.--Both keyboards have also a dedicated "_" key.

"EXTRA" KEYS of the TEC keyboard (15-KEY PAD). The 5 x 3 pad on the right can be used for typing the characters shown in Table 1. These characters can, of course, be typed on the main board, but there you have to use the SHIFT or CTRL keys for them.

TABLE 1. TEC 15-Key Pad

TAB	_	CALL
[	\	]
(	^	)
<	=	>
(1)	LF	HOLD

NOTE: (1) SHIFT/SHIFT LOCK in TEC version of TV-Edit.

TECHNICAL NOTE: These keys actually generate lowercase letters that are TEC display hardware commands. They are translated into the characters shown in Table 1 by IMSSS TENEX and the TV-Edit program.
^L

"EXTRA" KEYS of the DATAMEDIA keyboard. The 15-key pad on the right generates characters as labeled on the key caps. The 5-key vertical column on the left generates the following control characters (from top to bottom): ^L, ^\, ^], ^^, and ^A, octal values 14, 34, 35, 36, and 1, respectively. The 7 keys in the top row are labeled according to their effect on the terminal. DPLX and TAPE keys are local only, and they control the FULL DUPLEX and TAPE lights. MSTR CLEAR, CLEAR, ROLL, and UNLOCK keys transmit control characters to the PDP-10. If the terminal is in half duplex (FULL DUPLEX light is off --use the DPLX key to turn it on or off), these keys also perform the indicated control functions.

In normal use (TTY mode) ROLL and FULL DUPLEX lights should be on, and TAPE and I/D lights should be off. While editing with TV-Edit, FULL DUPLEX light must be on and TAPE light must be off.

Editor Commands, Notation

The general rule of TV-Edit is that straight typing goes to the text file as such, unless you are in read-only mode, in which case straight typing is ignored. To do anything else, you have to type EDITOR COMMANDS. These commands allow you to move the cursor within the current window (pointing), view the file through successive windows, search for characters and words, insert, delete, replace, and rearrange text (if in write mode), and, of course, finish the edit and save your work on a disk file.

There are two ways to tell the editor that you are typing a COMMAND instead of ordinary text:

1. by typing an 'ESC' and then the command, or

2. by typing the command while depressing the EDIT key.

The first method always works. The second is more convenient, but for some commands must be complemented by the first.

NOTATION. Suppose you are editing an old file, have read the current window, and now want to see the next window and then proceed reading the file from the beginning of page 3. You should type:

```
'ESC' W    or   (EDIT)-W       to see the next window,

'ESC' 3G   or   (EDIT)-3G      to go to page 3.
```

'ESC' W means to type the ESC key and then type the letter "W"; (EDIT)-W means to hold down the EDIT key while typing "W". In the rest of this manual typing 'ESC' (i.e., the ALT MODE) is indicated by the dollar sign "$". Typing something with the EDIT key down is indicated by enclosing that something in brackets "[" and "]". Thus the above two commands will henceforth be written as:

```
                    $W        or       [W]

                    $3G       or       [3G]
```

^L

It should be pointed out that

$3G 3G   [3G]   and   [3]G

are typed differently but mean the same thing in the editor.

In general, TV-Edit commands can be thought of as one-letter commands (more precisely, one-character commands) that can take ARGUMENTS (modifiers). Arguments BEFORE the command letter usually give a (repeat) COUNT; arguments AFTER the command letter specify a TARGET. If the count is omitted, 1 is assumed (exceptions: "$G" and "$W" commands). Example:

[3SA]

is a command to search the current line for the letter "A" (the cursor moves to the right to the third "A"; "3" is the count argument, "S" the command letter, and "A" the target.

DOUBLE ESC. Typing two ESCs before the command means one of two things depending on the command: Either

(a) a huge count is to be used, or
(b) a different function is to be performed.

An example of the former is "$$G" or "[$G]" which takes to end of the file; of the latter, "$$F" for finish ("$F" does nothing). Commands for which the double ESC option signifies a different command are:

=, F, H, N, X, and Y.

These commands are explained later on.

If you start the command with two ESCs for a huge count, but continue with an honest count, the count given will be used. Thus, "$$3G" goes to page 3. Double ESC followed by count makes the count sticky for the following "again" command (explained later).

FURTHER NOTATION:

'CR'	Refers to typing the Carriage Return key.
'LF'	Line Feed key.
'DEL'	DELETE key or RUBOUT.
'SPACE'	Space bar.
(CTRL)-A CTRL-A ^A	Control characters are typed with the aid of the CONTROL key (while held depressed). Shown here are three "standard" ways of indicating CTRL-A.
<...>	Brokets are sometimes used to enclose a description of a mandatory argument.

^L

(...) Parentheses are sometimes used to enclose optional arguments or information.

"..." Double quotes are used inside text to delimit a command or a string.

WARNING: Many of the characters used in the notation are also TV-Edit command characters (e.g., ', (,), <, >, ^, ", -). We rely on the alertness and good will of the reader in deciphering the notation.

Finishing, Save-and-Continue

There are two ways (apart from a system crash) to get out of TV-Edit: an orderly finish and a panicky exit. Unless you have a good reason for doing otherwise, you should finish by giving the command

$$F -- Finish

(that's two ESCs, ALT MODEs, if you please, followed by "F"). The program clears the screen and prints the message: ".YOU.WILL.HAVE .TO.WAIT..". The wait is rather short for read-only mode, longer if you were actually editing (in write mode). Once finished, the editor program is best restarted by "CONTINUE".

The panic exit is via (CTRL)-C system call. If you type the ^C by accident and want to return to the editor, do a "CONTINUE" and follow it with "$$N" to refresh the screen. The "^C" exit is of minor consequence if you are in read-only mode, but if you are actually editing a file, you will lose some of the work of the current editing session.

While editing your file, the edited text is in a temporary state and becomes permanent only when the edit is finished. A sound practice, therefore, is for you to save your work every so often. This is done by

$F -- save-and-continue.

^L

3. VIEWING AND POINTING

Viewing a File

In read-only mode, looking at your text and pointing are about the only things you can do (it is also possible to "lift" text from a read-only file for copying it to another file). In modifying a file, one constantly performs the functions of viewing and pointing. This section describes the most important commands for viewing, the next section for pointing. You should try them out as they are introduced.

W -- Window

[W]	Display the next window.
[-W]	Display the previous window.
[3W]	Scroll 3 lines.

G -- Go

[G]	Go to (and display the beginning of) next page.
[3G]	Go to page 3, line 1.
[3.45G]	Go to page 3, line 45.
[.G]	Go to current page, line 1.
[.45G]	Go to current page, line 45.
$$G	Go to end of file.
[@]	Go to the next place as given by the place list.

NOTE: The place list is accepted by the editor as a reply to the "(Page.Line)" request when starting to edit. It can be either typed in or read from a '#name-ext.PL' file (created by the SEARCH program).

X, Y -- set window width, height

$$11X	Make the window 11 columns wide.
$$11Y	Make the window 11 lines high.

The "-W" and "G" commands can cause the message ".YOU.WILL.HAVE.TO.WAIT..." to appear. While waiting, you will notice that one of the numbers on the message line keeps counting. It counts page marks as they leave the program's temporary storage area. If 5 is shown on the message line and you type "[4G]", you will have to wait while the program does a "file wraparound" (whatever that might be).

^L

Moving the Cursor (Pointing)

To change something in your file you must first point at that something. The cursor does the pointing, and there are a number of commands for moving the cursor.

COMMANDS THAT MOVE THE CURSOR IRRESPECTIVE OF THE TEXT
(free cursor movement)

< -- left

[<]	Move cursor left 1 position.
[22<]	Move left 22 positions.
$$<	Move to left margin.
[$<]	Same as "$$<".

> -- right

[>]	Move cursor right 1 position.
[22>]	Move right 22 positions.
[$>]	Move to right margin.

^ -- up

[(n)^]	Move cursor up a line (n lines), to top line at most.

\, LF -- down

[(n)\]	Move down (n) line(s), to bottom line at most.
'LF'	Move down a line.

X, Y -- absolute X, Y

[55X]	Move cursor to column 55 on the current line.
[11Y]	Move cursor to line 11 on the current column.

COMMANDS THAT MOVE THE CURSOR WITH RESPECT TO THE TEXT
(restricted cursor movement)

SPACE -- right, by characters

['SPACE']	Point to the next character (the one on the right).
[5'SPACE']	Point to the 5th character (on the right).
[$'SPACE']	Point to end of line.

^L

DEL, RUBOUT -- left, by characters, or up, by lines

'DEL'	Point to the previous character.
['DEL']	Same as 'DEL' alone, except that at beginning of line moves cursor up.
[5'DEL']	Point to the 5th character on the left or 5th line above.
[$'DEL']	Point to the beginning of current line or page.

) -- right, by words

[)]	Point to beginning of next word.
[3)]	Point to beginning of 3rd word on the right.

(-- left, by words

[(]	Point to the beginning of previous word.
[5(]	Point to the beginning of 5th word on the left.

CR -- down (or up), by lines

'CR'	Point to beginning of next line. Start a new line if at end of file and in write mode.
['CR']	Same as 'CR' alone (no new line at end of file).
[12'CR']	Point to beginning of 12th line down.
[0'CR']	Point to beginning of current line.
[-'CR']	Point to beginning of previous line.
[$'CR']	Point to page mark at end of current page.
[$-'CR']	Point to page mark at beginning of current page.

The "<", ">", "^", "\", ")", and "(" commands can be typed on the 5 x 3 pad with the aid of the EDIT key. The keys are laid out to remind you of the functions.

The commands in the first group ("<", ">", "^", "\") observe the window boundaries but NOT end of line and page marks. The commands in the second group ('SPACE', "(", ")", 'DEL', 'CR') observe end of line and page marks (i.e., text) but NOT window boundaries. Both respect the end of file ("X" and "Y" commands don't).

Pointing by searching is discussed later.

^L

3. WRITING AND MODIFYING

Editing an old file should be attempted only after some knowledge of the editor has been acquired by "editing" in read-only mode and by writing on a temporary (new) file. It is possible to lose text accidentally, but if noticed right away it is usually possible to recover lost text or to "panic" exit and retain the old file. Some care is advisable, but one need not be overly concerned.

Writing and Overtyping

Straight typing goes to the file as such. Things to notice in straight typing are:

OVERTYPING. If you are pointing to a character and type, the new character replaces the old. Exceptions:

1. TABs never replace other characters but are inserted in front of them.

2. Regular characters never replace TABs (the last TAB). The TAB is pushed to the right.

3. Insert mode, which is explained later.

LONG LINES. If you attempt to type past the right margin, the current line is shifted to the left so that your typing position is always visible. An attempt to enter a really long line (more than 132 characters) results in the editor's request to break that line. When the cursor leaves a long left-shifted line, the line gets shifted back.

Deleting

The basic commands for DELETING are $D and $K.

D 'CR' -- Delete lines

[D'CR']	Delete the current line.
[5D'CR']	Delete 5 lines starting from the current line.
[$D'CR']	Delete down to bottom of window or next page mark, whichever comes first.
[-3D'CR']	Delete current line and the 2 lines above.

^L

D 'DEL' -- Delete words

[(n)D'DEL']	Delete the current word (and n - 1 following words).
[-5D'DEL']	Delete the current word and 4 preceding words.

D <letter> -- Delete characters
K -- delete (Kill) characters

[DD]	Delete current character (the second "D" could be replaced by any letter).
[K]	Same as "[DD]". This is the preferred way to delete characters; you need to type only one letter.
[11K]	Delete 11 characters.
[$K]	Delete rest of current line.
[-(n)K]	Delete (n) characters LEFT of the current character.

NOTE: Deleting up to a target character is often the most efficient way of deleting. This will be explained later.

Inserting, Insert Mode

The basic command for INSERTING is $I. Page marks are inserted with the "P" command.

I -- enter Insert mode

INSERT MODE. The "I" command leaves the editor in insert mode. This condition is indicated by a blinking "I" on the message line. Once in insert mode, any text typed is inserted in the text instead of overtyping old text. To leave insert mode, type any pointing command.

Things to notice about typing in insert mode:

(a) DEL (RUBOUT key) without the EDIT key deletes the character to the left of the cursor (the one last inserted).

(b) CR (without the EDIT key) inserts a blank line between the current line and the one below it. An attempt to break a line into two lines by inserting a 'CR' fails. Use the "B" command for that (explained later).

I 'CR' -- Insert lines

[I'CR']	Insert a blank line above the current line.
[5I'CR']	Push current line down 5 lines and insert 5 blank lines.
[$I'CR']	Insert as many blank lines as there are lines from current line to bottom of screen.
[-3I'CR']	Push current line up 3 lines and insert 3 blank lines.

^L

I 'DEL' -- Insert words

[I'DEL'] (Prepare to) insert text in front of the current word.

P -- insert a Page mark

[P] Insert a page mark above the current line.

Page Marks

Page marks serve several useful functions:

(a) With the "G" command you can go quickly to the general area where you want to edit.

(b) The page marks limit the extent of the text-sensitive commands 'CR', 'DEL', D'CR', S'CR', Z'CR', and 'quote'. They can sometimes be utilized for terminating string execution. ("S", "Z", and 'quote' commands and string execution are explained later).

(c) In listing a file, page mark starts a new page.

Page marks are inserted with the "P" command, deleted with the "D'CR'" command.

* * * * *

By now we have introduced enough commands to allow you to do useful editing and to develop a feel for the editor. We would, in fact, discourage the beginner from reading the rest of the manual until he has used the editor for a while.

Once you feel that the editor program ought to do more for you, you should read ahead. Until then it might suffice to know that there are commands that search for characters and words, break a line in two, join two consecutive lines, recover deleted lines and words either at the spot where they were deleted or elsewhere on your file, copy and move lines and words within the window, and others. The full power of a TV-Edit is utilized only via the mechanism of (command) strings.

**** END OF BEGINNER'S MANUAL ****

^L

4. MORE POWER TO YOU

The rest of this manual contains commands and information that is helpful once you are somewhat familiar with TV-Edit.

Aborting

The need for emergency exit arises in several ways:

(a) You have partially typed in a command but decide not to execute the command after all (you have already typed "$55" when you had intended to type "$.55G").

(b) A command (or command string) that is being executed seems to be doing more than you originally intended.

The way out of the fix is:

^L -- Leap, abort

(CTRL)-L Abort the command (or command string) currently being input or executed. Flush TTY input buffer.

NOTE: Typing "^C" followed by "REE 'CR'" has a similar effect, but can also have undesirable side effects. It is recommended only in an emergency, when the ^L fails.

Suppress Output

There are times when you do not want to wait for the whole window to be printed out (especially if you are connected to the system over a slow communications line). You can interrupt the display generation by typing

^O -- suppress Output (shut up)

Every other (CTRL)-O stops window generation, every other starts it up again. Editing with self-calling strings can at times be speeded up considerably by letting the editor run "in the dark".

^L

Repeating the Last Command

A -- Again

- [A] Repeat the last command but use count 1.
- [5A] Repeat the last command using count 5.
- $$A Repeat the last command the way it was originally given (true for most commands).

The "A" command is a most versatile command since it can be used for repeating almost any other command. Some commands might be just as easily typed themselves again, but others are easier to repeat using the "A" command (for target commands it eliminates the need to respecify the target).

In executing a command that requires a target, you should first give a count that gets you almost there, and then finish up with the "A" command.

STICKY COUNT. From the example above you might have noticed that the original count is replaced by 1 when the command is repeated. To make the original count stick, type the original command with two ESCs. For example, "[$2D'CR']" deletes two lines, "[A]" thereafter deletes two lines instead of one.

The "again" command has not yet been developed to its fullest, but even in its current form is quite worthy of your consideration.

Obtaining Information

Some information about the state of the editor is continually displayed on the message line. If the text in the file contains control characters (and lowercase for TECs), additional means are needed for determining the exact contents of a line of text.

= -- what's the character, where are we

[=] Print on message line the following (an example):

```
%A='141...X=59...P.L=20.38...17.45.. FILE1.;1
```

i.e.,
- (a) current character is lowercase "A" ("%A" for lowercase A, "^A" for (CTRL)-A), and its octal value is 141;
- (b) cursor is at column 59 of page 20 line 38;
- (c) page 17 line 45 is the first line in the temporary memory storage (that's how far you can back up without having to wait); and
- (d) the name of the file is "FILE1.;1".

$$= Indicate, above the current line, the exact contents (uppercase, lowercase, control characters) of the current line. The following symbols are used on the indicator line:

Indicator character	Real character
.	space or regular punctuation (40-100,133-137)
^	(TECs only) uppercase letter (101-132)
_	(TECs only) lowercase letter (141-172)
!	(TECs only) lowercase punctuation (140,173-176)
BOX or *	DEL, RUBOUT (177)
:	TAB (11)
left-L or \	end of line (12,15)
@	NULL character, CTRL-@ (0)
A...Z	CTRL-A,...,CTRL-Z (1-32)
[..._	CTRL-[,...,CTRL-_ (33-37) (ESC is CTRL-[, TENEX EOL is CTRL-_)

The indicator line itself is indicated by the equals sign "=" at left and right margins in place of ":" or "*". It overprints the line above the current line, but does not affect your file in any way. "$-N" lets you view the messed-up line again.

"[A]" after an "=" command indicates the NEXT character or line.

N -- refresh screen ("N" for no-op)

[4N]	Refresh (reprint) 4 lines starting from current line.
[-4N]	Refresh 4 lines above the current line.
[0N]	Refresh line 0 (the message line).
[$N]	Refresh the entire screen.

Upper and Lower Case

Except for the ".L" and ".U" commands for converting old text to lower or upper case, this and the next section are primarily of interest to TEC users.

The "=" command allows you to see on an uppercase-only terminal whether a character in the file is in upper or lower case. Here we introduce commands for ENTERING text in either upper or lower case, and for CHANGING the case shift of text that already is in the file.

An "L" or "U" on the message line tells whether the text being typed is to be entered in upper or lower case. In starting the edit, one is selected as the primary input case. Regardless of the primary case, the current input case can be either "L" or "U" for the duration of one or more input characters. When the current case differs from the primary case, the current case letter on the message line blinks. This is the program's way of reminding you that you are entering text in the "wrong" (non-primary) case.

H -- shift

[H]	Shift (from "L" to "U" or from "U" to "L") for the next input character. NOTE: On TECs the preferred way to shift for one character is to type the lower-left key of the 5 x 3 pad by itself. On DATAMEDIAs use the SHIFT key.
[8H]	Shift next 8 input characters (or until end of line).
$$H	Shift lock: Permanently shift the case. NOTE: On TECs the preferred way to shift-lock is to type the lower-left key of the 5 x 3 pad while the EDIT key is depressed. On DATAMEDIAs use the LOCK key.

U, L -- enter in Upper/Lower case

[(n)U]	Enter next (n) input(s) in upper case.
[(n)L]	Enter next (n) input(s) in lower case.

The effect of these two commands is terminated by end of line.

.U, .L -- convert to Upper/Lower case

[(n).U]	Convert (next n) character(s) to upper case.
[(n).L]	Convert (next n) character(s) to lower case.

Lowercase Punctuation on Uppercase-Only Terminals (TECs): `{|}~

The reason for calling these characters "lowercase punctuation" is that they do not appear on uppercase-only terminals. Refer to Table 2 below to see how they are generated and displayed.

TABLE 2. "Lowercase Punctuation" on TECs

	Character	Octal value	Typed as(*)	Displayed at input time as	Otherwise displayed as
`	accent grave	140	@	'	@
{	left brace	173	[	<	[
|	vertical bar	174	\	!	\
}	right brace	175	]	>	]
~	tilde	176	^	-	^

NOTE: (*) To enter "lowercase punctuation" from the keyboard, the case shift indicator on the message line must be BLINKING, i.e., with a lowercase file you must be in upper case and with an uppercase file in lower case.

INPUT TIME DISPLAY. Ordinarily, these characters are displayed as their uppercase equivalents. However, at input time they are not displayed as the corresponding uppercase characters, but as other characters that resemble the lowercase characters that they really
^L

are. Hopefully this draws your attention to the fact that what goes in the file isn't exactly what you typed nor what you see on the screen.

In character and string search, the uppercase equivalents are to be typed as targets (just as is the case with lowercase letters). You might be amused when searching for these characters on a newly entered line.

The ".U" command does NOT convert lowercase punctuation characters in your file to the corresponding uppercase characters.

Entering Arbitrary Characters

' -- ASCII value in octal

[' 33'CR'	Enter the ASCII character whose value is 33 (octal) in the file.

NOTE: The octal number is accepted on the message line in the same way as string search argument is. See "T" command.

Searching

Several commands let you specify a target character or string. The editor program will then find the target, and depending on the command, will either point to it or delete text up to it. In many instances the target commands, coupled with the "again" command, provide the most efficient means for pointing and deleting.

No distinction between upper and lower case is made in searching.

Search that does not find terminates string execution. The significance of this is discussed later.

In the examples below, the letter "A" serves as an example. It could be replaced by any character other than 'CR' or 'DEL'.

S -- character Search

[SA]	Point to the next "A" on the current line.
[3SA]	Point to the 3rd "A" (on the right).
[-3SA]	Point to the 3rd "A" on the left of the current character.

S 'DEL' -- word Search

[S(n)'DEL'A]	Point to the 1st (nth) word that is to the right of the current word and starts with an "A".
[-S2'DEL'A]	Point to the 2nd word starting with "A" that is to the left of the current word.

^L

S 'CR' -- line Search

[S'CR'A]	Point to the 1st line starting with "A" that is below the current line. A line starts with "A" if the first visible character is "A". SPACE and TAB are invisible characters.
[-3S'CR'A]	Point to the 3rd line starting with "A" that is above the current line.

T <string> -- string search

[100T]THE'CR'	Scan up to 100 LINES starting from current cursor position (+1) and point to the first occurrence of the word "THE".
$$TEND'CR'	Scan for the first "END" that is to the right of or below the current character.

NOTE: The count refers to the maximum number of LINES that are to be scanned. The command is accepted in a standard fashion up to, and including, the command letter "T". The string argument is accepted on the message line, where 'DEL' can be used for editing the argument string (it deletes from the rear). The 'CR' that terminates the string argument is displayed as a left-L or "\". String search command that is given without a target (with empty target string) uses the most recent string target:

$$T'CR'	Search for the next occurrence of the string that was last searched for.

Search-and-Delete

Z -- delete (Zap) characters

[ZA]	Delete characters on the right up to the 1st "A".
[-3ZA]	Delete characters on the left up to and including the 3rd "A".

Z 'DEL' -- delete (Zap) words

[Z'DEL'A]	Delete words (on the right) up to the 1st starting with "A".
[-3Z'DEL'A]	Delete words (on the left) up to and including the 3rd one starting with "A".

Z 'CR' -- delete (Zap) lines

[Z'CR'A]	Delete lines (downwards) up to the 1st starting with "A".
[-3Z'CR'A]	Delete lines (upwards) up to and including the 3rd starting with "A".

Recovering

TV-Edit is forgiving. If you accidentally delete text, you can recover it by "oopsing" right after the deletion occurred (no intervening commands) or by recovering it later, provided that you have not deleted more than about a windowfull in the intervening time. It is always possible to "oops" what was just deleted.

O -- Oops

[O]	Undo the last command provided it was a delete. If it was not a delete but you have been editing a line, restore the line to what it was when you started editing it.

R 'DEL' -- Recover words

[(n)R'DEL']	Recover (n) most recently deleted word(s) and insert on the current line.
$$R'DEL'	Recover the words deleted with the last word-delete command. Observes word-delete count and direction.

R 'CR' -- Recover lines

[(n)R'CR']	Recover (n) most recently deleted line(s) and insert above the current line. Will undo "[(n)D'CR']".
[-3R'CR']	Recover three most recently deleted lines and insert them below the current line. Will undo "[-3D'CR']".
$$R'CR'	Recovers the lines deleted by the last line-delete command the best it can. Observes last line-delete count and direction (up or down).

NOTE: The maximum number of lines recovered by one "R" command is restricted to the number of lines between the current line and the top or bottom of window, depending on the direction of the command. It is therefore at times necessary to execute the command more than once to get all the saved lines (maximum of 24) back in the file.

The editor "remembers" the number of items (words or lines) last deleted, and the direction of deleting. The double ESC version of the "R" command uses this information to limit the extent of the recovered material. You can override the internal settings by giving your own sign and count with the command.

The word and line recovery mechanisms are independent of each other. No corresponding character recovery mechanism exists.

^L

Breaking and Joining of Lines

B -- Break a line

[B]	Cut current line into two at where the cursor is.

A way to insert blank lines without going to insert mode is to do a "[B]" while the cursor is at the beginning of a line.

J -- Join two lines

[J]	Join current line to the one below.
[-J]	Join current line to the one above.

Moving Text

Breaking and joining of lines is one form of rearranging. However, it does not allow you to change the order of the text. Deleting lines or words and then recovering them someplace else makes reordering possible. Here are other commands for reordering, as well as for producing duplicate copies of words and lines. Strings (see below) provide still another method of moving text.

M 'CR', M 'DEL' -- Move a line, Move a word

[(n)M'CR']	Move the current line down over one line (n lines). This command never moves past a page mark, or bottom (or top) of window.
[-2M'DEL']	Move current word left over two words.

C 'CR', C 'DEL' -- Copy lines, Copy words

[5C'CR']	Duplicate this and next 4 lines. The number of lines copied is restricted by a page mark or bottom (or top if "-C") of window.
[C'DEL']	Duplicate the current word.

.D 'CR', .D 'DEL' -- "remember" lines/words

[.D'CR']	Save a copy of the current line for later recovering.
[-3.D'CR']	Save copies of this and two preceding lines.
[$.D'DEL']	Save copies of words from here to end of line.

NOTE: The ".D" command uses the same mechanisms as the "D" command for saving lines or words. They are recovered by the appropriate "R" commands. In addition to leaving the file intact (only the cursor moves), the ".D" command causes internal counting whose function is to hang onto all the lines (words) that have been stored with the corre-

^L

sponding ".D" command or subsequent delete commands. When the internal storage is full, a message asking you to retrieve deleted lines (words) will appear. A line (word) once stored using the ".D" command has to be retrieved sooner or later to get rid of that message and allow unrestricted deleting.

Command Strings (Macros)

TV-Edit has the means for storing a sequence of typed commands and text as a string in such a way that at some later time you can reexecute the sequence by typing just one character, the string ID. Strings allow you to design your own super commands.

A string in TV-Edit consists of the string ID character, also called the name of the string or the string call character, followed by the effective string (the part that gets executed). Most control characters are valid string ID characters. The following ones are not: ^@, ^C, ^I, ^J, ^L, ^M, ^O, ^T, and ^[. The effective string may contain string calls (names of other strings), including calls of itself.

Some commands are restricted so as to work only if entered from the keyboard. An attempt to execute such commands under string control results in (orderly) string execution abort, and a message to that effect will be displayed. This is the editor program's way of preventing excessive damage by runaway strings.

In addition to illegal commands, string execution abort can be caused by other internal conditions, for example:

(a) Execution has been under string control for the last 1000 "input" characters. This is a precaution against "looping" strings.

(b) Search fails. This makes automatic search-substitute feasible. It is safe to define a string "^Q[SA]B'DEL'^Q", where ^Q is the string ID) and use it to replace all "A"s by "B"s on a line. String execution terminates when the cursor reaches end of line and fails to find more "A"s.

(c) String "attempts" to delete more lines (words) than can be recovered. It is always possible to undelete the lines (words) deleted by the string just executed. No such precaution against character deletion exists.

(d) String "attempts" to delete a page mark or write on a page mark. A useful device for stopping self-calling strings.

/ -- begin or end string definition

```
     [/]^QABC[/]  Define the string "^QABC", (CTRL)-Q is the string
                  ID, the effective string is "ABC".  This is the
                  standard way of defining strings.
     [/]^WXYZ[A]  Define the string "^WXYZ^W".  The first (CTRL)-W
                  is the string ID, the second (last) is a self-call
^L
```

due to terminating the string definition with an "[A]" instead of "[/]". The effective string is "XYZ^W".

After typing the initial "[/]", the message ".TYPE.STRING.ID." appears. After you type it (say, ^Q), the message changes to ".DEFINING..". It disappears as soon as you type the terminating "[/]" or "[A]", or the ^L.

If the ID you type is invalid or already defined, you will be properly warned and given a chance to try again.

UNDEFINING. If you type the terminating "[/]" right after the string ID has been accepted, the ID is freed (string undefined). Another way to undefine a string is to abort (with ^L) while defining a string. All strings can be undefined at once using the "$$.RO" command explained further down.

R <string ID> -- Retrieve a string

[R]^Q Writes (inserts) the string for ^Q in your file.

" -- define a string from file

["] Define a string: Take the current character as string ID, the immediately following text, up to "$/" or page mark, as the effective string.

The "R<ID>" command saves strings on the file the way 'quote' command wants to see them.

MOVING TEXT. The strings provide one more method of copying and moving text. You can use the 'double quote' command to store up to a windowfull of text "under" a control character, and retrieve it with the "R" command (or by entering insert mode and typing the control character--risky if the text contains control characters which are string calls). After retrieving, some clean-up might be necessary. To avoid overflowing the SAIL string space (and blowing up the editor), store only few pages of text under control characters at once, and "undefine" the control characters as soon as you have retrieved the text you want.

Saving and Loading of Sets of Strings

Currently defined strings form a set. File 'TV-STRINGS.PMAP' in your directory is for storing sets of strings. Sets in the string file are identified by the letters A,...,Z. The letters refer to TENEX pages of the file (letter "A" to page 101 octal, etc.).

The set in memory at the time of finish is referred to as the MOST RECENT SET, and is automatically stored on page 0 of the string file when finishing, and loaded when starting the editor. This provides continuity from one editing session to the next.
^L

.D <letter> -- save (Dump) strings

$$.Dx	Save current strings on page x of TV-STRINGS.PMAP.
$$.D.	Save current strings on a "letter" page last specified during this editing session.

NOTE: The set in memory REPLACES the designated set in the string file. If the set in memory is empty (as a result of "$$.R0", see below), the string file page in question is deleted.

.R <letter> -- load (Read) strings

$$.Rx	Load (merge) strings from page x of the string file.
$$.R.	Load strings last specified by a letter during the current editing session.
$$.R0	Clear strings from memory.

NOTE: The set in the file is MERGED to the set in memory. Thus a string in memory is redefined only if it is defined in the set being loaded from file.

If x is not a letter (non-0 digit, for instance), the most recent set is assumed.

^L

5. WHAT HAPPENS TO YOUR DIRECTORY AND FILES

In its current form TV-Edit uses two files for write-mode editing, OLD and NEW. The old file is the one you specify at the beginning of edit. Let it be "FILE1.A". The editor creates a new file and gives it the name "#FILE1-A.TV". When the message ".YOU.WILL .HAVE.TO.WAIT..." appears, file "#FILE1-A.TV" is closed and holds the updated file. It is renamed to "FILE1.A", and the old "FILE1.A" vanishes. Failure to rename "#FILE1-A.TV" to "FILE1.A" does not mean that you have lost your work. It is there, but under a new name.

If an editing session is aborted either by ^C system call or by a system crash, the "#name-ext.TV" file contains the beginning of the new file. Use TVFIX and TMERGE programs if you want to recover the work from the aborted session.

WARNING: If you switch between SOS and TV-Edit, the SOS line number are not displayed. If you specify read-only mode, they of course stay intact in your file, but they vanish if you edit in write mode. Also, BASIC line numbers are of the disappearing variety.

^L

6. IN CASE OF TROUBLE

1. MESSED-UP SCREEN. Type $$N to refresh.

2. RUNAWAY CURSOR. Cursor is off the main screen and every command seems to just ring the bell. Type

```
[1X1Y]
```

to get cursor to beginning of line 1.

3. RUNAWAY PROGRAM. Try ^L first. If that fails, type

```
^C          to stop the program. Once stopped, type
REE 'CR'    for REENTER, to get back to the editor,
$$N         to refresh the screen.
```

If that fails, type ^C again and give up. You lose.

4. FILE "name.ext" LOST. Check your directory for the file

```
#name-ext.TV
```

It possibly is the updated file. Run the program TVFIX, specify file "#name-ext.TV", and check what is in the file (use Read-only mode). Then either merge (use TMERGE) file "#name-ext.TV" to file "name.ext" or rename "#name-ext.TV" to "name.ext".

5. You are typing to the EXEC instead of the editor. You (accidentally) typed ^C. If you want to save your editing, type

```
CONT 'CR'    to continue,
$$N          to refresh the screen.
```

6. MYSTERIOUS MESSAGES. Some messages are there to help you, others are for debugging the editor program. The latter are of the form:

```
cryptic name: cryptic story
```

where the name is a name of a procedure, and the story a description of the error condition. These debugging messages are accompanied by long ringing of the bell. Two are known to surface from time to time: "SETXY: EMPTY W[1]" when starting to edit an old file that is of zero length, and "LOPTAIL: TS > MAXS INITIALLY" when the old file contains lines longer than 132 characters. Other debugging messages should be brought to the attention of the author. It ought to be safe to continue after such messages.

APPENDIX I. NOTATION USED IN THIS MANUAL

$	ESC key, also called ALT MODE or ENTER.
'ESC'	ESC key.
'CR'	(Carriage) Return key.
'LF'	Line Feed key.
'DEL'	DELETE or RUBOUT key.
'SPACE'	Space bar.
(CTRL)-A CTRL-A ^A	Control characters are typed with the aid of the CONTROL key. Shown here are three "standard" ways of indicating CTRL-A.
(EDIT)-A	Edit characters are typed with the aid of the EDIT key.
[...]	Brackets indicate typing that is done while holding down the EDIT key. (EDIT)-A and [A] mean the same thing.
<...>	Brokets are sometimes used to enclose a description of required arguments.
(...)	Parentheses are sometimes used to enclose optional arguments or information.
"..."	Double quotes are used inside text to delimit a command or a string.

APPENDIX II. TEC/DATAMEDIA TV-EDIT COMMANDS

NOTATION:

$	ESC, ALT MODE. Instead of typing the ESC and then the command, you can hold down the EDIT key (orange on TECs) as you type the command. They mean the same thing.
^	Control character indicator
CR	(Carriage) Return key
LF	Line Feed key
SPACE	Space bar
DEL	DELETE or RUBOUT key
-	minus sign before command letter is optional (its effect is to do the command backwards)
n	count before command letter is optional (its effect is to "repeat" the command n times)
w	"word mode" option available (type 'DEL' after command letter to delete (insert, etc.) words)
l	"line mode" option available (type 'CR' after command letter to delete (insert, etc.) lines)
c	command requires a target character
s	command requires a (target) string argument
numbers	page numbers in this manual

Pseudo-commands (not seen by the command interpreter):

^A	Again. Repeat last command. (Predefined string call), 20
^L	Leap. Abort a command, 19
^O	Suppress output, 19
SHIFT/ SHIFT LOCK	Invert keyboard input case. Generated by lower-left-corner key of the 5 x 3 pad (TECs only), 22

Genuine commands (seen by the command interpreter):

$n	LF		Cursor down, 14
$-n	CR		Cursor down, by lines, 15
$-n	SPACE		Cursor right, by characters, 14
$-n	DEL		Cursor left, by characters, or up, by lines, 15
$	"		Define a string to be the text on a file, 28
$	'	s	Enter a character in octal. s must be octal string, 23
$n	(		Cursor left, by words, 15
$n	)		Cursor right, by words, 15
$	/		Begin/end of string definition, 27
$n	<		Cursor left by columns, 14
$	=		where are we, what's the character? 20
$$	=		Indicate uppercase/lowercase/control characters of

^L

$n	>		Cursor right, by columns, 14
$n	\		Cursor down, 14
$n	^		Cursor up, 14
$	@		Go to next place according to Place List, 5, 6, 13
$-n	A		Again. Repeat last command, 20
$	B		Break a line, 26
$-n	C	wl	Copy words or lines, 26
$-n	D	wl	Delete words or lines, 16, 17
$-n	.D	wl	Hide (store) copies of words or lines, 26
$$	.D	c	Dump strings in TV-STRINGS.PMAP file, 29
$$	.D.		Dump strings in currently assigned set, 29
$	F		Save-and-Continue, 12
$$	F		Finish, 12
$p.l	G		Go to Page p Line l, 13
$n	H		Shift next n input characters (for TECs), 22
$$	H		Shift lock (for TECs), 22
$n	I	wl	Enter insert mode, insert, 17, 18
$-	J		Join two lines, 26
$-n	K		Delete characters, 17
$n	L		Shift to lower case (for TECs), 22
$n	.L		Convert text to lower case, 22
$-n	M	wl	Move a word or a line, 26
$	N		Refresh current line, 21
$	0N		Refresh message line, 21
$-n	N		Refresh n lines, 21
$$	N		Refresh entire screen, 21
$	O		Oops. Undo the delete, 25
$-	P		Insert a page mark, 18
$-n	R	wl	Recover words or lines, 25
$	R	c	Retrieve a string, enter it in file, 28
$$	.R	c	Read strings from TV-STRINGS.PMAP file, 29
$$	.R.		Read currently assigned set of strings, 29
$$	.R0		Clear strings, 29
$-n	S	wlc	Character search, 23, 24
$n	T	s	String search, 24
$n	U		Shift to upper case (for TECs), 22
$n	.U		Convert text to upper case, 22
$-n	W		Window, 13
$n	X		Cursor to column n of window, 14
$$n	X		Set window width to n columns, 11
$n	Y		Cursor to line n of window, 14
$$n	Y		Set window height to n lines, 13
$-n	Z	wlc	"Zap", delete to a target character, 24

A BRIEF GUIDE TO TVEDIT

If you make an error while typing a message you can either delete back to the point where the error was made or invoke an editor. To delete characters type the key marked "del" or "rubout" on your keyboard. To invoke an editor hold down the key marked CTRL and type a B. The computer will respond with the query:

```
        (Insert file or invoke editor (F,E, or ?)?
```

If you type an E at this point the computer will then ask for the type of editor as follows:

```
        (editor:
```

Typing "TVE" and the hitting the key marked ESC will put you into TVEDIT. Note that tvedit only works on datamedia terminals and will not function on TIs or other "hard copy" terminals.

There are a number of simple commands to facilitate editing your file. Most of these require that you hold down the key marked EDIT on your keyboard. If your terminal does not have the EDIT key you can get the same result by prefacing each command with an ESC.

Once you are inside TVEDIT there are two types of things you can do. You can do commands which modify the file and you can execute commands to reposition the cursor (the blinking underline which shows where you are). The following are cursor movement commands:

EDIT- >	Moves the cursor onecharacter to the right
EDIT- <	Moves the cursor one character to the left
EDIT- ^	Moves the cursor up one line
EDIT- lf	Moves the cursor down one line
EDIT- (	Moves the cursor one word to the left
EDIT-)	Moves the cursor one word to the right

All the TVEDIT commands will also to a numeric argument of the form
        EDIT- #- cmnd ,such as EDIT-3-< which will move you three characters to the left. To do this you would hold down the key marked EDIT, type a "3" and then type a "<".

If you type a line without holding down the EDIT key whatever you type will overwrite whatever is at the current cursor position. To insert text type EDIT- I. This will put you in insert mode. Now whatever you type will be inserted at the current cursor point. Moving the cursor will take you out of insert mode.

To delete a character hold down the EDIT key and type a K.This will cause the character at the current cursor position to be deleted.

When you are through editing you can return to MSG by typing esc esc F. Now you will be back in MSG at the end of you file. To see what you have already typed hold down he CTRL key and type E.

<DOCUMENTATION> TVEDIT, COMMAND

APPENDIX II. TEC/DATAMEDIA TV-EDIT COMMANDS

NOTATION:

$	ESC, ALT MODE. Instead of typing the ESC and then the command, you can hold down the EDIT key (orange on TECs) as you type the command. They mean the same thing.
^	Control character indicator
CR	(Carriage) Return key
LF	Line Feed key
SPACE	Space bar
DEL	DELETE or RUBOUT key
-	minus sign before command letter is optional (its effect is to do the command backwards)
n	count before command letter is optional (its effect is to "repeat" the command n times)
w	"word mode" option available (type 'DEL' after command letter to delete (insert, etc.) words)
l	"line mode" option available (type 'CR' after command letter to delete (insert, etc.) lines)
c	command requires a target character
s	command requires a (target) string argument
numbers	page numbers in this manual

Pseudo-commands (not seen by the command interpreter):

^A	Again. Repeat last command. (Predefined string call), 20
^L	Leap. Abort a command, 19
^O	Suppress output, 19
SHIFT/ SHIFT LOCK	Invert keyboard input case. Generated by lower-left-corner key of the 5 x 3 pad (TECs only), 22

Genuine commands (seen by the command interpreter):

$n	LF	Cursor down, 14
$-n	CR	Cursor down, by lines, 15
$-n	SPACE	Cursor right, by characters, 14
$-n	DEL	Cursor left, by characters, or up, by lines, 15
$	"	Define a string to be the text on a file, 28
$	' s	Enter a character in octal. s must be octal string, 23
$n	(	Cursor left, by words, 15
$n	)	Cursor right, by words, 15
$	/	Begin/end of string definition, 27
$n	<	Cursor left by columns, 14
$	=	Where are we, what's the character? 20
$$	=	Indicate uppercase/lowercase/control characters of a line, 21

$n	>		Cursor right, by columns, 14
$n	\		Cursor down, 14
$n	^		Cursor up, 14
$	@		Go to next place according to Place List, 5, 6, 13
$-n	A		Again. Repeat last command, 20
$	B		Break a line, 26
$-n	C	wl	Copy words or lines, 26
$-n	D	wl	Delete words or lines, 16, 17
$-n	.D	wl	Hide (store) copies of words or lines, 26
$$	.D	c	Dump strings in TV-STRINGS.PMAP file, 29
$$	.D.		Dump strings in currently assigned set, 29
$	F		Save-and-Continue, 12
$$	F		Finish, 12
$p.l	G		Go to Page p Line l, 13
$n	H		Shift next n input characters (for TECs), 22
$$	H		Shift lock (for TECs), 22
$n	I	wl	Enter insert mode, insert, 17, 18
$-	J		Join two lines, 26
$-n	K		Delete characters, 17
$n	L		Shift to lower case (for TECs), 22
$n	.L		Convert text to lower case, 22
$-n	M	wl	Move a word or a line, 26
$	N		Refresh current line, 21
$	ON		Refresh message line, 21
$-n	N		Refresh n lines, 21
$$	N		Refresh entire screen, 21
$	O		Oops. Undo the delete, 25
$-	P		Insert a page mark, 18
$-n	R	wl	Recover words or lines, 25
$	R	c	Retrieve a string, enter it in file, 28
$$	.R	c	Read strings from TV-STRINGS.PMAP file, 29
$$	.R.		Read currently assigned set of strings, 29
$$	.RO		Clear strings, 29
$-n	S	wlc	Character search, 23, 24
$n	T	s	String search, 24
$n	U		Shift to upper case (for TECs), 22
$n	.U		Convert text to upper case, 22
$-n	W		Window, 13
$n	X		Cursor to column n of window, 14
$$n	X		Set window width to n columns, 11
$n	Y		Cursor to line n of window, 14
$$n	Y		Set window height to n lines, 13
$-n	Z	wlc	"Zap", delete to a target character, 24

NEW FEATURES OF DATAMEDIA/TEC/IMLAC TV-EDIT

28-Sep-77

CONTENTS

1. NEW COMMANDS

Insert a single character when not in insert mode.

$Q	Insert the next character typed rather than overwriting, regardless of the setting of insert mode.

Transpose characters.

$V	Transpose the last character typed with the one before it.

Comment features.

$;	If a comment already exists on the current line, align it to the comment column. Otherwise start a new comment at the comment line.
$-;	Remove any comment from the current line.
$n;	Set the comment column to be column n.
$.;	Set the comment column to be the current x position.

A comment is delimited by a comment start string and a comment end string (possibly null), depending on the extension of the file being editted. The defaults are ";" and null

CCL startup.

$$-nF	Just like $$nF, except that the EXEC will reexecute the last CCL type command (ie LOAD, COMPILE, EXECUTE, etc). This facilitates movement between the editor and an assembler.

2. OLD BUGS

SNDMSG entry.

If TVEDIT is entered from SNDMSG, $$F will overwrite the old version, rather than creating a new one which SNDMSG would not find.

Terminal speed.

TVEDIT now determines the terminal's speed when started up and will pad accordingly.

10-Oct-75
=========

CONTENTS

1. CHANGES TO COMMANDS

Save-and-continue.

$F	Replaces $$.F

Go to Page.Line.

$+m.nG	Go to current page + m, line n. $+G is now equivalent to $G
$-m.nG	Go to current page - m, line n. $-G goes to top of previous page.
$@	Go to next place in the place list. Replaces $+G
$-@	Go to the last fixed place.

Fixing a place to go to (NEW FEATURE).

$.@	Fix current place for further reference.

Current place can also be fixed by typing two ESCs (or EDIT-ESC) before G and @ commands:

$$nG	Fix current place, go to n. NOTE: $$G no longer takes to end of file. Use $999G instead.
$$@	Fix current place and go to next place in place list.
$$-@	Fix current place and go to last fixed place. This allows bouncing back and forth between two places in a file.

RESTRICTIONS: There is only one fixed place at a time. It is stored as a page.line number and is NOT updated when page marks and lines are inserted and deleted. You cannot (yet) fix a place in one file and get there while editing another file. The fixed place is not saved from one editing session to the next.

Converting words and lines to upper or lower case.

$n.U<DEL>	Convert n words to upper case.
$n.L<DEL>	Convert n words to lower case.
$n.U<CR>	Convert n lines to upper case.
$n.L<CR>	Convert n lines to lower case.

Nothing comes free. To convert individial letters, an extra character must be typed, e.g.,

$n.UU	Convert n characters to upper case (this used to be "$n.U").
$n.LL	Convert n characters to lower case.

Pointing.

$nX	Point to column n of the current line. NEW FEATURE: If target column is beyond the righ or left margin, the line shifts to bring target to view.

2. CR IN INSERT MODE; TABS

Typing a RETURN while inserting within a line is now a no-op (it beeps). "$I<CR>" still inserts a line above the current line.

The TAB "accordian" is now less of an accordian than before. Inserting and deleting of text is propagated to the first TAB but not beyond. Inserting and deleting of TABs is not propagated. It is now easier to delete TABs. Setable TABs similar to those on a typewriter are yet to come.

3. STRINGS

TV-STRINGS.PMAP file is now taken from the log-in directory. It used to come from the connected directory, and has resulted in users' picking up strange strings and damaging their text files with them.

The '(Strings)' request at the beginning of edit is gone, and you can no longer permanently assign a set of strings to a file. This feature was used in only five of over 2500 TV files at IMSSS and SUMEX. String sets addressed by the letters A...Z still exist. $$.R. and $$.D. now refer to the last "letter" set accessed during an editing session, or to the default set if none has been accessed.

4. TV DIRECTORY

TV directory, which aids in locating text pages of a file, used to occupy file page (TENEX page) 777 (octal). This ment one-page overhead for each TV file. For files less than 20 pages, the directory is now stored on the page where the text ends, beyond the EOF pointer, provided that it fits there (in 98% of the files it does). If it does not fit and the file is at least 3 pages the directory is stored on page 777. For files with 20 pages or more (about 5% of TV files have over 20 pages) page 777 is always used as a protection against losing the directory when the file is appended to.

P.S. Due to the fact that a number of programs (SOS among them) read past the end-of-file pointer, this feature has been disabled. It was nice while it lasted.

5-Aug-75
========

CONTENTS

1. NEW COMMANDS

Searching backwards.

$-nT<string>	Backward string search of n lines. Searching proceeds up by lines, and left to right within a line. Backward search takes about 1.5 times as long as forward search.

Additional windowing control.

$n.W	Scroll the screen so as to make the current line the nth from the top. $.W puts current line to the top and $$.W puts it to the bottom.
$-n.W	Make current line the nth from the bottom.

Moving cursor past words to beginnings of blank fields. A way to get to ends of words.

$n.)	Point to the nth blank field on the right.
$n.(	Point to the nth blank field on the left.

Next two commands offer relief when a line or word delete buffer is full and the user no longer wishes to keep the stored lines/words. These commands only unprotect the saved items. They dissapear beyond recovery when further items are deleted.

$$.R<CR>	Release line delete buffer.
$$.R<DEL>	Release word delete buffer.

Changing file mode (uppercase/lowercase) in the middle of an edit.

$$.H	Shift file mode from the current case (upper or lower) to the opposite. The new case becomes the default for the file (it is remembered in the user setable word of the file descriptor block).

2. CHANGES TO OLD COMMANDS

Mistaken "oops".

$O	When you are entering text and accidentally type $O when intending to type $I your lose your line. You can now do a second oops to get the lost line back (sometimes it comes back on the third).

Joining and breaking of lines.

$nJ	Join current line to the next and insert n spaces in between.
$$J	Join current line to the next leaving leading blanks of the next line alone. Trailing blanks right of the cursor are lost. If cursor poins beyond the last visible character of a line, beginning of the next line is brought there.
$nB	Break current line and insert n spaces in front of the new line. $B means $0B instead of $1B.
$$B	Break the line leaving blanks as they were. This is equivalent to the old $B.
	Also try $-nJ, $-nB, $$-J, and $$-B.

'Up cursor' from top of screen, and 'down cursor' from bottom of screen or at end of file, have been too stingy in the past causing unwanted overtyping.

$^	Up. Screen will scroll if current line is top line. $n^ (and $$^) does not cause scrolling.
<LF>, $\	Down. Screen will scroll if current line is bottom line. Makes a new line at end of file. $n<LF> and $n\ do not.
$<CR>	New line. Makes a new line at end of file. $n<CR> does not.

Inserting and deleting.

$-nI<CR>	Insert n blank lines above the current line and leave cursor on the (old) current line.
$-nD<CR> $-nZ<CR><char>	Delete lines, zap lines. These commands delete lines from above the current line and leave the current line intact.
$$K	Puts the deleted end of the line in the line delete buffer from where it can be retrieved with $R<CR>.

Word mode handling has been rewritten. The commands most affected are:

$nD<DEL>	Delete n words. If cursor points at a blank, the blank area from the current position to the beginning of next word counts as one word.
$0D<DEL>	Delete to end of curren blank field, but do nothing if cursor points at a visible character. This is useful in left-justifying text.
$-nD<DEL> $-nZ<DEL><chr>	Delete words, zap words. These commands delete words on the left and leave the current word intact.

Converting consecutive lines to upper or lower case.

$$.U, $$.L	Again command $A after $$.U or $$.L converts the NEXT line.

3. NEW FEATURES

FILE LAST EDITED. The name of the file last edited is stored in TV-STRINGS.PMAP and offered as the default at the beginning of a new editing session.

WINDOW DIMENSIONS. The height and width of the window last in use is saved in TV-STRINGS.PMAP and restored at the beginning of a new editing session.

4. OLD BUGS

STRING SEARCH. Occasionally T-search would find a word but would display a wrong window. Two cases in which this used to occur have been identified and correcred.

LARGE FILES. TV uses file space between end of file and page 770 (octal) as temporary output space. Filling up of this space has caused the editing to abort rather than allowing orderly finish. Straightening out of the resulting file has been a tidious chore. New version of TV warns the user of insufficient work space and lets him finish the edit.

5-May-75
========

1. Maximum line length is 300, formerly 150.

2. Normal margins are marked with colons (:), long lines with asterisks (*), and indicator lines with the equals sign (=), formerly dots (.), dashes (-), and asterisks (*), respectively.

3. Long file names will not cause the "DIRECTORY FULL" message.

22-Mar-75
=========

1. RANDOM ACCESS.
2. WHAT HAPPENS TO YOUR FILE WHILE EDITING.
3. RECOVERING AFTER CRASH OR ^C EXIT.
4. WAYS TO LOSE A FILE.
5. COMMANDS FOR MAKING A NEW VERSION AT FINISH.
6. SOS FILES.
7. CHANGES TO OLD COMMANDS: $$/, $$", $$.F, $G, $T, $-W, and $=.

1. RANDOM ACCESS.

The main news is RANDOM ACCESS. The sole purpose of random access is to put any part of a text file within fast reach. The speed-up is most noticable with the $nG and $$.F commands, but also $$F and string search are somewhat faster. Furthermore, your most recent editing (since the last $$.F) has now a better chance of surviving a system crash.

Frequent use of $$.F is recommended.

The cost of random access is file (disk) space. One file page (page '777) is maintained as a directory. It tells where in the file the text page marks are located. Disk pages of a random access text file are not usually fully packed since both inserting and deleting of text can bring about wasted space. You should expect the resulting files to be 20% bigger than the old TV-Edit files (which were fully packed and had no page directory) and 10% bigger than SOS files (which have line numbers and other wasted space).

2. WHAT HAPPENS TO YOUR FILE WHILE EDITING.

While editing, the file is kept in a special TV-Edit format, and only the new TV-Edit can properly access it. When editing is finished (with $$F), the format is changed back to normal, making the file available to other programs. An attempt by other programs to read an unfinished file results in a warning message.

*** W A R N I N G ***

If editing is aborted with ^C or a system crash, you must next time edit the file with the NEW TV-Edit. Editing it with the old TV will destroy information necessary for recovering from the crash, resulting in the loss of the entire file!

In the edit-time file format the first 512-word disk page (file page 0) contains the warning message, subsequent disk pages contain the original file as it appeared at the start of the editing session (disk page 0 is moved to the end of the original file to make room for the warning message), updated disk pages are stored just below disk page 770 (octal), and disk pages 775-777 contain the access information to the file. When finished, disk page 777 remains and it tells where the page marks of the text file are located. It is this page directory that makes possible random accessing by TV. The directory is not necessary for reading the file by other programs.

3. RECOVERING AFTER CRASH OR ^C EXIT.

When starting to edit after a crash (or a ^C exit), the editor takes you to the approximate place of the crash and inserts two page marks. The text above the page marks is the most recent, and any duplication of it below is older and hence a candidate for deletion. (Unless the page marks appear at the beginning of the file, the first line below the page marks is the tail end of a line above the page marks.)--The duplication of text results from an attempt to hang onto as much of your most recent editing as possible.

SCRAMBLED WINDOW. Sometimes a crash leaves behind a scrambled window. It is always below the spot where the editor starts after a crash, and it fills exactly one window (23 lines). This scrambled window is marked off by double page marks both above and below, and the lines within are usually out of order. After cleanup, look for a copy of the bottom line just below the page marks.--A scrambled window is produced when the ".WAIT.0.." message appears, but is cleaned up subsequently unless preserved by a system crash. It is a monument to substandard programming.

4. WAYS TO LOSE A FILE.

If a crashed file is edited with another editor, the old TV, for instance, and that editing is finished, or if a crashed file is appended by some program, partial or total loss of the file occurs. The warning message in the beginning of the file is designed to minimize the hazard. The proper procedure is to abort, with CTRL-C, any program that allows you to read the warning message, and edit and finish with the new TV-Edit.

5. COMMANDS FOR MAKING A NEW VERSION AT FINISH.

The edited file can either replace the old version of the file or become a new version at the time of finish. The default is "create new version" for crashed files and files in other than connected directory, "replace old version" otherwise. You can beat the default by giving a count in conjunction with an "F" command:

```
$$1F   Finish, replace old version (results in 1 version).
$$2F   Finish, make a new version (results in 2 versions).
$$1.F  Save-and-continue, clear the new version request.
         --An appropriate time to use this command is after
         a crashed file has been cleaned up and a backup is
         no longer deemed necessary.
$$2.F  Save-and-continue, set the new version request.
         --Use this command as soon as you decide that you
```

```
        want to retain a backup version.  You will then
        be free to forget it when actually finishing.
$$F     Finish, new version generation according to the most
        recent request.
```

$$.F (no count given) leaves the new version request unchanged.

If a new version is generated, the old version is returned to its original form, i.e., as it was at the beginning of the editing session.

6. SOS FILES.

You can choose to retain or delete SOS line numbers when you start to edit. Request to retain or delete comes after the "(Strings)" request in the initial dialogue. The default is to Delete. Regardless of your choice, the file is copied and the original file retained as an old version.--In Read-only mode the SOS line numbers are always displayed.

7. CHANGES TO OLD COMMANDS: $$/, $$", $$.F, $G, $T, $-W, and $=.

DEFINING AND RETRIEVING STRINGS (MODIFIED). Users of strings are familiar with the message ".ALREADY.DEFINED...'Y'.TO.REDIFINE.. etc.". In the new TV you can avoid this message by typing two ESCs before the / or " when redefining a string:

```
$$/     Begin string definition; no hassles, please.
$$"     Define a string according to the text on the
        window; no hassles, please.
```

The double quote (") command no longer takes its string ID from the file. The user has to type it in. When retrieving strings with the "$R <string ID>" command, only the string body is written in the file (the old format was: String ID, string body, $/).--The new format is designed to be used for transporting text from one part of the file to another, or to a different file. The old format was better suited for storing string definitions in a text file. That function has since been taken over by the $$.Dx and $$.Rx commands, and the TV-STRINGS.PMAP file.

OTHER REFINEMENTS.

-- $$.F, $G and $T commands are more economical in updating the display.

-- $-W puts cursor to the top of window if current line moves out of view.

-- Output format of the $= command has been changed.

-- Several old bugs were discovered and replaced with a host of new ones yet to be discoversed.

<DOCUMENTATION> TVEDIT.UPDATE

12-Sept-74
==========

Current changes to TV-Edit deal with:

```
  (i) starting the edit, .PL files;
 (ii) new Datamedia keyboard with EDIT key;
(iii) save-and-continue; and
 (iv) saving and loading of command strings, TV-STRINGS.PMAP files.
```

New commands are:
$+G, $$.F, $$.Dx, $$.D., $$.Rx, $$.R., and $$.R0,
where x stands for a letter.

STARTING

The starting dialog has four questions. Whenever a default option exists, it is displayed to the left of a double slash "//". Use 'CR' or 'ESC' to accept the default. 'ESC' will skip over the remaining questions (it is the expert's way to "get with it"). Use the ? to find out about the possible answers to each question.

The starting questions are:

(FILE NAME). For default options, [confirm] the name with an 'ESC'.

(PAGE.LINE). Place to go to. Default is the place of last finish. A list of places to edit can also be read from page 1 of a specially named file. The place list file for the file "name.ext" is called "#name-ext.PL". The editor command $+G takes through consecutive places in the list.--The SEARCH program is able to generate appropriate place list files. (See SEARCH.MANUAL for details.)

(MODE). Default is Write mode for files in your own (connected) directory, Read-only for files in other directories. In write mode, the "L" or "U" of last time is the default.

(STRINGS). You can make or break an association between the text file and a set of command strings. No loading of strings takes place here. Loading and saving is done by editor commands. For details, see Section "STRINGS" below.

IMPLEMENTATION. File mode (U or L), the place of last finish, and the assignment of a set of strings is "remembered" in the user setable word of the File Descriptor Block (FDB). This word is updated only when editing in write mode. Note also thet in COPYing the file or rewriting it by some other program (e.g., in switching between different editors), this information is not copied.

EDIT KEY

The EDIT key is used to type commands. Instead of typing an 'ESC' and then a command letter, type the command letter while depressing the EDIT key. Commands that used to require the typing of two ESCs can now be typed with one ESC and the EDIT key.

SAVE-AND-CONTINUE

$$.F will finish writing the output file, restart the edit, and return to the place where the $$.F command was typed while ".YOU.WILL.HAVE.TO.WAIT.". This gives some protection against system crash.

STRINGS

Commands now exist for saving and loading of all strings at once. Sets of strings are identified by the letters A,...,Z. The commands for saving and loading a set are (x stands for a letter):

SAVING (*)

```
$$.Dx   Strings currently in memory are filed (Dumped) under  x.
$$.D.   Strings in memory are filed in the currently assigned set.
```

LOADING (**)

```
$$.Rx   Load (Read) strings from set  x.
$$.R.   Load currently assigned strings.
$$.R0   Clear strings in memory.
```

NOTES:

(*) In saving, the set in the file is REPLACED by the set currently in memory.

(**) In loading, the set in file is MERGED to the set currently in memory. Thus a string in memory is redefined only if it is defined in the set being loaded from file.

AUTOMATIC SAVING AND LOADING. In addition to the sets of strings identified by the letters, there is a set called the "last set". Strings in memory are saved as the last set whenever editing is FINISHED. The last set is automatically loaded when the editor is STARTED. The $$.Dx and $$.Rx commands refer to the last set if x is not a letter or a special character (non-0 digit is ok.).

IMPLEMENTATION. A special file 'TV-STRINGS.PMAP' is created in your directory. The letters refer to TENEX pages of this file. The letter A, for instance, refers to page 101 octal, the "last set" to page 0. The strings are stored in the TV-STRINGS.PMAP file in a special TV-Edit format (which is not standard text format). The file cannot be read using the TYPE command.

DELETING PAGES of the 'TV-STRINGS.PMAP'-FILE. If $$.Dx command is given when there are no strings in memory (you have just executed the $$.R0 command), page x of the string file is removed from the file. This is a way to delete unnecessary pages from the string file.

There is no point in trying to delete the file. It comes back every time when you use TV-Edit.