

DIGITAL RESEARCH

Post Office Box 579, Pacific Grove, California 93950, (408) 373-3403

CP/M CONTEXT EDITOR (ED)

CP/M VERSION _____

COPYRIGHT © 1976

DIGITAL RESEARCH

P. O. BOX 579

PACIFIC GROVE, CA. 93950

SER. # _____

```

1> /* ED , THE CP/M CONTEXT EDITOR */
2> /* COMMANDS */
3> 4>100H: DECLARE B005H, BOOT LITERALLY '0';
4>101H: DECLARE B005H, BOOT LITERALLY '0';
5> /* COPYRIGHT (C) 1976
6> DIGITAL RESEARCH
7> BOX 579 PACIFIC GROVE
8> CALIFORNIA 93950
9> */
10>.DECLARE COPYRIGHT DATA ('COPYRIGHT (C) 1976, DIGITAL RESEARCH')
11>
12> /* COMMAND      FUNCTION
13> -----      -----
14> A      APPEND LINES OF TEXT TO BUFFER      COPYRIGHT © 1976
15> B      MOVE TO BEGINNING OR END OF TEXT      DIGITAL RESEARCH
16> C      SKIP CHARACTERS
17>
18> D      DELETE CHARACTERS
19> E      END OF EDIT
20> F      FIND STRING IN CURRENT BUFFER      P. O. BOX 579
21> H      MOVE TO TOP OF FILE (HEAD)
22> I      INSERT CHARACTERS FROM KEYBOARD
23> J      UP TO NEXT <ENDFILE>
24> K      JUXTAPOSITION OPERATION - SEARCH FOR FIRST STRING,
25>       INSERT SECOND STRING, DELETE UNTIL THIRD STRING
26> L      DELETE LINES
27> M      SKIP LINES
28> N      MACRO DEFINITION (SEE COMMENT BELOW)
29>       FIND NEXT OCCURRENCE OF STRING
30>       WITH AUTO SCAN THROUGH FILE
31> O      RE-EDIT OLD FILE
32> P      PAGE AND DISPLAY (MOVES UP OR DOWN 24 LINES AND
33>       DISPLAYS 24 LINES)
34> Q      QUIT EDIT WITHOUT UPDATING THE FILE
35> R<FILENAME> READ FROM FILE <FILENAME>.LIB UNTIL <ENDFILE> AND
36>       INSERT INTO TEXT
37> S      SEARCH FOR FIRST STRING, REPLACE BY SECOND STRING
38> T      TYPE LINES
39> U      MOVE UP LINES (SAME AS -NP)
40> W      WRITE LINES OF TEXT TO FILE
41> Z      SLEEP FOR 1/2 SECOND (USED IN MACROS TO STOP DISPLAY)
42> <CR>   MOVE UP OR DOWN AND PRINT ONE LINE
43>
44> A NUMBER OF COMMANDS CAN BE PRECEDED BY A POSITIVE OR
45> NEGATIVE INTEGER BETWEEN 0 AND 65535 (1 IS DEFAULT IF NO VALUE
46> IS SPECIFIED). THIS VALUE DETERMINES THE NUMBER OF TIMES THE
47> COMMAND IS APPLIED BEFORE RETURNING FOR ANOTHER COMMAND.
48> THE COMMANDS
49> C D K L T P U <CR>
50> CAN BE PRECEDED BY AN UNSIGNED, POSITIVE, OR NEGATIVE NUMBER.
51> THE COMMANDS
52> A F J H W Z
53> CAN BE PRECEDED BY AN UNSIGNED OR POSITIVE NUMBER.
54> THE COMMANDS
55> E H O Q
56> CANNOT BE PRECEDED BY A NUMBER. THE COMMANDS
57> F I J M R S
58> ARE ALL FOLLOWED BY ONE OR MORE STRINGS OF CHARACTERS WHICH CAN
59> BE OPTIONALLY SEPARATED OR TERMINATED BY EITHER <ENDFILE> OR <CR>.
60> THE <ENDFILE> IS GENERALLY USED TO SEPARATE THE SEARCH STRINGS.

```

CP/M VERSION	

A	COPYRIGHT © 1976
B	DIGITAL RESEARCH
C	P. O. BOX 579
D	PACIFIC GROVE, CA 93950
E	SER. # ED 1.0

61> IN THE S AND J COMMANDS, AND IS USED AT THE END OF THE COMMANDS IF
62> ADDITIONAL COMMANDS FOLLOW. FOR EXAMPLE, THE FOLLOWING COMMAND
63> SEQUENCE SEARCHES FOR THE STRING 'GAMMA', SUBSTITUTES THE STRING
64> 'DELTA', AND THEN TYPES THE FIRST PART OF THE LINE WHERE THE
65> CHANGE OCCURRED, FOLLOWED BY THE REMAINDER OF THE LINE WHICH WAS
66> CHANGED.
67> SCAMMA<ENDFILE>HDELTAK<ENDFILE>BTT<CR>

68>

69> THE CONTROL-L CHARACTER IN SEARCH AND SUBSTITUTE STRINGS IS
70> REPLACED ON INPUT BY <CR><LF> CHARACTERS. THE CONTROL-I KEY
71> IS TAKEN AS A TAB CHARACTER.

72>

73> THE COMMAND R MUST BE FOLLOWED BY A FILE NAME (WITH ASSUMED FILE
74> TYPE OF 'LIB') WITH A TRAILING <CR> OR <ENDFILE>. THE COMMAND
75> I IS FOLLOWED BY A STRING OF SYMBOLS TO INSERT, TERMINATED BY
76> A <CR> OR <ENDFILE>. IF SEVERAL LINES OF TEXT ARE TO BE INSERTED,
77> THE I CAN BE DIRECTLY FOLLOWED BY AN <ENDFILE> OR <CR> IN WHICH CASE
78> THE EDITOR ACCEPTS LINES OF INPUT TO THE NEXT <ENDFILE>.
79> THE COMMAND OT PRINTS THE FIRST PART OF THE CURRENT LINE,
80> AND THE COMMAND OL MOVES THE REFERENCE TO THE BEGINNING OF THE
81> CURRENT LINE. THE COMMAND BP PRINTS THE CURRENT PAGE ONLY, WHILE
82> THE COMMAND OZ READS THE CONSOLE RATHER THAN WAITING (THIS IS USED
83> AGAIN WITHIN MACROS TO STOP THE DISPLAY - THE MACRO EXPANSION
84> STOPS UNTIL A CHARACTER IS READ. IF THE CHARACTER IS NOT A BREAK
85> THEN THE MACRO EXPANSION CONTINUES NORMALLY).

86>

87> NOTE THAT A POUND SIGN IS TAKEN AS THE NUMBER 65535, ALL
88> UNSIGNED NUMBERS ARE ASSUMED POSITIVE, AND A SINGLE - IS ASSUMED -1

89>

90> A NUMBER OF COMMANDS CAN BE GROUPED TOGETHER AND EXECUTED
91> REPETITIVELY USING THE MACRO COMMAND WHICH TAKES THE FORM
92> <NUMBER>MC1C2...CN<DELIMITER>

93>

94>

95> WHERE <NUMBER> IS A NON-NEGATIVE INTEGER N, AND <DELIMITER> IS
96> <ENDFILE> OR <CR>. THE COMMANDS C1 ... CN FOLLOWING THE M ARE
97> EXECUTED N TIMES, STARTING AT THE CURRENT POSITION IN THE BUFFER.
98> IF N IS 0, 1, OR OMITTED, THE COMMANDS ARE EXECUTED UNTIL THE END
99> IF THE BUFFER IS ENCOUNTERED.

100>

101> THE FOLLOWING MACRO, FOR EXAMPLE, CHANGES ALL OCCURRENCES OF
102> THE NAME 'GAMMA' TO 'DELTA', AND PRINTS THE LINES WHICH
103> WERE CHANGED.

104>

105> MFGAMMA<ENDFILE>-5DIDELTAK<ENDFILE>BLT<CR>

106>

107> (NOTE: AN <ENDFILE> IS THE CP/M END OF FILE MARK - CONTROL-Z)

108>

109> IF ANY KEY IS DEPRESSED DURING TYPING OR MACRO EXPANSION, THE
110> FUNCTION IS CONSIDERED TERMINATED, AND CONTROL RETURNS TO THE
111> OPERATOR.

112>

113> ERROR CONDITIONS ARE INDICATED BY PRINTING ONE OF THE CHARACTERS,

114>

SYMBOL	ERROR CONDITION
-----	-----
GREATER	FREE MEMORY IS EXHAUSTED - ANY COMMAND CAN BE ISSUED WHICH DOES NOT INCREASE MEMORY REQUIREMENTS.
QUESTION	UNRECOGNIZED COMMAND OR ILLEGAL NUMERIC FIELD
POUND	CANNOT APPLY THE COMMAND THE NUMBER OF TIMES SPECIFIED

115>

116>

117>

118>

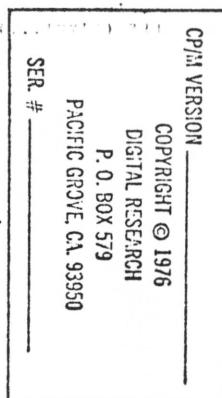
119>

120>

```

121>          (OCCURS IF SEARCH STRING CANNOT BE FOUND).    11  .5242 YT
122> LETTER D  CANNOT OPEN <FILENAME>.LIB IN R COMMAND
123>
124> THE ERROR CHARACTER IS ALSO ACCOMPANIED BY THE LAST CHARACTER
125> SCANNED WHEN THE ERROR OCCURRED.          00000000      */
126>
127>HDECLARE LIT LITERALLY 'LITERALLY',
128>     DCL LIT 'DECLARE',
129>     PROC LIT 'PROCEDURE',
130>     ADDR LIT 'ADDRESS',
131>     CTL LIT '0CH',
132>     CTLU LIT '15H', /* LINE DELETE IN INSERT MODE */
133>     LCA LIT '110$0001B', /* LOWER CASE A */
134>     LZC LIT '111$1010B', /* LOWER CASE Z */
135>     ENDFILE LIT '1AH', /* CP/M END OF FILE */
136>
137>DECLARE MAXA ADDRESS INITIAL(0000H), /* MONITOR LOCATION */
138>     MAX ADDRESS,           /* FILLED FROM MAXA */
139>     MAXM ADDRESS,          /* MINUS 1 */
140>     HMAX ADDRESS,          /* HALF THE VALUE OF MAXM */
141>     MAXB BASED MAXA ADDRESS; /* TEMP TO COMPUTE ADDR */
142>
143>HDECLARE HBUF LITERALLY '7', /* NUMBER OF BUFFERS-1 */
144>     BUFS LITERALLY '1024', /* (HBUF+1) * 128 */
145>     BDISKA ADDRESS INITIAL(4), /* BOOT DISK ADDRESS */
146>     BDISK BASED BDISKA BYTE, /* BOOT DISK VALUE */
147>     FCBA ADDRESS INITIAL(5CH),
148>     FCB BASED FCBA (33) BYTE, /* DEFAULT FCB AT 5CH */
149>     BUFA ADDRESS INITIAL(80H), /* DISK IO BUFFER ADDRESS */
150>     RFCB (33) BYTE /* READER FILE CONTROL BLOCK */
151>     INITIAL(0, /* FILE NAME */
152>             /* FILE TYPE */ 'LIB'),
153>     FPP BYTE,              /* READ BUFFER POINTER */
154>     BUFF BASED BUFA (33) BYTE, /* DISKIO BUFFER */
155>     SFCB BASED FCBA (33) BYTE, /* SOURCE FILE CONTROL */
156>     SEUFF(BUFS) BYTE,        /* SOURCE BUFFER */
157>     DFCB (33) BYTE,         /* DEST FILE CONTROL BLOCK */
158>     DBUF(BUFS) BYTE,        /* DESTINATION BUFFER */
159>     HSOURCE ADDRESS,        /* NEXT SOURCE CHARACTER */
160>     HDEST ADDRESS,         /* NEXT DESTINATION CHAR */
161>
162>>DECLARE SDISK BYTE, /* SOURCE FILE DISK */
163>     DDISK BYTE, /* DESTINATION FILE DISK */
164>     /* IO SECTION */
165>MON1: PROCEDURE(F,A); DECLARE F BYTE, A ADDRESS;
166>     GO TO BDOS;
167>     END MON1;
168>
169>MON2: PROCEDURE(F,A) BYTE; DECLARE F BYTE, A ADDRESS;
170>     GO TO BDOS;
171>     END MON2;
172>
173>READCHAR: PROCEDURE BYTE; RETURN MON2(1,0);
174>     END READCHAR;
175>
176>HDECLARE TRUE LITERALLY '1', FALSE LITERALLY '0';
177>     FOREVER LITERALLY 'WHILE TRUE',
178>     CR LITERALLY '13',
179>     LF LITERALLY '10',
180>     WHAT LITERALLY '63';

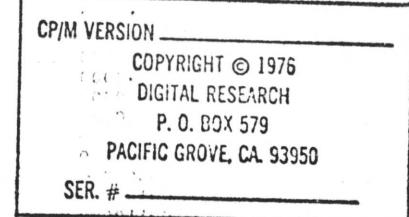
```



```

181>
182>PRINTCHAR: PROCEDURE(CHAR);
183>     DECLARE CHAR BYTE;
184>     CALL MON1(2,CHAR);
185>     END PRINTCHAR;
186>
187>CRLF: PROCEDURE; /* C RLF */
188>     CALL PRINTCHAR(CR); CALL PRINTCHAR(LF);
189>     END CRLF;
190>
191>^PRINT: PROCEDURE(A);
192>     DECLARE A ADDRESS;
193>     CALL CRLF; CALL MON1(9,A);
194>     END PRINT;
195>
196>READ: PROCEDURE(A);
197>     DECLARE A ADDRESS;
198>     CALL MON1(10,A);
199>     END READ;
200>
201>HDECLARE DCNT BYTE;
202>
203>
204>OPEN: PROCEDURE(FCB);
205>     DECLARE FCB ADDRESS;
206>     DCNT = MON2(15,FCB);
207>     END OPEN;
208>
209>CLOSE: PROCEDURE(FCB);
210>     DECLARE FCB ADDRESS;
211>     DCNT = MON2(16,FCB);
212>     END CLOSE;
213>
214>SEARCH: PROCEDURE(FCB);
215>     DECLARE FCB ADDRESS;
216>     DCNT = MON2(17,FCB);
217>     END SEARCH;
218>
219>>DELETE: PROCEDURE(FCB);
220>     DECLARE FCB ADDRESS;
221>     CALL MON1(19,FCB);
222>     END DELETE;
223>
224>HDISKREAD: PROCEDURE(FCB) BYTE;
225>     DECLARE FCB ADDRESS;
226>     RETURN MON2(20,FCB);
227>     END DISKREAD;
228>
229>HDISKWRITE: PROCEDURE(FCB) BYTE;
230>     DECLARE FCB ADDRESS;
231>     RETURN MON2(21,FCB);
232>     END DISKWRITE;
233>
234>MAKE: PROCEDURE(FCB);
235>     DECLARE FCB ADDRESS;
236>     DCNT = MON2(22,FCB);
237>     END MAKE;
238>
239>RENAME: PROCEDURE(FCB);
240>     DECLARE FCB ADDRESS;

```



```

241> CALL MOH1(23,FCB);
242> END RENAME;
243>
244>DECLARE (MAXLEN,COMLEN) BYTE, COMBUFF(128) BYTE,
245>      (TCBP,CBP) BYTE;
246>
247>READCOM: PROCEDURE;
248>   MAXLEN = 128; CALL READ(.MAXLEN);
249> END READCOM;
250>
251>BREAK$KEY: PROCEDURE BYTE;
252>   IF MOH2(1,0) THEN
253>     DO; /* CLEAR CHAR */;
254>     CALL MOH1(1,0); RETURN TRUE;
255>   END;
256>   RETURN FALSE;
257> END BREAK$KEY;
258>
259>CSELECT: PROCEDURE BYTE;
260>   /* RETURN CURRENT DRIVE NUMBER */;
261>   RETURN MOH2(25,0);
262> END CSELECT;
263>
264>SELECT: PROCEDURE(DISK);
265>   DECLARE DISK BYTE;
266>   /* SET DRIVE NUMBER */;
267>   CALL MOH1(14,DISK);
268>   END SELECT;
269>
270>SETDMA: PROCEDURE(A),
271>   DECLARE A ADDRESS;
272>   /* SET DMA ADDRESS */;
273>   CALL MOH1(26,A);
274> END SETDMA;
275>
276>LIFTHEAD: PROCEDURE;
277>   /* LIFT HEAD ON SA 3900 DISK */;
278>   CALL MOH1(12,0);
279> END LIFTHEAD;
280> /* INPUT / OUTPUT BUFFERING ROUTINES */
281>
282>MOVE: PROCEDURE(S,D,N);
283>   DECLARE (S,D) ADDRESS, N BYTE;
284>   DECLARE A BASED S BYTE, B BASED D BYTE;
285>   DO WHILE (N := N - 1) <> 255;
286>     B = A; S = S + 1; D = D + 1;
287>   END;
288> END MOVE;
289>
290>FERR: PROCEDURE;
291>   CALL CRLF;
292>   CALL PRINT (.'FILE ERROR$');
293>   CALL CRLF;
294>   GO TO BOOT;
295> END FERR;
296>
297>SETTYPE: PROCEDURE(A);
298>   DECLARE A ADDRESS;
299>   CALL MOVE(A,DFCB+9,3);
300> END SETTYPE;

```

CP/M VERSION
COPYRIGHT © 1976
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA. 93950
SER. #

```

301>302>SETUP: PROCEDURE;
303>   NSOURCE = LENGTH(SBUFF); NDEST = 0;
304>   SFCB(12) = 0; SFCB(32) = 0;
305>   /* REEL AND RECORD ZEROED */;
306>   /* COPY NAME TO DESTINATION FCB */;
307>   CALL MOVE(FCBA,.DFCB,33);
308>   /* SOURCE AND DESTINATION DISKS SET */;
309>   CALL SELECT(SDISK);
310>   CALL OPEN(FCBA);
311>   IF DCNT = 255 THEN
312>     DO; CALL MAKE(FCBA);
313>     IF DCNT = 255 THEN CALL FERR;
314>     CALL PRINT(.'NEW FILE$');
315>     CALL CRLF;
316>   END;
317>   CALL SETTYPE(.BAK');
318>   CALL DELETE(.DFCB);
319>   IF SDISK <> DDISK THEN
320>     DO; /* REMOVE BAK FILES FROM DESTINATION DISK */;
321>     CALL SELECT(DDISK);
322>     CALL DELETE(.DFCB);
323>   END;
324>   CALL SETTYPE('$$$');
325>   CALL DELETE(.DFCB);
326>   CALL MAKE(.DFCB);
327>   DFCB(32) = 0; /* NEXT RECORD IS ZERO */;
328>   IF DCNT = 255 THEN CALL FERR;
329>   /* THE TEMP FILE IS NOW CREATED */;
330>   END SETUP;
331>
332>FILLSOURCE: PROCEDURE;
333>   DECLARE I BYTE;
334>   ZH: PROCEDURE;
335>     NSOURCE = 0;
336>   END ZH;
337>
338>   CALL ZH;
339>   CALL SELECT(SDISK);
340>   DO I = 0 TO NBUF;
341>     CALL SETIMAC(SBUFF(NSOURCE));
342>     IF (DCNT := DISKREAD(FCBA)) <> 0 THEN
343>       DO; IF DCNT > 1 THEN CALL FERR;
344>       SBUFF(NSOURCE) = ENDFILE;
345>       I = NBUF;
346>     END;
347>   ELSE
348>     NSOURCE = NSOURCE + 80H;
349>   END;
350>   CALL ZH;
351> END FILLSOURCE;
352>
353>GETSOURCE: PROCEDURE BYTE;
354>   DECLARE B BYTE;
355>   IF NSOURCE > = LENGTH(SBUFF) THEN CALL FILLSOURCE;
356>   IF (B := SBUFF(NSOURCE)) <> ENDFILE THEN
357>     NSOURCE = NSOURCE + 1;
358>   RETURN B;
359> END GETSOURCE;
360>

```

CP/M VERSION

COPYRIGHT © 1976
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA. 93950

SER. #

```

361>WRITEDEST: PROCEDURE;
362>    /* WRITE OUTPUT BUFFER UP TO (NOT INCLUDING) NDEST.
363>    LOW 7 BITS OF NDEST ARE ZERO */
364>    DECLARE (I,N) BYTE;
365>    ZH: PROCEDURE;
366>        NDEST = 0;
367>    END ZH;
368>
369>    CALL SELECT(DDISK);
370>    IF LOU((N := SHR(NDEST,7)) - 1) = 255 THEN RETURN;
371>    CALL ZH;
372>        DO I = 0 TO N;
373>            CALL SETDMAC(DBUFF(NDEST));
374>            IF DISKWRITE(.DFCB) <> 0 THEN CALL FERR;
375>            NDEST = NDEST + 128;
376>        END;
377>    CALL ZH;
378>    END WRITEDEST;
379>
380>PUTDEST: PROCEDURE(B);
381>    DECLARE B BYTE;
382>    IF NDEST >= LENGTH(DBUFF) THEN CALL WRITEDEST;
383>    DBUFF(NDEST) = B;
384>    NDEST = NDEST + 1;
385>    END PUTDEST;
386>
387>FINIS: PROCEDURE;
388>    MOVEUP: PROCEDURE;
389>        CALL MOVE(.DFCB,.DFCB+16,16);
390>    END MOVEUP;
391>
392>    /* CLEAR OUTPUT */
393>    DO WHILE (LOU(NDEST) AND 7FH) <> 0;
394>        CALL PUTDEST(ENDFILE);
395>    END;
396>    CALL WRITEDEST;
397>
398>    CALL CLOSE(.DFCB);
399>    IF DCHT = 255 THEN CALL FERR;
400>    /* RENAME OLD FILE TO BAK */
401>    CALL SETTYPE('.BAK'); CALL MOVEUP;
402>    CALL SELECT(DDISK);
403>    CALL MOVE(FCBA,.DFCB,16);
404>    CALL RENAME(.DFCB);
405>    CALL MOVEUP;
406>    /* RENAME $$ TO OLD NAME */
407>    CALL SETTYPE('.$$');
408>    CALL SELECT(DDISK);
409>    CALL RENAME(.DFCB);
410>    END FINIS;
411>
412>
413>NDECLARE
414>    LPP LIT '23';           /* LINES PER PAGE */
415>    FORWARD LIT '1';
416>    BACKWARD LIT '0';
417>    RUBOUT LIT '07FH';
418>    FOUND LIT '23H';
419>    MACSIZE LIT '128';      /* MAX MACRO SIZE */
420>    SCRFSIZE LIT '100';      /* SCRATCH BUFFER SIZE */

```

CP/M VERSION _____
 COPYRIGHT © 1976
 DIGITAL RESEARCH
 P. O. BOX 579
 PACIFIC GROVE, CA. 93950
 SER. # _____

```

421>    COMSIZE LIT 'ADDRESS' /* DETERMINES MAX COMMAND NUMBER*/
422>
423>NDCL MACRO(MACSIZE) BYTE;
424>    SCRATCH(SCRFSIZE) BYTE; /* SCRATCH BUFFER FOR F,N,S */
425>    (UBP, UBE, UBJ) BYTE; /* END OF F STRING, S STRING, J STRING */
426>    (FLAG, MP, MI, XP, COLUMN) BYTE;
427>    MT COMSIZE;
428>
429>NDCL (START, RESTART, OVERCOUNT, OVERFLOW, RESET, BADCOM) LABEL;
430>
431>NDCL INSERTING BYTE; /* TRUE IF INSERTING CHARACTERS */
432>    READBUFF BYTE; /* TRUE IF END OF READ BUFFER */
433>
434>DECLARE
435>    TAB LITERALLY '110';
436>    EOS LITERALLY '0FFH';
437>
438>
439>TTYCHAR: PROCEDURE(CHAR);
440>    DECLARE CHAR BYTE;
441>    IF CHAR >= ' ' THEN COLUMN = COLUMN + 1;
442>    IF CHAR = CR THEN COLUMN = 0;
443>    CALL PRINTCHAR(CHAR);
444>    END TTYCHAR;
445>
446>PRINTC: PROCEDURE(CHAR);
447>    DECLARE (CHAR,I,J) BYTE;
448>    I = CHAR = TAB AND 7 - (COLUMN AND 7);
449>    IF CHAR = TAB THEN CHAR = ' ';
450>    DO J = 0 TO I;
451>        CALL TTYCHAR(CHAR);
452>    END;
453>    END PRINTC;
454>
455>PRINTNMAC: PROCEDURE(CHAR);
456>    DECLARE CHAR BYTE;
457>    /* PRINT IF NOT IN MACRO EXPANSION */
458>    IF MP <> 0 THEN RETURN;
459>    CALL PRINTC(CHAR);
460>    END PRINTNMAC;
461>
462>
463>
464>
465>NDECLARE TRANSLATE BYTE; /* TRUE IF TRANSLATION TO UPPER CASE */
466>    UPPER BYTE; /* TRUE IF GLOBALLY TRANSLATING TO UC */
467>
468>LOWERCASE: PROCEDURE(C) BYTE;
469>    DECLARE C BYTE;
470>    /* RETURN TRUE IF LOWER CASE ALPHABETIC */
471>    RETURN C >= LCA AND C <= LCA;
472>    END LOWERCASE;
473>
474>UTRAN: PROCEDURE(C) BYTE;
475>    DECLARE C BYTE;
476>    /* TRANSLATE TO UPPER CASE IF ALPHABETIC LOWER AND TRANSLATE */
477>    IF TRANSLATE AND LOWERCASE(C) THEN C = C AND 1011111B;
478>    RETURN C;
479>    END UTRAN;
480>

```

CP/M VERSION _____
 COPYRIGHT © 1976
 DIGITAL RESEARCH
 P. O. BOX 579
 PACIFIC GROVE, CA. 93950
 SER. # _____

```

481>READC: PROCEDURE BYTE;
482>    /* MAY BE MACRO EXPANSION */
483>    IF MP > 0 THEN
484>        DO;
485>            IF BREAK$KEY THEN GO TO RESET;
486>            IF XP >= MP THEN
487>                DO; /* START AGAIN */
488>                    IF MT <> 0 THEN
489>                        DO; IF (MT,=MT-1) = 0 THEN
490>                            GO TO RESET;
491>                        END;
492>                        XP = 0;
493>                    END;
494>                    RETURN UTRAN(MACRO((XP := XP + 1) - 1));
495>                END;
496>                IF INSERTING THEN RETURN UTRAN(READCHAR);
497>
498>    /* GET COMMAND LINE */
499>    IF READBUFF THEN
500>        DO; READBUFF = FALSE; CALL LIFTHEAD;
501>        CALL PRINTC('*');
502>        CALL READCOM; CBP = 0;
503>        CALL PRINTC(LF);
504>        COLUMN = 0;
505>    END;
506>    IF (READBUFF := CBP = COMLEN) > THEN COMBUFF(CBP) = CR;
507>    RETURN UTRAN(COMBUFF((CBP := CBP + 1) - 1));
508> END READC;

510>READFILE: PROCEDURE BYTE;
511>    IF RBP >= 68H THEN
512>        DO; CALL SELECT(SDISK);
513>        IF DISKREAD(.RFCB) <> 0 THEN RETURN ENDFILE;
514>        REP = 0;
515>    END;
516>    RETURN UTRAN(BUFF((RBP := RBP + 1) - 1));
517> END READFILE;

519>GETMEM: PROC(I) BYTE;
520>    DCL I ADDR;
521>    RETURN MEMORY(I);
522>    END GETMEM;

524>PUTMEM: PPDG(I,J);
525>    DCL I ADDR, J-BYTE;
526>    MEMORY(I) = J;
527>    END PUTMEM;

529>TRANSFER: PROC(I,J);
530>    DCL (I,J) ADDR;
531>    CALL PUTMEM(I,GETMEM(J));
532>    END TRANSFER;

534>DCL (DISTANCE, TDIST) COMSIZE,
535>    (DIRECTION, CHAR) BYTE,
536>    (FRONT, BACK, FIRST, LAST) ADDR;
537>
538>SETFF: PROCEDURE;
539>    DISTANCE = 0FFFFH;
540>    END SETFF;

```

CP/M VERSION
COPYRIGHT © 1976
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA. 93950
SER. # _____

```

541>
542>NDISTZERO: PROCEDURE BYTE;
543>    /* RETURN TRUE IF DISTANCE IS ZERO */
544>    RETURN DISTANCE = 0;
545>    END DISTZERO;

547>ZERODIST: PROCEDURE;
548>    DISTANCE = 0;
549>    END ZERODIST;

551>NDISTNZERO: PROCEDURE BYTE;
552>    /* CHECK FOR ZERO DISTANCE AND DECREMENT */
553>    IF NOT DISTZERO THEN
554>        DO; DISTANCE = DISTANCE - 1;
555>        RETURN TRUE;
556>    END;
557>    RETURN FALSE;
558>    END NDISTNZERO;

560>SETLIMITS: PROC;
561>    DCL (I,K,L,M) ADDR, (MIDDLE,LOOPING) BYTE;
562>    IF DIRECTION = BACKWARD THEN
563>        DO; DISTANCE = DISTANCE+1; I = FRONT; L = 0; K = 0FFFFH;
564>    END; ELSE
565>        DO; I = BACK; L = MAXM; K = 1;
566>    END;
567>
568>    LOOPING = TRUE;
569>    DO WHILE LOOPING;
570>        DO WHILE (MIDDLE := I <> L) AND (I <> K);
571>            GETMEM(M,I+K) <> LF;
572>            I = M;
573>        END;
574>        LOOPING = (DISTANCE := DISTANCE - 1) <> 0;
575>        IF NOT MIDLINE THEN
576>            DO; LOOPING = FALSE;
577>            I = I - K;
578>        END; ELSE
579>        IF LOOPING THEN I = M;
580>    END;
581>
582>    IF DIRECTION = BACKWARD THEN
583>        DO; FIRST = I; LAST = FRONT - 1;
584>    END; ELSE
585>        DO; FIRST = BACK + 1; LAST = I + 1;
586>    END;
587>    END SETLIMITS;

589>SETPTRS: PROCEDURE;
590>    IF DIRECTION = FORWARD THEN BACK=LAST; ELSE FRONT=FIRST;
591>    END SETPTRS;

594>INCFRONT: PROC; FRONT = FRONT + 1;
595>    END INCFRONT;

596>INCBACK: PROCEDURE; BACK = BACK + 1;
597>    END INCBACK;

598>NDECFRONT: PROC; FRONT = FRONT - 1;
599>    END DECFRONT;

600>NDECBACK: PROC; BACK = BACK - 1;

```

CP/M VERSION

COPYRIGHT © 1976
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA. 93950
SER. # _____

```

601> END DECBACK;
602>
603>MOVER: PROC;
604> IF DIRECTION = FORWARD THEN
605>   DO WHILE BACK < LAST; CALL INCBACK;
606>   CALL TRANSFER(FRONT,BACK); CALL INCFRONT;
607> END; ELSE
608>   DO WHILE FRONT > FIRST; CALL DECFRONT;
609>   CALL TRANSFER(BACK,FRONT); CALL DECBACK;
610> END;
611> END MOVER;
612>
613>MOVELINES: PROC;
614>   CALL SETLIMITS;
615>   CALL MOVER;
616> END MOVELINES;
617>
618>SETCLIMITS: PROC;
619>   IF DIRECTION = BACKWARD THEN
620>     DO; LAST = BACK;
621>     IF DISTANCE > FRONT THEN
622>       FIRST = 1; ELSE FIRST = FRONT - DISTANCE;
623>     END; ELSE
624>       DO; FIRST = FRONT;
625>       IF DISTANCE >= MAX - BACK THEN
626>         LAST = MAXM; ELSE LAST = BACK + DISTANCE;
627>       END;
628>   END SETCLIMITS;
629>
630>READLINE: PROCEDURE;
631>   DECLARE B BYTE;
632>   /* READ ANOTHER LINE OF INPUT */
633>   CTRAN: PROCEDURE(B) BYTE;
634>   DECLARE B BYTE;
635>   /* CONDITIONALLY TRANSLATE TO UPPER CASE ON INPUT */
636>   IF UPPER THEN RETURN UTRAN(B);
637>   RETURN B;
638> END CTRAN;
639> DO FOREVER;
640> IF FRONT >= BACK THEN GO TO OVERFLOW;
641> IF (B := CTRAN(GETSOURCE)) = ENDFILE THEN
642>   DO; CALL ZERODIST; RETURN;
643>   END;
644> CALL PUTMEM(FRONT,B);
645> CALL INCFRONT;
646> IF B = LF THEN RETURN;
647> END;
648> END READLINE;
649>
650>WRITELINE: PROCEDURE;
651>   /* WRITE ONE LINE OUT */
652>   DECLARE B BYTE;
653>   DO FOREVER;
654>   IF BACK >= MAXM THEN /* EMPTY */
655>     DO; CALL ZERODIST; RETURN;
656>   END;
657>   CALL INCBACK;
658>   CALL PUTDEST(B:=GETHEM(BACK));
659>   IF B = LF THEN RETURN;
660> END;

```

CP/M VERSION

COPYRIGHT © 1976
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA. 93950

SER. #

```

661> END WRITELINE;
662>
663>WRHALF: PROCEDURE;
664>   /* WRITE LINES UNTIL AT LEAST HALF THE BUFFER IS EMPTY */
665>   CALL SETFF;
666>   DO WHILE DISTNZERO;
667>     IF HMAX >= (MAXM - BACK) THEN CALL ZERODIST; ELSE
668>       CALL WRITELINE;
669>   END;
670> END WRHALF;
671>
672>WRITEOUT: PROCEDURE;
673>   /* WRITE LINES DETERMINED BY "DISTANCE",
674>      CALLED FROM W AND E COMMANDS */
675>   DIRECTION = BACKWARD; FIRST = 1; LAST = BACK;
676>   CALL MOVER;
677>   IF DISTZERO THEN CALL WRHALF;
678>   /* DISTANCE = 0 IF CALL WRHALF */
679>   DO WHILE DISTZERO;
680>     CALL WRITELINE;
681>   END;
682>   IF BACK < LAST THEN
683>     DO; DIRECTION = FORWARD; CALL MOVER;
684>   END;
685> END WRITEOUT;
686>
687>CLEARMEM: PROCEDURE;
688>   /* CLEAR MEMORY BUFFER */
689>   CALL SETFF;
690>   CALL WRITEOUT;
691> END CLEARMEM;
692>
693>TERMINATE: PROCEDURE;
694>   /* CLEAR BUFFERS */
695>   CALL CLEARMEM;
696>   DO WHILE (CHAR := GETSOURCE) <> ENDFILE;
697>     CALL PUTDEST(CHAR);
698>   END;
699>   CALL FINIS;
700> END TERMINATE;
701>
702>
703>INSERT: PROCEDURE;
704>   /* INSERT CHAR INTO MEMORY BUFFER */
705>   IF FRONT = BACK THEN GO TO OVERFLOW;
706>   CALL PUTMEM(FRONT,CHAR); CALL INCFRONT;
707> END INSERT;
708>
709>SCANNING: PROCEDURE BYTE;
710>   /* READ A CHARACTER AND CHECK FOR ENDFILE OR CR */
711>   RETURN NOT ((CHAR := READC) = ENDFILE OR
712>               (CHAR = CR AND NOT INSERTING));
713> END SCANNING;
714>
715>COLLECT: PROCEDURE;
716>   /* READ COMMAND BUFFER AND INSERT CHARACTERS INTO
717>      SCRATCH 'TIL NEXT CONTROL-Z OR CR FOR FIND, NEXT, JUXT, OR
718>      SUBSTITUTE COMMANDS - FILL AT UBE AND INCREMENT UBE SO IT
719>      ADDRESSES NEXT EMPTY POSITION OF SCRATCH */
720>   SETSCR( PROCEDURE);

```

CP/M VERSION

COPYRIGHT © 1976
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA. 93950

SER. #

```

721> SCRATCH(CUBE) = CHAR;
722> IF (WBE := WBE + 1) > LAST(SCRATCH) THEN GO TO OVERFLOW;
723> END SETSCR;
724> DO WHILE SCANNING;
725>   IF CHAR = CTLN THEN
726>     DO; CHAR = CR; CALL SETSCR;
727>     CHAR = LF;
728>   END;
729>   IF CHAR = @ THEN GO TO RESET;
730>   CALL SETSCR;
731> END;
732> END COLLECT;
733>
734> FIND: PROCEDURE(PA,PB) BYTE;
735>   DECLARE (PA,PB) BYTE;
736>   /* FIND THE STRING IN SCRATCH STARTING AT PA AND ENDING AT PB */
737>   DECLARE J ADDRESS;

749>   RETURN MATCH;
750> END FIND;

751>
752> SETFIND: PROCEDURE;
753>   /* SETUP THE SEARCH STRING FOR F, N, AND S COMMANDS */
754>   WBE = @; CALL COLLECT; WBP = WBE;
755>   END SETFIND;
756>
757> CHKFOUND: PROCEDURE;
758>   /* CHECK FOR FOUND STRING IN F AND S COMMANDS */
759>   IF NOT FIND(@,WBP) THEN /* NO MATCH */ GO TO OVERCOUNT;
760>   END CHKFOUND;
761>
762>
763>
764> SETRFCB: PROCEDURE;
765>   /* PLACE CHAR INTO READ FILE CONTROL BLOCK AND INCREMENT */
766>   RFCB((RBP := RBP + 1) - 1) = CHAR;
767>   END SETRFCB;
768>
769> TYPELINES: PROCEDURE;
770>   ICL I ADDR;
771>   CALL SETLIMITS;
772>   IF GETMEM(FIRST - 1) = LF AND COLUMN <> 0 THEN
773>     DO; CALL PRINTC(CR); CALL PRINTC(LF);
774>   END;
775>   DO I = FIRST TO LAST;
776>     IF BREAK$KEY THEN
777>       GO TO RESET;
778>     CALL PRINTC(GETMEM(I));
779>   END;
780> END TYPELINES;
781>
782> SETLPP: PROCEDURE;
783>   /* SET DISTANCE TO LINES PER PAGE */
784>   DISTANCE = LPP;
785>   END SETLPP;

```

CP/M VERSION _____
 COPYRIGHT © 1976
 DIGITAL RESEARCH
 P. O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____

```

706>
707> SAVEDIST: PROCEDURE;
708>   TDIST = DISTANCE;
709>   END SAVEDIST;
710>
711> RESTDIST: PROCEDURE;
712>   DISTANCE = TDIST;
713>   END RESTDIST;
714>
715> ^PAGE: PROCEDURE;
716>   DECLARE I BYTE;
717>   CALL SAVEDIST;
718>   CALL SETLPP;
719>   CALL MOVELINES;
720>   I = DIRECTION;
721>   DIRECTION = FORWARD;
722>   CALL SETLPP;
723>   CALL TYPELINES;
724>   DIRECTION = I;
725>   IF LAST = MAXN OR FIRST = 1 THEN CALL ZERODIST;
726>   ELSE CALL RESTDIST;
727>   END PAGE;
728>
729> WAIT: PROCEDURE;
730>   /* 1/2 SECOND TIME OUT */
731>   DECLARE I BYTE;
732>   DO I = 0 TO 19;
733>     IF BREAK$KEY THEN GO TO RESET;
734>     CALL TIME(250);
735>   END;
736>   END WAIT;
737>
738> SETFORWARD: PROCEDURE;
739>   DIRECTION = FORWARD;
740>   DISTANCE = 1;
741>   END SETFORWARD;
742>
743> APPHALF: PROCEDURE;
744>   /* APPEND 'TIL BUFFER IS AT LEAST HALF FULL */
745>   CALL SETFF; /* DISTANCE = 0FFFFH */
746>   DO WHILE DISTNZERO;
747>     IF FRONT >= HMAX THEN CALL ZERODIST; ELSE
748>       CALL READLINE;
749>   END;
750>   END APPHALF;
751>
752> INSCRLF: PROCEDURE;
753>   /* INSERT CR LF CHARACTERS */
754>   CHAR = CR; CALL INSERT;
755>   CHAR = LF; CALL INSERT;
756>   END INSCRLF;
757>
758> TESTCASE: PROCEDURE;
759>   DECLARE T BYTE;
760>   /* TEST FOR UPPER OR LOWER CASE COMMAND AND SET TRANSLATE
761>   FLAG (USED TO DETERMINE IF CHARACTERS WHICH FOLLOW GO TO UPPER */
762>   TRANSLATE = TRUE;
763>   T = LOWERCASE(CHAR);
764>   CHAR = UTRAN(CHAR);
765>   TRANSLATE = NOT T;

```

CP/M VERSION _____
 COPYRIGHT © 1976
 DIGITAL RESEARCH
 P. O. BOX 579
 PACIFIC GROVE, CA 93950

SER. # _____

```

846> END TESTCASE;
847> READCTRAH: PROCEDURE;
848> /* SET TRANSLATE TO FALSE AND READ NEXT CHARACTER */
849> TRANSLATE = FALSE;
850> CHAR = READC;
851> CALL TESTCASE;
852> END READCTRAH;
853>
854>
855> SINGLECOM: PROCEDURE(C) BYTE;
856> /* RETURN TRUE IF COMMAND IS ONLY CHARACTER, NOT IN MACRO */
857> DECLARE C BYTE;
858> RETURN CHAR = C AND COMLEN = 1 AND MP = CP/M VERSION
859> END SINGLECOM;
860>
861>
862>
863>
864>
865>
866>
867> /* INITIALIZE THE SYSTEM */
868> MAX = MAXB - .MEMORY - 1;
869> MEMORY(MAX) = 0; /* STOPS MATCH AT END OF BUFFER */
870> MAXM = MAX - 1;
871> MAXX = SHR(MAXM,1);
872> /* INITIALLY ASSUME NO TRANSLATE TO UPPER CASE */
873> UPPER = FALSE;
874> /* GET SOURCE AND DESTINATION DISKS */
875> IF (FCB(1) = ' ') OR (FCB(17) <> ' ') THEN CALL FERR;
876> IF (SDISK := FCB) = 0 THEN SDISK = CSELECT; ELSE
877> DO; SDISK = SDISK - 1; FCB = 6; /* CLEAR DISK NAME */
878> END;
879> IF (DDISK := FCB(16)) = 0 THEN DDISK = SDISK; ELSE
880> DDISK = DDISK - 1;
881>
882> PESTART:
883> CALL SETUP;
884> MEMORY = LFS;
885> FRONT = 1; BACK = MAXM;
886> COLUMN = 0;
887> GO TO START;
888>
889> OVERCOUNT: FLAG = POUND; GO TO RESET;
890>
891> BADCOM: FLAG = WHAT; GO TO RESET;
892>
893> OVERFLOW: /* ARRIVE HERE ON OVERFLOW CONDITION (I/F,S COMMAND) */
894> FLAG = '>';
895>
896> RESET: /* ARRIVE HERE ON SYSTEM RESET OR OVERFLOW CONDITION */
897> CALL PRINTC(CR); CALL PRINTC(LF);
898> CALL PRINTC(CHAR);
899> CALL PRINTC(FLAG);
900> CALL CRLF;
901>
902> START:
903> READBUFF = TRUE;
904> MP = 0;
905>

```

CP/M VERSION
 COPYRIGHT © 1976
 DIGITAL RESEARCH
 P. O. BOX 579
 PACIFIC GROVE, CA. 93950
 SER. # _____

```

906>
907> DO FOREVER; /* OR UNTIL THE POWER IS TURNED OFF */
908>
909> /* **** */
910> SIMPLE COMMANDS (CANNOT BE PRECEDED BY DIRECTION/DISTANCE):
911> E END-THE EDIT NORMALLY
912> H MOVE TO HEAD OF EDITED FILE
913> I INSERT CHARACTERS
914> O RETURN TO THE ORIGINAL FILE
915> R READ FROM LIBRARY FILE
916> Q QUIT EDIT WITHOUT CHANGES TO ORIGINAL FILE
917> ****
918>
919>
920>
921> INSERTING = FALSE;
922> CALL READCTRAH;
923> MI = CBP; /* SAVE STARTING ADDRESS FOR <CR> AT COMMAND */
924> IF SINGLECOM('E') THEN
925> DO; CALL TERMINATE;
926> IF SDISK <> DDISK THEN DDISK = SDISK;
927> GO TO BOOT;
928> END; ELSE
929>
930>
931> IF SINGLECOM('H') THEN /* GO TO TOP */
932> DO; CALL TERMINATE;
933> CHAR = DDISK; DDISK = SDISK; SDISK = CHAR;
934> /* PING - PONG DISKS */
935> GO TO RESTART;
936> END; ELSE
937>
938>
939> IF CHAR = 'I' THEN /* INSERT CHARACTERS */
940> DO;
941> INSERTING = (CBP = COMLEN) AND (MP = 0);
942> DO WHILE SCANNING;
943> DO WHILE CHAR <> 0;
944> IF CHAR = CTLU THEN /* LINE DELETE */
945> DD; CALL PRINTC(CR); CALL PRINTC(LF);
946> /* ELIMINATE LINE */ DIRECTION = BACKWARD;
947> DISTANCE = 0; CALL SETLIMITS; CALL SETPTRS;
948> END; ELSE
949> IF CHAR = RUBOUT THEN
950> DO; IF FRONT = 1 THEN GO TO RESET;
951> CALL DECFRONT; CALL PRINTC(GETMEM(FRONT));
952> END; ELSE
953> IF CHAR = CTL THEN CALL INSCRLF; ELSE
954> CALL INSERT;
955> IF CHAR = CR THEN
956> CALL PRINTNMAC(CHAR=LF); ELSE CHAR=0;
957> END;
958> IF CHAR <> ENDFILE THEN /* MUST HAVE STOPPED ON CR */
959> CALL INSCRLF;
960> END; ELSE
961>
962>
963>
964> IF SINGLECOM('O') THEN /* FORGET THIS EDIT */
965> GO TO RESTART; ELSE

```

CP/M VERSION	COPYRIGHT © 1976
DIGITAL RESEARCH	P. O. BOX 579
PACIFIC GROVE, CA. 93950	
SER. # _____	

```

966>
967> IF CHAR = 'R' THEN
968>   DO; /* READ FROM 'LIB' FILE */
969>   TRANSLATE = TRUE;
970>   RBP = 1;
971>   DO WHILE SCANNING;
972>     IF RBP > 8 THEN GO TO OVERCOUNT;
973>     CALL SETRFCB;
974>   END;
975>   CHAR = ',';
976>   DO WHILE RBP <= 8;
977>     CALL SETRFCB;
978>   END;
979>   RFCB(12), RFCB(32) = 0; /* FILL REEL, AND NEXT RECORD */
980>   CALL OPEN(RFCB); RBP = 80H;
981>   IF DCHT = 255 THEN
982>     DO; FLAG = '0'; GO TO RESET;
983>   END;
984>   DO WHILE (CHAR := READFILE) <> ENDFILE;
985>   IF UPPER THEN CHAR = UTRAN(CHAR);
986>   CALL INSERT;
987>   END;
988> END, ELSE
989>
990> IF SINGLECOM('Q') THEN
991>   DO; CALL DELETEx(RFCB); GO TO BOOT;
992> END; ELSE
993>
994> /* MAY BE A COMMAND WHICH HAS AN OPTIONAL DIRECTION AND DISTANCE */
995> DO; /* SCAN A SIGNED INTEGER VALUE (IF ANY) */
996> DCL I BYTE;
997> CALL SETFORWARD;
998>
999> IF CHAR = '--' THEN
1000>   DO; CALL READCTRAN; DIRECTION = BACKWARD;
1001> END;
1002>
1003> IF CHAR = POUND THEN
1004>   DO; CALL SETFF; CALL READCTRAN;
1005> END; ELSE
1006>
1007> IF (I := CHAR - '0') <= 9 THEN
1008>   DO; DISTANCE = I; CALL READCTRAN;
1009>   DO WHILE (I := CHAR - '0') <= 9;
1010>     DISTANCE = SHL(DISTANCE, 3) + SHL(DISTANCE, 1) + I;
1011>     CALL READCTRAN;
1012>   END;
1013> END;
1014> IF DISTZERO THEN DIRECTION = BACKWARD;
1015> /* DIRECTION AND DISTANCE ARE NOW SET */
1016>
1017> /* ***** */
1018> MAY BE A COMMAND WHICH HAS DIRECTION AND DISTANCE SPECIFIED.
1019> B   BEGINNING/BOTTOM OF BUFFER
1020> C   MOVE CHARACTER POSITIONS

```

CP/M VERSION _____
 COPYRIGHT © 1976
 DIGITAL RESEARCH
 P. O. BOX 579
 PACIFIC GROVE, CA. 93950
 SER. # _____

```

1026> D   DELETE CHARACTERS
1027> K   KILL LINES
1028> L   MOVE LINE POSITION
1029> P   PAGE UP OR DOWN (LPP LINES AND PRINT)
1030> T   TYPE LINES
1031> U   OPPOSITE L
1032> <CR> MOVE UP OR DOWN LINES AND PRINT LINE
*****  

1033>
1034> IF CHAR = 'B' THEN
1035>   DO; DIRECTION = 1 - DIRECTION;
1036>   FIRST = 1; LAST = MAXH; CALL MOVER;
1037> END; ELSE
1038>
1039>
1040> IF CHAR = 'C' THEN
1041>   DO; CALL SETCLIMITS; CALL MOVER;
1042> END; ELSE
1043>
1044>
1045> IF CHAR = 'D' THEN
1046>   DO; CALL SETCLIMITS;
1047>   CALL SETPTRS; /* SETS BACK/FRONT */
1048> END; ELSE
1049>
1050> IF CHAR = 'K' THEN
1051>   DO; CALL SETLIMITS;
1052>   CALL SETPTRS;
1053> END; ELSE
1054>
1055>
1056> IF CHAR = 'L' THEN CALL MOVELINES; ELSE
1057>
1058> IF CHAR = 'P' THEN /* PAGE MODE PRINT */
1059>   DO; IF DISTZERO THEN
1060>     DIRECTION = FORWARD;
1061>     CALL SETLPP; CALL TYPELINES;
1062>   END; ELSE
1063>   DO WHILE DISTNZERO; CALL PAGE;
1064>     CALL WAIT;
1065>   END;
1066> END; ELSE
1067>
1068>
1069> IF CHAR = 'T' THEN
1070>   CALL TYPELINES; ELSE
1071>
1072>
1073>
1074>
1075>
1076>
1077>
1078>
1079>
1080>
1081>
1082>
1083> IF CHAR = CR THEN /* MAY BE MOVE/TYPE COMMAND */
1084>   DO;
1085>     IF MI = 1 AND MP = 0 THEN /* FIRST COMMAND */
1086>       DO; CALL MOVELINES; CALL SETFORWARD; CALL TYPELINES;
1087>     END;

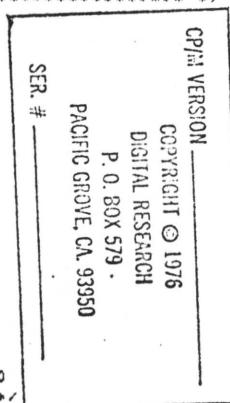
```

CP/M VERSION	COPYRIGHT © 1976 DIGITAL RESEARCH P. O. BOX 579 PACIFIC GROVE, CA. 93950
SER. #	

```

1086>     END; ELSE
1087>     IF DIRECTIONH = FORWARD OR DISTZERO THEN
1088>       DO;
1089>       .
1090>       .
1091> /* ****COMMANDS WHICH ALLOW ONLY A PRECEDING NUMBER**** */
1092>     A APPEND LINES
1093>     F FIND NTH OCCURRENCE
1094>     M APPLY MACRO
1095>     H SAME AS F WITH AUTOSCAN THROUGH FILE
1096>     S PERFORM H SUBSTITUTIONS
1097>     W WRITE LINES TO OUTPUT FILE
1098>
1099> ****
1100>
1101>
1102>
1103>     IF CHAR = 'A' THEN
1104>       DO;
1105>         FIRST = FRONT; LAST = MAXM; CALL MOVER;
1106>         /* ALL STORAGE FORWARD */
1107>         IF DISTZERO THEN CALL APPHALF;
1108>         /* DISTANCE = 0 IF APPHALF CALLED */
1109>         DO WHILE DISTNZERO;
1110>           CALL READLINE;
1111>           END;
1112>           DIRECTION = BACKWARD; CALL MOVER;
1113>           /* POINTERS REPOSITIONED */
1114>         END; ELSE
1115>
1116>
1117>     IF CHAR = 'F' THEN
1118>       DO; CALL SETFIND; /* SEARCH STRING SCANNED
1119> AND SETUP BETWEEN B AND WBP-1 IN SCRATCH */
1120>       DO WHILE DISTNZERO; CALL CHKFOUND;
1121>       END;
1122>     END; ELSE
1123>
1124>
1125>     IF CHAR = 'J' THEN /* JUXTAPOSITION OPERATION */
1126>       DO; DECLARE T ADDRESS;
1127>       CALL SETFIND; CALL COLLECT;
1128>       WBJ = WBE; CALL COLLECT;
1129>       /* SEARCH FOR STRING B - WBP-1, INSERT STRING WBP TO WBE-1,
1130> AND THEN DELETE UP TO STRING WBJ TO WBE-1 */
1131>       DO WHILE DISTNZERO; CALL CHKFOUND;
1132>       /* INSERT STRING */ MI = WBP - 1
1133>       DO WHILE (MI := MI + 1) < WBJ;
1134>         CHAR = SCRATCH(MI); CALL INSERT;
1135>       END;
1136>       T = FRONT; /* SAVE POSITION FOR DELETE */
1137>       IF NOT FIND(WBJ,WBE) THEN GO TO OVERCOUNT;
1138>       /* STRING FOUND, SO MOVE IT BACK */
1139>       FIRST = FRONT - (WBE - WBJ);
1140>       DIRECTION = BACKWARD; CALL MOVER;
1141>       /* NOW REMOVE THE INTERMEDIATE STRING */
1142>       FRONT = T;
1143>     END;
1144>   END; ELSE
1145>
1146>

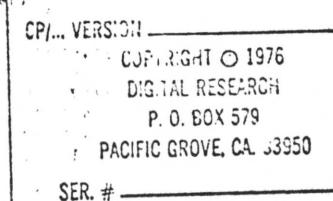
```



```

1147>
1148>     IF CHAR = 'M' AND MP = 0 THEN /* MACRO DEFINITION */
1149>       DO; XP = 255;
1150>       IF DISTANCE = 1 THEN CALL ZERODIST;
1151>       DO WHILE (MACRO(XP := XP + 1) = READC) <> CR;
1152>       END;
1153>       MP = XP; XP = 0; MT = DISTANCE;
1154>     END; ELSE
1155>
1156>
1157>     IF CHAR = 'N' THEN
1158>       DO; /* SEARCH FOR STRING WITH AUTOSCAN */
1159>       CALL SETFIND; /* SEARCH STRING SCANNED */
1160>       DO WHILE DISTNZERO;
1161>         /* FIND ANOTHER OCCURRENCE OF STRING */
1162>         DO WHILE NOT FIND(B,WBP); /* NOT IN BUFFER */
1163>         IF BREAK$KEY THEN GO TO RESET;
1164>         CALL SAVEDIST; CALL CLEARMEM;
1165>         /* MEMORY BUFFER WRITTEN */
1166>         CALL APPHALF;
1167>         DIRECTION = BACKWARD; FIRST = 'I'; CALL MOVER;
1168>         CALL RESTDIST; DIRECTION = FORWARD;
1169>         /* MAY BE END OF FILE */
1170>         IF BACK >= MAXM THEN GO TO OVERCOUNT;
1171>       END;
1172>     END; ELSE
1173>
1174>
1175>     IF CHAR = 'S' THEN /* SUBSTITUTE COMMAND */
1176>       DO; CALL SETFIND;
1177>       CALL COLLECT;
1178>       /* FIND STRING FROM B TO WBP-1, SUBSTITUTE STRING
1179> BETWEEN WBP AND WBE-1 IN SCRATCH */
1180>       DO WHILE DISTNZERO;
1181>         CALL CHKFOUND;
1182>         /* FRONT AND BACK NOW POSITIONED AT FOUND
1183> STRING - REPLACE IT */
1184>         FRONT = FRONT - (MI := WBP); /* BACKED UP */
1185>         DO WHILE MI < WBE;
1186>           CHAR = SCRATCH(MI);
1187>           MI = MI + 1; CALL INSERT;
1188>         END;
1189>       END;
1190>     END; ELSE
1191>
1192>
1193>     IF CHAR = 'W' THEN
1194>       CALL WRITEDOUT; ELSE
1195>
1196>     IF CHAR = 'Z' THEN /* SLEEP */
1197>       DO;
1198>         IF DISTZERO THEN
1199>           DO; IF READCHAR = ENDFILE THEN GO TO RESET;
1200>           END;
1201>           DO WHILE DISTNZERO; CALL WAIT;
1202>           END;
1203>         END; ELSE
1204>       IF CHAR <> '0' THEN /* NOT BREAK LEFT OVER FROM STOP */
1205>       /* DIRECTION FORWARD, BUT NOT ONE OF THE ABOVE */
1206>       GO TO BADCOM;

```



1207>
1236>
1239> END; ELSE /* DIRECTION NOT FORWARD */
1210> GO TO BADCOM;
1211> END;
1212> END;
1213>EOF

CP/M VERSION _____
COPYRIGHT © 1976
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. # _____

317>MEMORY.....1F00H 377>AA.....11C0H 437>PPUTMEM.....0700H 496>SAVEDIST.....0C8E..
 318>COPYRIGHT.....0106H 378>LIFTHEAD.....0267H 438>I.....1ECCH 497>RESTDIST.....0C8E..
 319>MAXA.....14C2H 379>MOVE.....0271H 439>J.....1ECFH 498>PPAGE.....0C8E..
 320>MAX.....14C4H 380>S.....10C2H 440>TRANSFER.....071AH 499>I.....1EF5H
 321>MAXM.....14C6H 381>HD.....10C4H 441>I.....1ED0H 500>WAIT.....0121H
 322>NHMAX.....14C8H 382>N.....10C7H 442>J.....1ED2H 501>I.....1EFF4H
 323>BDISKA.....14CAH 383>FERR.....029EH 443>IDISTANCE.....1ED4H 502>SETFORWARD.....0132H
 324>FCPA.....14CCH 384>GETTYPE.....029CH 444>TDIST.....1ED6H 503>APPHALF.....0132H
 325>BUFA.....14CEH 385>AA.....10C8H 445>INDIRECTION.....1ED8H 504>INSCRLF.....0132H
 326>RFCB.....14D1H 386>SETUP.....02D0H 446>CHAR.....1ED9H 505>TESTCASE.....0137H
 327>PEP.....14F2H 387>FILLSOURCE.....03A0H 447>FRONT.....1EDAH 506>T.....0137H
 328>SBUFF.....14F3H 388>I.....10CBH 448>BACK.....1EDCH 507>READCTRA.....0138H
 329>NDFCB.....18F3H 389>ZH.....03A3H 449>FIRST.....1EE0H 508>SINGLECOM.....01A8H
 330>HDEBUFF.....1914H 390>GETSOURCE.....0413H 450>LAST.....0736H 509>C.....1EFFCH
 331>NSOURCE.....1D14H 391>B.....10CCH 451>SETFF.....0736H 510>I.....1EFFCH
 332>HDEST.....1D16H 392>WRITEDEST.....0446H 452>DDISTZERO.....073FH 511>T.....1EFFCH
 333>SDISK.....1D19H 393>I.....10CDH 453>ZERODIST.....0750H
 334>DDISK.....1D1AH 394>N.....10CEH 454>NDISTZERO.....0759H
 335>MON1.....012AH 395>ZH.....0443H 455>SETLIMITS.....076DH
 336>F.....1D1BH 396>PUTDEST.....0480H 456>I.....1EE2H
 337>AA.....1D1CH 397>B.....10CFH 457>K.....1EE4H
 338>MON2.....013CH 398>FINIS.....04DAH 458>L.....1EE6H
 339>F.....1D1FH 399>MOVEUP.....04DDH 459>M.....1EE8H
 340>AA.....1D28H 400>MACRO.....10D0H 460>MIDDLE.....1EE8H
 341>READCHAR.....0142H 401>SCRATCH.....1E50H 461>LOOPING.....1EECH
 342>FPRTCHAR.....014CH 402>WSP.....1EB4H 462>SETPTRS.....0875H
 343>CHAR.....1D23H 403>WEE.....1FB5H 463>INCFRONT.....0895H
 344>CPFL.....0159H 404>WEJ.....1EB6H 464>INCBACK.....089DH
 345>PPRINT.....0164H 405>FLAG.....1EB7H 465>NDECFRONT.....08A5H
 346>AA.....1D24H 406>MP.....1EB8H 466>NDECBACK.....08ADH
 347>READ.....0179H 407>MI.....1EB9H 467>MOVER.....08B5H
 348>PA.....1D26H 408>XP.....1EBAH 468>MOVELINES.....0900H
 349>NCNT.....1D28H 409>COLUMN.....1EBBH 469>SETLIMITS.....0912H
 350>OPEH.....0185H 410>MT.....1EBCH 470>READLINE.....0993H
 351>FCB.....1D2AH 411>START.....0EB3H 471>B.....1EE0H
 352>CLOSE.....0190H 412>RESTART.....0E60H 472>CTRA.....0996H
 353>FCB.....1D2CH 413>OVERCOUNT.....0EB3H 473>B.....1EEEH
 354>SEARCH.....0191H 414>OVERFLOW.....0E93H 474>WRITELINE.....09EEH
 355>FCB.....1D2EH 415>RESET.....0E98H 475>B.....1EEFH
 356>HDELETE.....01C5H 416>BADCOM.....0E00H 476>URHALF.....0A22H
 357>FCB.....1D30H 417>INSERTING.....1EBFH 477>WRITEOUT.....0A55H
 358>NIISKREAD.....01D5H 418>READBUFF.....1EC0H 478>CLEARMEM.....0A9AH
 359>FCB.....1D32H 419>TTYCHAR.....0569H 479>TERMINATE.....0AA1H
 360>HISKWRITE.....01E5H 420>CHAR.....1E1CH 480>INSERT.....0A8BH
 361>FCB.....1D34H 421>PPRINTC.....0509H 481>SCANNING.....0AE0H
 362>PAKE.....01F5H 422>CHAR.....1EC2H 482>COLLECT.....0A9EH
 363>FCB.....1D36H 423>I.....1EC3H 483>SETSSCR.....0B01H
 364>FENAME.....0209H 424>J.....1EC4H 484>FIND.....0B5DH
 365>FCB.....1D38H 425>*PRINTRNMAC.....05C7H 485>PPA.....1EF0H
 366>MAXLEN.....1D3BH 426>CHAR.....1EC5H 486>PPB.....1EF1H
 367>CMLEN.....1D3CH 427>TRANSLATE.....1EC6H 487>J.....1EF2H
 368>COMBUFF.....1D3DH 428>UPPER.....1EC7H 488>K.....1EF4H
 369>TCBP.....1D5DH 429>LOWERCASE.....05D0H 489>MATCH.....1EF5H
 370>CBP.....1D8EH 430>C.....1EC8H 490>SETFIN.....0BE0H
 371>READCOM.....0219H 431>UTRAN.....05E9H 491>CHKFOUND.....0BF0H
 372>NPBK.....0225H 432>C.....1EC9H 492>SETRFCB.....0C0EH
 373>CSELECT.....0240H 433>READC.....0602H 493>TYPELINES.....0C21H
 374>SELECT.....024AH 434>READFILE.....060EH 494>I.....1EFGH
 375>RDISK.....1DBFH 435>GETMEM.....06F0H 495>SETLPP.....0C92H
 376>SETDMA.....0257H 436>I.....1ECAH