PDP-1 COMPUTER
ELECTRICAL ENGINEERING DEPARTMENT
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
CAMBRIDGE, MASSACHUSETTS 02139

PDP-22

EXPENSIVE TYPEWRITER

August 1, 1972

EXPENSIVE TYPEWRITER (ET) is a PDP-1 program that facilitates text editing on-line. The program is controlled from the console typewriter. It can accept additions and corrections from the typewriter. Using the DECTAPE FILE SYSTEM (see memo PDP-42), files may be read into the ET text buffer or the buffer may be filed onto DECTAPE. ET can also read paper tape or punch out the buffer contents onto paper tape. ET directly provides the assembler CERTAINLY with the source program to be assembled.

Operating Procedure

Normally ET is brought in by typing E to ID, which is reached when the console is turned on or when the call button is depressed. If the user had no text buffer, an empty one is created. Otherwise the old buffer contents are retained and the user may continue editing where he left off.

ET stores the text buffer on drum field 2. (Drum field 1 is normally used for the storage of binary programs). For an average program of up to around twenty pages of text, field 2 is sufficient. Another field is required for each additional twenty pages or so. Estimates assume assembler code as text. Fields are assigned sequentially, starting with field 3 for additional text storage. ET assigns and deassigns fields automatically. Only the fields that were previously used for storage of text can be dismissed, however. Thus fields that are assigned to the console but are not used by ET remain unchanged. If ET cannot get a field when it needs one, it will type "field n?" where n is the field number which it needs. It is then the user's problem to get it. Typing any character causes ET to try again.

ET assigns and deassigns the reader automatically for the r command. ET assigns the punch for any operation involving punching and does not dismiss it, except after the P command, until x is typed or ET is left by a normal exit (i.e. by any means other than hitting call or logging out). In either case, if the device is not available, ET types "busy" and the operation is not executed.

ET has one all purpose error comment, "?". If this is printed, in general, nothing will have been disturbed and the user should carefully examine what he had just typed, before trying again.

EXPENSIVE TYPEWRITER has two operating modes, control mode and text mode. In the control mode, characters typed in are commands to ET. In the text mode, typed input is copied directly into the text buffer. The only exception to this is the character backspace, which in the text mode has the effect of cancelling the previous character. Note that tabs and case shifts are characters. One backspace will delete one tab or one case shift.

To go from text mode to control mode, type a backspace immediately following a carriage return or after deleting all characters from a line. Certain commands in the control mode put ET into the text mode. In control mode the typewriter prints in red, and in the text mode it prints in black.

Most lower case commands to ET are not acted upon until the command is terminated by typing in a carriage return or a tab. At any time before the terminator is typed, the command can be cancelled by typing the character centerdot. All capital letter commands to ET are acted upon instantly and do not wait for a terminator.

Many commands need a numeric argument. This may be a decimal integer or one of the following characters:

/ (slash)
. (period)
" (double quote)
' (single quote)
→ (right arrow)
⊃ (imply)

[these characters are explained later].

The search commands, $\overline{S}$, $\underline{S}$, <, and > [explained later] may also be used as numeric arguments before certain commands. The argument may be an expression made up of search commands, decimal integers or the characters, separated by space or + to add, or - to subtract. Arguments typed together with no separator will be added, unless they are both numbers. Certain commands may take two arguments; these appear before command letter, separated from each other by a comma.

Comments may be typed at any time in the control mode. A comment is initiated by a left (open) parenthesis, and ended by the first matching right (close) parenthesis. The material within the parentheses is completely ignored.

The buffer is intended to hold an arbitrary amount of typescript, divided into pages, with each page organized as a list of lines. Each line has an address within its page which is a decimal number. The third line on a page, for example, is called line 3 when examining that page, and cannot be reached when examining any other page. Use of search commands and the character period make it unnecessary, in most cases, to be concerned with actual line numbers. There does not need to be anything typed on a line; each carriage return ends a line. All lines in the buffer must end with a carriage return.

Special Characters

Control Character	Action
.	(period) Represents the number of the line most recently referred to, i.e., the current line.
/	(slash) Has the value of the number of lines on the current page, i.e., the number of the last line on the page.
ↄ	(imply) Has the value of the number of the current page.
→	(right arrow) Has the value of the number of pages in the buffer, i.e., the number of the last page.
"	(double quote) Has the value of the number of characters from the beginning of the line at which the most recent search command found a match to the first character of the match.
'	(single quote) Has the value of the number of characters from the beginning of the line at which the last search command found a match to the last character of the match.

In the following examples, the letters m and n preceding the commands are used to signify numeric expressions used as arguments.

ne	(equals) The letter e , or equivalently = , causes the value of the numeric expression n to be typed out as a decimal integer.

Input Commands

a

(append) The following typed text is added onto the end of the current page.

A

(append) The following text is appended at the end of the buffer on a new page.

ni

(insert) The text following, which can consist of any number of lines, is inserted on the current page immediately before line n. That is, the first line of the inserted text becomes the new line n. The text is squeezed in between the old lines n-1 and n. No material is lost.

nc

(change) Line n is deleted and the text following the c command is inserted in its place.

m,nc

(change) Lines m through n inclusive are deleted and the following text is inserted where they were. The number of lines inserted need not be equal to the number of lines removed.

r

(read) The paper tape in the photoelectric reader is read in and appended to a new page of the buffer. Blank tape is ignored. Parity is checked and characters with erroneous parity cause an informative printout. The character is placed in the buffer regardless of whether or not it has proper parity; however parity is ignored if Sense Switch 6 is on. Regardless of parity checking, deleted characters, those with the 7th hole punched, are ignored. The expected format for tapes is to have a stop code, octal code 13, on the tape at the end of each page. The r command stops reading and returns control to the typewriter upon encountering the end of the tape. If there is no stop code on the end of the tape, one will be supplied. If the buffer is empty, i.e., contains one page with zero lines, at the time the command is given, that page will be deleted first.

nr

(read) Read n pages of text into the buffer.

Deletion of Information in the Buffer

K

(kill) The entire buffer is deleted.

nk

(kill) Starting with the current page, n pages are deleted. The next page becomes current. If the last page is deleted, page 1 becomes current and the command terminates, even if n was too large.

nd

(delete) Line n is removed.

m,nd

(delete) Lines m through n are removed.

Printout Commands

w

(write) The current page is printed out, in black.

nw

(write) N pages, starting with the current page, are printed out, in black.

W

(write) The entire buffer is printed out, in black.

nl

(line) Line n is printed out, in red.

m,nl

(line) Lines m through n are printed out, in red.

backspace

Prints out the next line, has the same effect as .+1l (Note that in text mode, backspace has the effect of cancelling the previous character, or if at the left margin by backspacing or a carriage return, the mode is changed to control mode.)

↑

(line) Prints out the previous line; has the same effect as .-1l.

nL

(lines) This sets the maximum number of lines that will be printed or punched on any page. If n is 0 or absent, there is no limit on the number of lines that may be printed or punched on a page. The parameter set by L initially has the value 60.

nS

(size) This sets the total length of the page in lines. This length is effective only if the last L command had a non-zero argument. The parameter set by S initially has the value 66.

The parameter set by L must be at least as large as the parameters set by S. If an S or L command is given which violates this condition, the command is executed and then the other parameter is adjusted to make the above condition true.

Note: Printout can be terminated at any point by turning on Sense Switch 1.

Punching Paper Tape

P

(punch) The entire buffer is punched on tape with correct parity. The buffer is not modified in any way by this operation. A stop code is punched after each page. The first non-blank line is title punched unless Sense Switch 6 is up.

nt

(punch) Starting with this page, n pages are successively punched with correct parity. A stop code is punched after each page.

T

(title punch) All characters up to the next carriage return are punched in "readable" format. The delete hole is punched so that the title will be ignored when the tape is read back in.

nf

(feed) This feeds n characters of blank tape. If n is larger than 256, only 256 lines are fed.

x

(dismiss) The punch assignment, if any, is dismissed.

Note: Punching can be terminated at any time by turning on Sense Switch 1.

Page Control

n

(next) The next page becomes current.

mn

(next) Move m pages forward in the buffer.

j

(jump) The first page in the buffer becomes current.

nj

(jump) Page n becomes the current page.

o

(combine) This page is combined with the following one.

no

(combine) This page is combined with the following n pages.

nv

(divide) A stop code is inserted before line n , thereby dividing the current page into two pages. The first of these becomes the current page.

Search Commands

S‾

(search limited) Search for S ,where S is an arbitrary string of characters not containing the break character (normally ‾(overbar)) on the current page.

nS‾

(search limited) Search for S on current page beginning at line n .

S_

(search unlimited) The entire buffer will be searched for the string S. The page and line where the string is found become current.

nS_

(search unlimited) Search the entire buffer beginning at line n of the current page.

n<

(continue search) The above search commands store the string S in a special buffer. The command n< continues the search for that string on the current page, starting on line n of the current page. The special buffer is destroyed by the same commands which destroy the special buffer for command "u" (q.v.).

<

Same as .+1<.

n>
and > Similar to n< and <, but the entire buffer is searched.

On all search commands:

The line where the match is found immediately becomes the current line. The entire search command is then treated as if the current line number had been typed in its place, and may be used in numeric expressions.

The string must be preceded by an ‾ (overbar) for a limited search or an _(underbar) for an unlimited search.

The number of characters from the beginning of the line to the beginning of the match is made the value of " (double quote).

The number of characters from the beginning of the line to the end of the match, even if it is on another line, is made the value of ' (single quote).

For example if the fifth line of the current page is

abc, _ lac xyz

typing xyz= will cause ET to set the current line to 5 set the value of " to 9, set the value of ' to 12, and type 5.

Typing ‾abc,T will cause the line to be printed.

B

(break) To facilitate searching for strings containing an ↑ (overbar), the break character can be changed by typing B followed by the new break character.

?

(question mark) Print the current break character.

Note: The search may be terminated at the end of the current page by turning up Sense Switch 1.

Miscellaneous Commands

nu

(put) This stores line n in a special buffer, which is changed constructively only by u and searches, and is destroyed by commands A, a, c, d, i, r, v, ", and _ .

m,nu

(put) Puts lines m through n into the special buffer.

ng

(get) This inserts the special buffer mentioned above before line n in a manner similiar to the command i.

m,nqS"

(change) The m through n characters after the beginning of the current line are replaced by the character string S that is typed in. The string is terminated by the break character, not by backspace. The string S may contain carriage returns and the string that it replaces may contain carriage returns. If fewer than two arguments are given, the missing ones will be taken as zero, so that nq (one argument) will change the first n characters on the current line, and q (no arguments) will insert its following string at the beginning of the current line.

For example,
2qfoo" changes the first 2 characters of the current line to foo.
qbarf" inserts the word barf at the beginning of the current line.

zS"

(change) The string in the position found during the last search and on the current line, is changed to the string S. This is equivalent to ",'qS".

For example,
foozbar" changes the first foo on the current page to bar.

X

(exchange) The user can have two text buffers, the main one occupying drum fields 2, 3, etc., and an auxiliary one occupying fields 7, 10, etc. All editing commands refer to the main text buffer. The command X physically exchanges the two text buffers, so that what was the auxiliary buffer becomes the main buffer occupying fields 2, 3, etc., and vice versa. If no auxiliary buffer exists, an empty one is created.

E

Load a new copy of ET. All parameters are reset to their initial values. The text buffer is unchanged.

Q

Invert the state of line number printing. Line numbers are normally not printed.

Transfer Commands

nF

(file) Causes the FILE SYSTEM on DECTAPE transport n to be loaded and started (see memo PDP-42). This command can also be used to call the FORTRAN compiler from tape transport n.

N

(Nightmare) Transfers control to the Nightmare version of the assembler CERTAINLY, which gets the source program directly from ET's text buffer. The resulting binary program is placed on drum field 1, the program is brought into the users core memory, ID is provided with the symbol table and control is returned to ID, where a P (proceed) command will start the newly assembled program running.

M

(Merged assembly) Transfer control to the merging version of CERTAINLY. CERTAINLY will listen for typed commands. (For a description of these commands, see memo PDP-45, CERTAINLY).

C

(Calculate, compile, or whatever) Transfers control to location 102 of whatever program the user has stored on drum field 15. This command is the same as the command 13C.

nC

Transfers control to location 102 of the program stored on drum field n. Note that n, like any other number typed to ET, is taken as decimal, as opposed to the usual octal convention for numbering drum fields.

b

(back) This causes control to be returned to ID.

Iteration

n[

(iterate) ET starts saving all characters that are typed. While this is happening, commands are executed normally.

]

(end of iteration range) This command is only valid if preceeded by the [(open bracket) command. When this command is typed, ET repeats the commands typed since the preceding [n-1 times. If n is absent, it is assumed to be 262144. The occurrence of any error will cause iteration to be stopped. Iteration may also be stopped at any time by raising Sense Switch 1.

For example,
[foozbar1] will cause all occurrences of the string foo to be replaced by the string bar and every line that is changed to be printed. Note that the typewriter paper will not look quite like the example given above since the first printing will be done immediately after the tab following the 1 command is typed.

nI

(iterate) ET will execute the last command in [ ]'s, n times.

Sense Switches

SS1

Stops printout, punchout, searching, or iteration at any time.

SS3

Causes backspace to be treated like any other character in text mode, i. e., it is inserted in the text and does not cause any deletion.

SS6

Causes parity checking to be suppressed when reading paper tape. Deleted characters are still ignored. Suppresses title punch with the P command.

SUMMARY OF COMMANDS TO EXPENSIVE TYPEWRITER

a	append to end of current page
A	append on new page at end of text
b	back to ID
B	set break character to immediately following character
nc	change line n to typed text
m,nc	change lines m through n to typed text
nC	start user program
nd	delete line n
m,nd	delete lines m through n
ne	type value of n (decimal)
E	get a fresh copy of ET
nf	feed n lines of paper tape
nF	enter file system on tape n
ng	get special buffer and insert before line n
ni	insert typed text before line n
nI	execute last [command]
J	JUMP TO PAGE 4
NJ	JUMP TO PAGE N
K	KILL THIS PAGE
NK	KILL N PAGES BEGINNING WITH THIS ONE
K	

SUMMARY OF COMMANDS TO EXPENSIVE TYPEWRITER

a	append to end of current page
A	append on new page at end of text
b	back to ID
B	set break character to immediately following character
nc	change line n to typed text
m,nc	change lines m through n to typed text
nC	start user program
nd	delete line n
m,nd	delete lines m through n
ne	type value of n (decimal)
E	get a fresh copy of ET
nf	feed n lines of paper tape
nF	enter file system on tape n
ng	get special buffer and insert before line n
ni	insert typed text before line n
nI	execute last [command]
j	jump to page 1
nj	jump to page n
k	kill this page
nk	kill n pages beginning with this one
K	kill entire buffer
nl	print line n
m,nl	print lines m through n
nL	set lines per page
M	merged assembly
n	next page
nn	jump forward n pages
N	clear field 1, then assemble
o	combine this page with next
no	combine this page with next n pages
P	punch everything
m,nq	change characters m to n of current line to typed text
Q	invert state of printing line numbers
r	read entire paper tape
nr	read n pages from paper tape
nS	set total page size
t	punch this page
nt	punch n pages, beginning with this one
T	title punch following line
nu	put line n into special buffer
m,nu	put lines m through n into special buffer
nv	divide page before line n
w	print current page
nw	print n pages, beginning with this one
W	print everything
x	dismiss punch
X	exchange text buffers
z	replace characters in last search with typed text (same as ",'q)

↑	print preceding line
backspace	text mode - delete last character, or, if at left margin, return to control mode control mode - print next line
n[	start of iteration range
]	end of iteration range, begin iteration
S	search for S on current page
nS	search for S on current page, beginning at line n
S	search for S in entire buffer
nS	search for S in entire buffer, beginning at line n of current page
<	Continue search on current page
>	Continue search in entire buffer
•	(centerdot) cancel command
/	number of last line on page
.	number of last line typed (current line)
→	number of last page
ↄ	number of current page
"	number of first character of match in last search
'	number of last character of match + 1
space	add
+	add
-	subtract
n=	print value of n (decimal)
(	begin a comment
)	end a comment
tab or c.r.	terminate and execute lower case command
?	print break character
SS1	stop printout, punchout, searching, or iteration
SS3	cause backspace to be quoted in text mode
SS6	suppress parity check when reading paper tape
t	