

7/9/70
J. Morris

Title: The Text Editor

A. Entering the editor

The editor is used to create and modify text files.
It is called by typing

C, EDITOR, S, <fname>, <user-name>

where <fname>, <user-name> is the file to be edited.

As usual, if ~~<fname>~~^{<user-name>} is omitted the last declared one is assumed. If <fname> is omitted NULL is assumed.

In any case if the file does not previously exist an empty file is created.

After the EDITOR has typed EDIT, one may type a variety of editing commands. Eventually, one types

F, <fname>, <user name>

which will return control to the BEAD after a revised version of the file is written under the name given. If no <fname> is given the original file is changed. There will be no change to the file until the F command is given.

B. Editing commands in general.

At any time in the editing process the editor is positioned at a particular line of the text file.

The following commands may be typed to move about the file while doing certain things. Each command is terminated by a carriage return or semi-colon.

T Move to the tope of the file

Mn Move n lines forward

Pn Print n lines

Dn delete n lines

I Insert the following lines.

C/<S₁>/<S₂>/n Replace S₁ by S₂ throughout the next n lines

En Edit next n lines, using the line editor.

R<fname>, <user-name> Read in the named. file.

W<from>, <user-name>, n Write n lines onto the named file

All those editing commands involving the parameter *n* cause the editor to move forward in the file as it performs the task. If the *n* is omitted 1 is assumed.

Instead of a number a string of characters to search for may be given. A slash (/) followed by a string, calls for the action to be repeated until a line beginning with the string (ignoring leading blanks) is reached. A dot (.) followed by a string calls for the action to be repeated until a line containing the string anywhere is found. These two kinds of search are sometimes called anchored and unanchored search, respectively.

Finally, if *n* is the character \$, the movement proceeds to the end of the file.

Examples

P prints one line, doesn't move file
M1 moves one line forward
M/PIG moves forward to line starting with PIG
P.AL prints through a line containing AL
T;P\$ prints the whole file.

C. Details of the editing commands

T Top

This command moves the editor to the ~~top of~~ beginning of the file. During editing the file always begins with an empty line. This line, called a pseudo-line cannot be deleted and is not included in the final copy of the file. Its only role is to act as a line one can insert something after to put a line at the beginning of the file.

Mn Move

This command moves the editor forward in the file until the condition is met. If *n* is a number the condition is that *n* lines have been passed. If *n* is /<string> The conditions are that the line begin with <string> (ignoring leading blanks). If *n* is .<string> the condition is that the line contain <string> somewhere

If n is $\$$ the condition is that the last line of the file be reached. In the first three cases, if the end of the file is reached before the condition is satisfied, scanning proceeds at the top of the file. If the editor ever returns to the line it started from, it discontinues searching and prints `*NOT FOUND*`. Move is the only command that searches in this end-around manner; all others stop at the end of the file and print `*BOTTOM*`.

`Pn` Print

n lines are printed until the condition is matches. The editor is left at the last line printed.

`Dn` Delete

n lines are deleted from the file, starting with the current one. The n lines are replaced by a single pseudo-line which prints as `*DELETED*`, but disappears as soon as the next file movement occurs.

`I` Insert

This command must be followed immediately by a carriage return (i.e. not a semi-colon). All lines subsequently typed are inserted in the file at the current position. Typing a carriage return at the beginning of a line returns the editor to editing mode; i.e. the next line typed is interpreted as a command.

`C/<S1>/<S2>/ n` Change

This command allows one to change lines without retyping the whole line. The first occurrence (if it exists) of string $\langle S_1 \rangle$ is replaced by string $\langle S_2 \rangle$ in each line considered. The diagnostic `*NO CHANGE*` is printed if the file was not changed. Any character not contained in $\langle S_1 \rangle$ or $\langle S_2 \rangle$ may be used instead of the `/`. When the condition n , is a string match the matching is done before any changing.

CG/<S₁>/<S₂>/n Change Global

This command is just like C ., except that all occurrences of <S₁> are replaced by <S₂>

E n Edit

This command offers another way to change lines. The current line is treated as if it had just been typed in and the user is expected to use the line editor conventions (see the Line Editor in the Internals Manual) to specify a new one which will replace it.

R,<fname>, <user-name> Read

The entire file designated is inserted in the current text file at the Current position.

W,<fname>, <user-name>,n Write ^{the}

The designated file is written as all [^]lines, starting with the current one through the one designated by n .

Q Quit (Not implemented 7/9/70)

Ceases the editing process without writing a file.

D. Files

The editor works by copying files in a purely sequential fashion. At any point one file is being read as input and a new one is being created with those modifications called for by the editing commands. A T (top) command causes the editor to copy the remainder of the input file to the output file, to make the output file the new input, and start creating a new output file. An M command which moves off the end of the file also causes this activity. The two scratch files used for this juggling process are given the names 1<fname> and 2<fname> where <fname> is the name of the original file. In the event of a minor disaster (e.g., you deleted half the file inadvertently), a fairly recent version of the file may be found under one of these names. The procedure to follow is

- (1) Panic (Control-shift P) out of the editor
- (2) Do a DLIST to see what files are around

- (3) Edit the files 1<fname> and/or 2<fname> to see which one is the latest complete version.

The file manipulation procedure, in general, goes like this. If the user enters the editor by typing

C, EDITOR, S, BLATZ

- 1) Start reading BLATZ and writing 1BLATZ
- 2) Then read 1BLATZ and write 2BLATZ
- 3) Then read 2BLATZ and write 1BLATZ, etc.
- 4) In the end, rename^a the most recent file written (1BLATZ or 2BLATZ) and delete the other two files.