File created: 19-Feb-2021 11:40:35 **{DSK}<Users>kaplan>Local>medley3.5>git-medley>lispusers>EMACS.;9**

changes to: (VARS EMACSCOMS)

previous date: 19-Feb-2021 11:24:35 {DSK}<Users>kaplan>Local>medley3.5>git-medley>lispusers>EMACS.;6

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

;;


```
(RPAQQ EMACSCOMS
       [(* EMACS -- By Kelly Roach *)
        ;; Patched by Ron Kaplan (2021) to require TEDIT and to eliminate a dependency on a historical attempt at BQUOTE.
        ;; So it loads, but it really doesn't work.
        ;; This has to be compiled with EXPORTS.ALL
        (DECLARE%: (FILES TEDIT))
        (COMS (* EMACS *)
              (INITVARS (BytesPerPage 512)
                     (EMACS.COMMANDS NIL)
                     (EMACS.MCOMMANDS NIL)
                     (EMACS.XCOMMANDS NIL)
                     (EMACS.LIST %' ((1 EMACS.GOTO.BOL)
                                     (2 EMACS.BACK.BYTE)
                                     (4 EMACS.FWD.DELETE.BYTE)
                                     (5 EMACS.GOTO.EOL)
                                     (6 EMACS.FWD.BYTE)
                                     (9 EMACS.TAB)
                                     (11 EMACS.KILL.LINE)
                                     (12 EMACS.REDISPLAY)
                                     (14 EMACS.NEXT.LINE)
                                     (16 EMACS.PREVIOUS.LINE)
                                     (17 EMACS.QUOTE.BYTE)
                                     (19 EMACS.SEARCH)
                                     (20 EMACS.TRANSPOSE.BYTES)
                                     (22 EMACS.NEXT.SCREENFULL)
                                     (24 EMACS.CX)
                                     (26 EMACS.CZ)
                                     (41 EMACS.RPAREN)
                                     (93 EMACS.RBRACKET)
                                     (125 EMACS.RBRACE)
                                     (127 EMACS.BACK.DELETE.BYTE)))
                     (EMACS.MLIST %' ((1 EMACS.GOTO.BOD)
                                      (2 EMACS.SAFE.BACK.SEXPR)
                                      (5 EMACS.GOTO.EOD)
                                      (6 EMACS.FWD.SEXPR)
                                      (11 EMACS.KILL.SEXPR)
                                      (60 EMACS.GOTO.BOF)
                                      (62 EMACS.GOTO.EOF)
                                      (66 EMACS.BACK.WORD)
                                      (68 EMACS.FWD.DELETE.WORD)
                                      (69 EMACS.EDIT)
                                      (70 EMACS.FWD.WORD)
                                      (71 EMACS.GRIND)
                                      (52 EMACS.SNARF)
                                      (86 EMACS.PREVIOUS.SCREENFULL)
                                      (94 EMACS.JOIN.LINES)
                                      (127 EMACS.BACK.DELETE.WORD)))
                     (EMACS.XLIST %' ((22 EMACS.CXCV)
                                      (23 EMACS.CXCW)
                                      (26 EMACS.CXCZ)))
                     (\BQUOTE.LEVEL 0))
              (RECORDS EMACSSTREAM)
              (FNS EMACS.INIT EMACS.INIT.BACKGROUND DEDITEmacs EMACS.INIT.COMMANDS EMACS.COMMAND EMACS.OPERATE
                   EMACS.GETKEY EMACS EMACS.PROCESS EMACS.TEDIT1 EMACS.WINDOW EMACS.SETFILEPTR EMACS.GETCARETPTR
                   EMACS.SETCARETPTR EMACS.SHOWCARET EMACS.BOL EMACS.EOL EMACS.DELETE.BYTES EMACS.BOFP
                   EMACS.EOFP EMACS.CCHAR EMACS.PEEKBIN EMACS.FBYTE EMACS.FWORD EMACS.BYTEP EMACS.FSKIP
                   EMACS.FSKIPTO EMACS.BBYTE EMACS.BCHAR EMACS.BPEEKCHAR EMACS.BWORD EMACS.BSKIP EMACS.BSKIPTO
                   EMACS.SET.EOF EMACS.GOTO.BOL EMACS.BACK.BYTE EMACS.FWD.DELETE.BYTE EMACS.GOTO.EOL
                   EMACS.FWD.BYTE EMACS.KILL.LINE EMACS.DELETE.CHARS EMACS.REDISPLAY EMACS.NEXT.LINE
                   EMACS.PREVIOUS.LINE EMACS.QUOTE.BYTE EMACS.SEARCH EMACS.TRANSPOSE.BYTES EMACS.NEXT.SCREENFULL
                   EMACS.CXCV EMACS.CXCW EMACS.CXCZ EMACS.FWD.SEXPR EMACS.BACK.DELETE.BYTE EMACS.GOTO.BOD
                   EMACS.BOD EMACS.GOTO.EOD EMACS.EOD EMACS.KILL.SEXPR EMACS.GOTO.BOF EMACS.GOTO.EOF
                   EMACS.BACK.WORD EMACS.FWD.DELETE.WORD EMACS.EDIT EMACS.FWD.WORD EMACS.GRIND EMACS.SNARF
                   EMACS.MT EMACS.PREVIOUS.SCREENFULL EMACS.JOIN.LINES EMACS.BACK.DELETE.WORD
                   NEW.TEDIT.SELECT.LINE.SCANNER))
        (COMS (* BALANCE *)
              (PROPS (ACCESSFNS EMACS.TAB)
                     (DATATYPE EMACS.TAB)
                     (DEFEXPR EMACS.TAB)
```

```
                    (DEFFEXPR EMACS.TAB)
                    (DEFVAR EMACS.TAB)
                    (DO EMACS.TAB)
                    (FOR EMACS.TAB)
                    [LAMBDA EMACS.TAB]
                    (PROG EMACS.TAB)
                    (RECORD EMACS.TAB)
                    (SELECT EMACS.TAB)
                    (SELECTQ EMACS.TAB)
                    (UNTIL EMACS.TAB)
                    (WHILE EMACS.TAB))
             (INITVARS (EMACS.DELIMS NIL)
                    (EMACS.SDELIMS NIL)
                    (EMACS.LDELIMS NIL)
                    (EMACS.RDELIMS NIL)
                    (EMACS.SCACHE NIL)
                    (EMACS.BCACHE NIL)
                    (EMACS.SYNTAX NIL)
                    (EMACS.CR 1)
                    (EMACS.WS 2)
                    (EMACS.SD 4)
                    (EMACS.NONCR 8)
                    (EMACS.NONWS 16)
                    (EMACS.NONSD 32)
                    (EMACS.BQ 64)
                    (EMACS.ALPHA 128)
                    (EMACS.BD 256)
                    (EMACS.SPACE 512))
             (FNS EMACS.DELIMS EMACS.CR EMACS.RPAREN EMACS.RBRACKET EMACS.RBRACE EMACS.RANGLE
                  EMACS.SDELIM.COMMAND EMACS.LDELIM.COMMAND EMACS.RDELIM.COMMAND EMACS.SDELIM EMACS.LDELIM
                  EMACS.RDELIM EMACS.OPEN.STRING EMACS.CLOSE.STRING EMACS.OPEN.BALANCE EMACS.CLOSE.BALANCE
                  EMACS.FLUSH.CACHE EMACS.SCACHE EMACS.BCACHE EMACS.SAFE.BACK.SEXPRS EMACS.SAFE.BACK.SEXPR
                  EMACS.BACK.SEXPR EMACS.BACK.SKIPSEPRS EMACS.BACK.ESCAPEDP EMACS.TAB EMACS.TAB.INDENT
                  EMACS.INIT.SYNTAX))
        (DECLARE%: DONTEVAL@LOAD DOCOPY (P (EMACS.INIT)
                                           (MOVD? %'TEDIT.SELECT.LINE.SCANNER %' OLD.TEDIT.SELECT.LINE.SCANNER)
                                           (MOVD %'NEW.TEDIT.SELECT.LINE.SCANNER %' TEDIT.SELECT.LINE.SCANNER)
                                           (MOVD %'EMACS %'TEDIT])
```

(* * EMACS -- By Kelly Roach *)

;; Patched by Ron Kaplan (2021) to require TEDIT and to eliminate a dependency on a historical attempt at BQUOTE.

;; So it loads, but it really doesn't work.

;; This has to be compiled with EXPORTS.ALL

```
(DECLARE%:

(FILESLOAD TEDIT)
)
```

(* * EMACS *)

```
(RPAQ? BytesPerPage 512)

(RPAQ? EMACS.COMMANDS NIL)

(RPAQ? EMACS.MCOMMANDS NIL)

(RPAQ? EMACS.XCOMMANDS NIL)

(RPAQ? EMACS.LIST %' ((1 EMACS.GOTO.BOL)
                      (2 EMACS.BACK.BYTE)
                      (4 EMACS.FWD.DELETE.BYTE)
                      (5 EMACS.GOTO.EOL)
                      (6 EMACS.FWD.BYTE)
                      (9 EMACS.TAB)
                      (11 EMACS.KILL.LINE)
                      (12 EMACS.REDISPLAY)
                      (14 EMACS.NEXT.LINE)
                      (16 EMACS.PREVIOUS.LINE)
                      (17 EMACS.QUOTE.BYTE)
                      (19 EMACS.SEARCH)
                      (20 EMACS.TRANSPOSE.BYTES)
                      (22 EMACS.NEXT.SCREENFULL)
                      (24 EMACS.CX)
                      (26 EMACS.CZ)
                      (41 EMACS.RPAREN)
                      (93 EMACS.RBRACKET)
                      (125 EMACS.RBRACE)
                      (127 EMACS.BACK.DELETE.BYTE)))

(RPAQ? EMACS.MLIST %' ((1 EMACS.GOTO.BOD)
                       (2 EMACS.SAFE.BACK.SEXPR)
                       (5 EMACS.GOTO.EOD)
                       (6 EMACS.FWD.SEXPR)
```

```
                        (11 EMACS.KILL.SEXPR)
                        (60 EMACS.GOTO.BOF)
                        (62 EMACS.GOTO.EOF)
                        (66 EMACS.BACK.WORD)
                        (68 EMACS.FWD.DELETE.WORD)
                        (69 EMACS.EDIT)
                        (70 EMACS.FWD.WORD)
                        (71 EMACS.GRIND)
                        (52 EMACS.SNARF)
                        (86 EMACS.PREVIOUS.SCREENFULL)
                        (94 EMACS.JOIN.LINES)
                        (127 EMACS.BACK.DELETE.WORD)))

(RPAQ? EMACS.XLIST %' ((22 EMACS.CXCV)
                       (23 EMACS.CXCW)
                       (26 EMACS.CXCZ)))

(RPAQ? \BQUOTE.LEVEL 0)

(DECLARE%: EVAL@COMPILE

[ACCESSFNS EMACSSTREAM ((TEXTOBJ (fetch (STREAM F3) of DATUM))
                        (WINDOW (fetch (TEXTOBJ SELWINDOW) of (fetch (EMACSSTREAM TEXTOBJ) of DATUM)))
                        (SELECTION (fetch (TEXTOBJ SEL) of (fetch (EMACSSTREAM TEXTOBJ) of DATUM)))
                        (CARETPTR (EMACS.GETCARETPTR DATUM))
                        (FILEPTR (GETFILEPTR DATUM))
                        (DIRTY (fetch (TEXTOBJ \DIRTY) of (fetch (EMACSSTREAM TEXTOBJ) of DATUM)))
                        (BCACHE (EMACS.BCACHE DATUM))
                        (SCACHE (EMACS.SCACHE DATUM]
)

(DEFINEQ
```

(EMACS.INIT

```
  [LAMBDA NIL                                                    (* kbr%: "12-Jul-86 16:54")
                                                                 (* Initializes EMACS. *)
    (PROG NIL
          [SETQ TEDIT.INTERRUPTS '((7 HELP]
          (SETQ EMACS.READTABLE (COPYREADTABLE FILERDTBL))
          (EMACS.INIT.COMMANDS)
          (EMACS.INIT.SYNTAX)
          (EMACS.INIT.BACKGROUND)
          (ADDTOVAR *DEDIT-MENU-COMMANDS* (Emacs DEDITEmacs))
          (CHANGECCODE 'NILL 'TTYDISPLAYSTREAM '\TEDIT.COMMAND.LOOP])
```

(EMACS.INIT.BACKGROUND

```
  [LAMBDA NIL                                                    (* kbr%: "24-Jul-85 16:36")
                                                                 (* Fix up BackgroundMenu. *)
    (PROG NIL
          (SETQ BackgroundMenuCommands (FOR BUCKET IN BackgroundMenuCommands
                                          WHEN (NOT (EQ (CAR BUCKET)
                                                        'TEdit))
                                          COLLECT BUCKET))
          (ADDTOVAR BackgroundMenuCommands (EMACS '(EMACS)
                                                  "Opens an Edit Window."))
          (SETQ BackgroundMenu NIL)                              (* BackgroundMenu recreated by WINDOW package next time
                                                                 user buttons background. *)
      ])
```

(DEDITEmacs

```
  [LAMBDA NIL                                                    (* kbr%: "24-Jul-85 16:36")
                                                                 (* Fn to let DEDIT call EMACS on DEDIT top selection.
                                                                 *)
    (PROG (EXPR)
          (CURSOR T)
          (SETQ EXPR (CAR (TOPSELECTION)))
          (SETQ EXPR (READ (EMACS (MKSTRING EXPR)
                                  NIL T)
                           EMACS.READTABLE))
          (DEDITZAPCAR (TOPSELECTION)
                 EXPR])
```

(EMACS.INIT.COMMANDS

```
  [LAMBDA NIL                                                    (* kbr%: "24-Jul-85 16:36")
                                                                 (* Initialize TEDIT.READTABLE. *)
    (PROG NIL
          (SETQ EMACS.COMMANDS (ARRAY 128 'POINTER NIL 0))
          (SETQ EMACS.MCOMMANDS (ARRAY 128 'POINTER NIL 0))
          (SETQ EMACS.XCOMMANDS (ARRAY 128 'POINTER NIL 0))
          (FOR BUCKET IN EMACS.LIST DO (SETA EMACS.COMMANDS (CAR BUCKET)
                                             (CADR BUCKET)))
          (FOR BUCKET IN EMACS.MLIST DO (SETA EMACS.MCOMMANDS (CAR BUCKET)
                                              (CADR BUCKET)))
          (FOR BUCKET IN EMACS.XLIST DO (SETA EMACS.XCOMMANDS (CAR BUCKET)
```

```
                                        (CADR BUCKET)))
        (FOR I FROM 0 TO 255 DO (TEDIT.SETFUNCTION I (EMACS.COMMAND I)
                                      TEDIT.READTABLE])
```

(EMACS.COMMAND

```
  [LAMBDA (I)                                                    (* kbr%: "24-Jul-85 16:36")
    `(LAMBDA (STREAM)
       (EMACS.OPERATE ,I STREAM])
```

(EMACS.OPERATE

```
  [LAMBDA (STREAM)                                               (* kbr%: "27-Jul-86 17:26")
                                                                 (* Accept token from user *)
    (PROG (TEXTOBJ I N FN PTR CH)
          (TTYDISPLAYSTREAM (fetch (EMACSSTREAM WINDOW) of STREAM))
          (SETQ TEXTOBJ (fetch (EMACSSTREAM TEXTOBJ) of STREAM))
          [while (\SYSBUFP) do                                   (* Handle user type-in)
                            (SETQ I (\GETKEY))
                            (SETFILEPTR STREAM (fetch (EMACSSTREAM CARETPTR) of STREAM))
                            (SETQ N 1)
                            (while (EQ I (CHARCODE ^U)) do (SETQ N (ITIMES 4 N))
                                                           (SETQ I (\GETKEY)))
                            [SELCHARQ I
                                 ((ESC ^Z)
                                      (SETQ FN (ELT EMACS.MCOMMANDS (\GETKEY))))
                                 (^X (SETQ FN (ELT EMACS.XCOMMANDS (\GETKEY))))
                                 (COND
                                    ((ILESSP I 128)
                                     (SETQ FN (ELT EMACS.COMMANDS I)))
                                    ((ILESSP I 256)
                                     (SETQ FN (ELT EMACS.MCOMMANDS (IDIFFERENCE I 128]
                            (COND
                               ((NULL FN)                        (* Insert char I N times. *)
                                                                 (* Handle blue pending delete, if there is one.)
                                (TEDIT.DO.BLUEPENDINGDELETE SEL TEXTOBJ)
                                (SETQ PTR (GETFILEPTR STREAM))
                                (COND
                                   ([AND (NOT (ZEROP PTR))
                                         (EQ (\BACKPEEKBIN STREAM)
                                             (CHARCODE CR))
                                         (NOT (MEMB I (CHARCODE (SP TAB]
                                                                 (* Start of a def *)
                                    (EMACS.FLUSH.CACHE)))
                                [COND
                                   ((IEQP N 1)
                                    (TEDIT.\INSERT I SEL TEXTOBJ))
                                   (T (SETQ CH (MKSTRING (CHARACTER I)))
                                      (TEDIT.INSERT STREAM (ALLOCSTRING N CH]
                                (SETFILEPTR STREAM (IPLUS PTR N)))
                               (T (for J from 1 to N do (APPLY* FN STREAM))
                                  (COND
                                     ([AND (ILESSP I 256)
                                           (NOT (BITTEST (ELT EMACS.SYNTAX I)
                                                         (LOGOR EMACS.CR EMACS.SD EMACS.BD]
                                      (EMACS.FLUSH.CACHE]
          (EMACS.SHOWCARET STREAM])
```

(EMACS.GETKEY

```
  [LAMBDA NIL                                                    (* kbr%: "24-Jul-85 16:36")
    (PROG (CODE)
          (CARET 'OFF)
          (SETQ CODE (\GETKEY))
          (CARET T)
          (RETURN CODE])
```

(EMACS

```
  [LAMBDA (TEXT WINDOW DONTSPAWN PROPS)                          (* kbr%: "24-Jul-85 16:36")
    (PROG (PROCESS)                                              (* Get TEXT. *)
          [COND
             ((AND (NOT (NULL TEXT))
                   (LITATOM TEXT))
              (SETQ TEXT (OPENFILE TEXT 'INPUT 'OLD]             (* Get WINDOW. *)
          [COND
             ((NULL WINDOW)
              (SETQ WINDOW (EMACS.WINDOW DONTSPAWN PROPS]
          (COND
             (DONTSPAWN                                          (* Don't spawn a process. *)
                    (RETURN (EMACS.TEDIT1 TEXT WINDOW T PROPS)))
             (T                                                  (* Spawn a process. *)
                (SETQ PROCESS (ADD.PROCESS `(EMACS.PROCESS ',TEXT ',WINDOW ',PROPS)
                                           'EMACS
                                           'NO))
                (TTY.PROCESS PROCESS)
                (RETURN PROCESS])
```

(EMACS.PROCESS

```
[LAMBDA (TEXT WINDOW PROPS)                                    (* kbr%: "24-Jul-85 16:36")
  (PROG NIL
        (WINDOWPROP WINDOW 'PROCESS (THIS.PROCESS))
        (RETURN (EMACS.TEDIT1 TEXT WINDOW NIL PROPS])
```

(EMACS.TEDIT1

```
[LAMBDA (TEXT WINDOW UNSPAWNED PROPS)                          (* kbr%: "24-Jul-85 16:36")
  (PROG (ANSWER)
        (RESETLST
            (RESETSAVE (TTYDISPLAYSTREAM WINDOW))
            (RESETSAVE NIL (LIST 'INPUT (INFILE T)))
            (RESETSAVE NIL (LIST 'OUTPUT (OUTFILE T)))
            (SETQ ANSWER (\TEDIT1 TEXT WINDOW UNSPAWNED PROPS)))
        (RETURN ANSWER])
```

(EMACS.WINDOW

```
[LAMBDA (DONTSPAWN PROPS)                                      (* kbr%: "24-Jul-85 16:36")
  (PROG (WINDOW)
        [COND
           ((AND DONTSPAWN TEDIT.DEFAULT.WINDOW)
            (SETQ WINDOW TEDIT.DEFAULT.WINDOW))
           (T (SETQ WINDOW (TEDIT.CREATEW "Indicate region for EMACS"]
        (WINDOWPROP WINDOW 'TEDIT.PROPS PROPS)
        (RETURN WINDOW])
```

(EMACS.SETFILEPTR

```
[LAMBDA (STREAM PTR)                                           (* kbr%: "24-Jul-85 16:36")
                                                               (* Patch around bug in TEDIT SETFILEPTR.
                                                               *)
  (PROG NIL
        (COND
           ((IGREATERP (GETEOFPTR STREAM)
                   0)
            (SETFILEPTR STREAM PTR)
            (SETFILEPTR STREAM PTR])
```

(EMACS.GETCARETPTR

```
[LAMBDA (STREAM)                                               (* kbr%: "24-Jul-85 16:36")
  (PROG (SELECTION ANSWER)
        (SETQ SELECTION (fetch (EMACSSTREAM SELECTION) of STREAM))
        (SETQ ANSWER (SELECTQ (fetch (SELECTION POINT) of SELECTION)
                         (LEFT (SUB1 (fetch (SELECTION CH#) of SELECTION)))
                         (RIGHT (fetch (SELECTION CHLIM) of SELECTION))
                         (SHOULDNT)))
        (RETURN ANSWER])
```

(EMACS.SETCARETPTR

```
[LAMBDA (STREAM PTR)                                           (* kbr%: "24-Jul-85 16:36")
                                                               (* Move caret to new filepos. *)
  (PROG (EOF)
        (SETQ EOF (GETEOFPTR STREAM))
        (SETQ PTR (IMIN (IMAX PTR 0)
                        EOF))
        (TEDIT.SETSEL STREAM (ADD1 PTR)
               0
               'LEFT)
        (EMACS.SETFILEPTR STREAM PTR])
```

(EMACS.SHOWCARET

```
[LAMBDA (STREAM)                                               (* kbr%: "24-Jul-85 16:36")
  (PROG (PTR)
        (SETQ PTR (GETFILEPTR STREAM))
        (EMACS.SETCARETPTR STREAM PTR)
        (TEDIT.NORMALIZECARET (fetch (EMACSSTREAM TEXTOBJ) of STREAM))
        (EMACS.SETFILEPTR STREAM PTR])
```

(EMACS.BOL

```
[LAMBDA (STREAM PTR)                                           (* kbr%: "24-Jul-85 16:36")
                                                               (* Beginning of line wrt filepos PTR.
                                                               *)
  (PROG (OLDPTR BOL)
        (SETQ OLDPTR (GETFILEPTR STREAM))
        (EMACS.SETFILEPTR STREAM PTR)
        (EMACS.BSKIP STREAM EMACS.NONCR)
        (SETQ BOL (GETFILEPTR STREAM))
        (EMACS.SETFILEPTR STREAM OLDPTR)
        (RETURN BOL])
```

(EMACS.EOL

```
[LAMBDA (STREAM PTR)                                    (* kbr%: "24-Jul-85 16:36")
                                                        (* End of line wrt filepos PTR. *)

   (PROG (OLDPTR EOL)
         (SETQ OLDPTR (GETFILEPTR STREAM))
         (EMACS.SETFILEPTR STREAM PTR)
         (EMACS.FSKIP STREAM EMACS.NONCR)
         (SETQ EOL (GETFILEPTR STREAM))
         (EMACS.SETFILEPTR STREAM OLDPTR)
         (RETURN EOL])
```

(EMACS.DELETE.BYTES

```
[LAMBDA (STREAM PTR1 PTR2)                              (* kbr%: "19-Feb-85 15:11")
                                                        (* Delete between PTR1 & PTR2 inclusive.
                                                        *)

   (PROG (PTR LENGTH)
         (SETQ PTR (GETFILEPTR STREAM))
         (SETQ PTR1 (IMAX 0 PTR1))
         (SETQ PTR2 (IMIN (GETEOFPTR STREAM)
                          PTR2))
         (SETQ LENGTH (IPLUS PTR2 (IMINUS PTR1)
                             1))
         (TEDIT.DELETE STREAM (ADD1 PTR1)
               LENGTH)
         (COND
            ((ILEQ PTR PTR1)
             (EMACS.SETFILEPTR STREAM PTR))
            ((ILEQ PTR PTR2)
             (EMACS.SETFILEPTR STREAM PTR1))
            (T (EMACS.SETFILEPTR STREAM (IDIFFERENCE PTR LENGTH])
```

(EMACS.BOFP

```
[LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
   (ZEROP (GETFILEPTR STREAM])
```

(EMACS.EOFP

```
[LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
   (IEQP (GETFILEPTR STREAM)
         (GETEOFPTR STREAM])
```

(EMACS.CCHAR

```
[LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
                                                        (* Caret char. Char being pointed at by caret.
                                                        *)

   (PROG (ANSWER)
         (SETQ ANSWER (\BIN STREAM))
         (\BACKBIN STREAM)
         (RETURN ANSWER])
```

(EMACS.PEEKBIN

```
[LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
   (PROG (PTR ANSWER)
         (SETQ PTR (GETFILEPTR STREAM))
         (SETQ ANSWER (\BIN STREAM))
         (EMACS.SETFILEPTR STREAM PTR)
         (RETURN ANSWER])
```

(EMACS.FBYTE

```
[LAMBDA (STREAM)                                        (* kbr%: "19-Feb-85 15:11")
                                                        (* Forward a char. *)

   (COND
      ((NOT (EMACS.EOFP STREAM))
       (\BIN STREAM])
```

(EMACS.FWORD

```
[LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
                                                        (* Forward a word. *)

   (PROG NIL
         (EMACS.FSKIP STREAM EMACS.WS)
         (EMACS.FSKIP STREAM EMACS.NONWS])
```

(EMACS.BYTEP

```
[LAMBDA (N)                                             (* kbr%: "24-Jul-85 16:38")
   (AND (SMALLP N)
        (ILESSP N 256)
        N])
```

(EMACS.FSKIP

```
[LAMBDA (STREAM CLASS LIMIT)                                (* kbr%: "24-Jul-85 16:36")
                                                            (* Skip chars in CLASS. *)

   [COND
      ((NULL LIMIT)
       (SETQ LIMIT (GETEOFPTR STREAM]
   (PROG NIL
         (while (AND (ILESSP (GETFILEPTR STREAM)
                             LIMIT)
                     (BITTEST (ELT EMACS.SYNTAX (OR (EMACS.BYTEP (EMACS.PEEKBIN STREAM))
                                                    256))
                              CLASS))
            do (\BIN STREAM])
```

(EMACS.FSKIPTO

```
[LAMBDA (STREAM CLASS)                                      (* kbr%: "24-Jul-85 16:36")
                                                            (* Skip chars in CLASS. *)
   (PROG NIL
         (WHILE (AND (NOT (EMACS.EOFP STREAM))
                     (NOT (BITTEST (ELT EMACS.SYNTAX (OR (EMACS.BYTEP (\BIN STREAM))
                                                         256))
                                   CLASS)))
            DO                                              (* Continue reading. *)])
```

(EMACS.BBYTE

```
[LAMBDA (STREAM)                                            (* kbr%: "19-Feb-85 15:12")
                                                            (* Backward a byte. *)
   (COND
      ((NOT (EMACS.BOFP STREAM))
       (\BACKBIN STREAM])
```

(EMACS.BCHAR

```
[LAMBDA (STREAM)                                            (* kbr%: "24-Jul-85 16:36")
                                                            (* Backward a char. *)
   (PROG NIL
         (COND
            ((NOT (EMACS.BOFP STREAM))
             (\BACKBIN STREAM])
```

(EMACS.BPEEKCHAR

```
[LAMBDA (STREAM)                                            (* kbr%: "19-Feb-85 15:12")
                                                            (* Backwards peek at char. *)
   (PROG (PTR BYTE)
         (SETQ PTR (GETFILEPTR STREAM))
         (SETQ BYTE (EMACS.BCHAR STREAM))
         (SETFILEPTR STREAM PTR)
         (RETURN BYTE])
```

(EMACS.BWORD

```
[LAMBDA (STREAM)                                            (* kbr%: "24-Jul-85 16:36")
                                                            (* Backward a word. *)
   (PROG NIL
         (EMACS.BSKIP STREAM EMACS.WS)
         (EMACS.BSKIP STREAM EMACS.NONWS])
```

(EMACS.BSKIP

```
[LAMBDA (STREAM CLASS LIMIT)                                (* kbr%: "24-Jul-85 16:36")
                                                            (* Skip chars in CLASS. *)
   (COND
      ((NULL LIMIT)
       (SETQ LIMIT 0)))
   (PROG NIL
         (while (AND (IGREATERP (GETFILEPTR STREAM)
                                LIMIT)
                     (BITTEST (ELT EMACS.SYNTAX (OR (EMACS.BYTEP (\BACKPEEKBIN STREAM))
                                                    256))
                              CLASS))
            do (\BACKBIN STREAM])
```

(EMACS.BSKIPTO

```
[LAMBDA (STREAM CLASS)                                      (* kbr%: "24-Jul-85 16:36")
                                                            (* Skip chars in CLASS. *)
   (PROG NIL
         (WHILE (AND (NOT (EMACS.BOFP STREAM))
                     (NOT (BITTEST (ELT EMACS.SYNTAX (OR (EMACS.BYTEP (\BACKBIN STREAM))
                                                         256))
                                   CLASS)))
            DO                                              (* Continue reading. *)])
```

(EMACS.SET.EOF

```
[LAMBDA (STREAM PTR)                                    (* kbr%: "19-Feb-85 15:12")
                                                        (* Temporarily reset eof of STREAM.
                                                        *)
    (PROG NIL
          (replace (STREAM EPAGE) of STREAM with (LRSH PTR 8))
          (replace (STREAM EOFFSET) of STREAM with (LOGAND PTR 255))
          (replace (TEXTOBJ TEXTLEN) of (fetch (EMACSSTREAM TEXTOBJ) of STREAM) with PTR])
```

(EMACS.GOTO.BOL

```
[LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
                                                        (* Go to beginning of line. *)
    (PROG NIL
          (EMACS.BSKIP STREAM EMACS.NONCR])
```

(EMACS.BACK.BYTE

```
[LAMBDA (STREAM)                                        (* kbr%: "19-Feb-85 15:12")
                                                        (* Go back a byte. *)
    (PROG NIL
          (EMACS.BBYTE STREAM])
```

(EMACS.FWD.DELETE.BYTE

```
[LAMBDA (STREAM)                                        (* kbr%: "19-Feb-85 15:12")
                                                        (* Delete byte. *)
    (PROG (PTR)
          (SETQ PTR (GETFILEPTR STREAM))
          (EMACS.DELETE.BYTES STREAM PTR PTR])
```

(EMACS.GOTO.EOL

```
[LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
                                                        (* Go to end of line. *)
    (PROG NIL
          (EMACS.FSKIP STREAM EMACS.NONCR])
```

(EMACS.FWD.BYTE

```
[LAMBDA (STREAM)                                        (* kbr%: "19-Feb-85 15:12")
                                                        (* Go forward a byte. *)
    (PROG NIL
          (EMACS.FBYTE STREAM])
```

(EMACS.KILL.LINE

```
[LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
                                                        (* Delete a line. *)
    (PROG (PTR EOL)
          (SETQ PTR (GETFILEPTR STREAM))
          (EMACS.FSKIP STREAM EMACS.NONCR)
          (SETQ EOL (GETFILEPTR STREAM))
          (COND
             ((IGREATERP EOL PTR)
              (EMACS.DELETE.CHARS STREAM PTR (SUB1 EOL)))
             ((ILESSP EOL (GETEOFPTR STREAM))
              (EMACS.DELETE.CHARS STREAM EOL EOL)))
          (EMACS.SETFILEPTR STREAM PTR])
```

(EMACS.DELETE.CHARS

```
[LAMBDA (STREAM PTR1 PTR2)                              (* kbr%: "18-Jun-86 23:23")
                                                        (* Delete between PTR1 & PTR2 inclusive.
                                                        *)
    (PROG (PTR LENGTH)
          (SETQ PTR (GETFILEPTR STREAM))
          (SETQ PTR1 (IMAX 0 PTR1))
          (SETQ PTR2 (IMIN (GETEOFPTR STREAM)
                           PTR2))
          (SETQ LENGTH (IPLUS PTR2 (IMINUS PTR1)
                              1))
          (TEDIT.DELETE STREAM (ADD1 PTR1)
                 LENGTH)
          (COND
             ((ILEQ PTR PTR1)
              (SETFILEPTR STREAM PTR))
             ((ILEQ PTR PTR2)
              (SETFILEPTR STREAM PTR1))
             (T (SETFILEPTR STREAM (IDIFFERENCE PTR LENGTH])
```

(EMACS.REDISPLAY

```
[LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
                                                        (* Redisplay EMACS screen. *)
    (PROG (PTR)
```

```
          (SETQ PTR (GETFILEPTR STREAM))
          (REDISPLAYW (fetch (EMACSSTREAM WINDOW) of STREAM))
          (EMACS.SETFILEPTR STREAM PTR])
```

(EMACS.NEXT.LINE

```
  [LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
                                                          (* Go down a line. *)
    (PROG (PTR BOL EOL NBOL NEOL OFFSET)
          (SETQ PTR (GETFILEPTR STREAM))
          (SETQ BOL (EMACS.BOL STREAM PTR))               (* First char on line is at OFFSET = 0.0 *)
          (SETQ OFFSET (IPLUS PTR (IMINUS BOL)))
          (SETQ EOL (EMACS.EOL STREAM PTR))
          (SETQ NBOL (ADD1 EOL))
          (COND
             ((ILEQ (GETEOFPTR STREAM)
                    NBOL)
              (EMACS.SETFILEPTR STREAM (GETEOFPTR STREAM)))
             (T (SETQ NEOL (EMACS.EOL STREAM NBOL))
                (SETQ OFFSET (IMIN OFFSET (IDIFFERENCE NEOL NBOL)))
                (EMACS.SETFILEPTR STREAM (IPLUS NBOL OFFSET])
```

(EMACS.PREVIOUS.LINE

```
  [LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
                                                          (* Go up a line. *)
    (PROG (PTR BOL PBOL PEOL OFFSET)
          (SETQ PTR (GETFILEPTR STREAM))
          (SETQ BOL (EMACS.BOL STREAM PTR))               (* First char on line is at OFFSET = 0.0 *)
          (SETQ OFFSET (IPLUS PTR (IMINUS BOL)))
          (SETQ PEOL (SUB1 BOL))
          (COND
             ((IGEQ 0 PEOL)
              (EMACS.SETFILEPTR STREAM 0))
             (T (SETQ PBOL (EMACS.BOL STREAM PEOL))
                (SETQ OFFSET (IMIN OFFSET (IDIFFERENCE PEOL PBOL)))
                (EMACS.SETFILEPTR STREAM (IPLUS PBOL OFFSET])
```

(EMACS.QUOTE.BYTE

```
  [LAMBDA (STREAM)                                        (* kbr%: "18-Jun-86 22:59")
                                                          (* Quote next byte. *)
    (PROG (PTR CH)                                        (* TBW%: Fix use TEDIT's use of terminal table.
                                                          *)
          (SETQ PTR (GETFILEPTR STREAM))
          (SETQ CH (\GETKEY))
          (TEDIT.INSERT STREAM CH (ADD1 PTR))
          (EMACS.SETFILEPTR STREAM (ADD1 PTR])
```

(EMACS.SEARCH

```
  [LAMBDA (STREAM)                                        (* kbr%: "18-Jun-86 23:12")
                                                          (* Case sensitive search, with "*" and "#" wildcards *)
    (PROG (PTR TEXTOBJ W OFILE SEL CH)
          (SETQ PTR (GETFILEPTR STREAM))
          (SETQ TEXTOBJ (fetch (EMACSSTREAM TEXTOBJ) of STREAM))
          (SETQ W (fetch (EMACSSTREAM WINDOW) of STREAM))
          (ERSETQ (RESETLST
                      [RESETSAVE (\TEDIT.MARKACTIVE TEXTOBJ)
                             '(AND (\TEDIT.MARKINACTIVE OLDVALUE]
                      (replace (TEXTOBJ EDITOPACTIVE) of TEXTOBJ with 'Find)
                      (SETQ OFILE (WINDOWPROP W 'TEDIT.LAST.FIND.STRING))
                      [SETQ OFILE (TEDIT.GETINPUT STREAM "Text to find: " OFILE (CHARCODE (EOL LF ESC ^S]
                      [COND
                         (OFILE (WINDOWPROP W 'TEDIT.LAST.FIND.STRING OFILE)
                                (SETQ SEL (fetch (TEXTOBJ SEL) of TEXTOBJ))
                                (\SHOWSEL SEL NIL NIL)
                                (RESETLST
                                    (RESETSAVE (CURSOR WAITINGCURSOR))
                                    (SETQ CH (TEDIT.FIND TEXTOBJ (MKSTRING OFILE)
                                                         NIL NIL T)))
                                (COND
                                   (CH                    (* We found the target text.)
                                                          (* Set up SELECTION to be the found text)
                                       (replace (SELECTION CH#) of SEL with (CAR CH))
                                       (replace (SELECTION CHLIM) of SEL with (CADR CH))
                                       [replace (SELECTION DCH) of SEL with (ADD1 (IDIFFERENCE (CADR CH)
                                                                                               (CAR CH]
                                       (replace (SELECTION POINT) of SEL with 'RIGHT)
                                       (TEDIT.RESET.EXTEND.PENDING.DELETE SEL)
                                                          (* And never pending a deletion.)
                                       (\FIXSEL SEL TEXTOBJ)
                                       (TEDIT.NORMALIZECARET TEXTOBJ)
                                       (\SHOWSEL SEL NIL T)
                                       (EMACS.SETFILEPTR STREAM (EMACS.GETCARETPTR STREAM))
                                                          (* And get it into the window *)
                                       )
```

```
                                    (T (FRESHLINE PROMPTWINDOW)
                                       (printout PROMPTWINDOW "String '" OFILE "' not found." T)
                                       (\SHOWSEL SEL NIL T)
                                       (EMACS.SETFILEPTR STREAM PTR]
                              (replace (TEXTOBJ \INSERTNEXTCH) of TEXTOBJ with -1))])
```

(EMACS.TRANSPOSE.BYTES

```
  [LAMBDA (STREAM)                                                    (* kbr%: "19-Feb-85 15:12")
                                                                      (* Transpose bytes. *)
    (PROG (PTR CODE CH)
          (COND
             ((OR (EMACS.BOFP STREAM)
                  (EMACS.EOFP STREAM))
              (RETURN)))
          (SETQ PTR (GETFILEPTR STREAM))
          (SETQ CODE (\BIN STREAM))
          (COND
             [(NUMBERP CODE)
              (SETQ CH (MKSTRING (CHARACTER CODE]
             (T                                                       (* IMAGEOBJ *)
                (SETQ CH CODE)))
          (EMACS.DELETE.BYTES STREAM PTR PTR)
          (EMACS.SETFILEPTR STREAM (SUB1 PTR))
          (TEDIT.INSERT STREAM CH PTR)
          (EMACS.SETFILEPTR STREAM (ADD1 PTR])
```

(EMACS.NEXT.SCREENFULL

```
  [LAMBDA (STREAM)                                                    (* kbr%: "24-Jul-85 16:36")
                                                                      (* Forward one screenfull. *)
    (PROG (WINDOW DELTAX DELTAY)
          (SETQ WINDOW (fetch (EMACSSTREAM WINDOW) of STREAM))
          (SETQ DELTAX 0)
          [SETQ DELTAY (IDIFFERENCE (WINDOWPROP WINDOW 'HEIGHT)
                                    (FONTPROP (DSPFONT NIL WINDOW)
                                              'HEIGHT]
          (replace (TEXTOBJ EDITOPACTIVE) of (fetch (EMACSSTREAM TEXTOBJ) of STREAM) with NIL)
          (\TEDIT.SCROLLFN WINDOW DELTAX DELTAY])
```

(EMACS.CXCV

```
  [LAMBDA (STREAM)                                                    (* kbr%: "24-Jul-85 16:36")
                                                                      (* Visit a file. *)
    (PROG (FILE)
          (SETQ FILE (TEDIT.GETINPUT (fetch (EMACSSTREAM TEXTOBJ) of STREAM)
                                     "File to GET:"))
          (COND
             ((NULL FILE)
              (RETURN)))
          (COND
             ((fetch (EMACSSTREAM DIRTY) of STREAM)
              (EMACS.CXCW STREAM)))
          (TEDIT.GET (fetch (EMACSSTREAM TEXTOBJ) of STREAM)
                 FILE)
          (EMACS.SETFILEPTR STREAM 0])
```

(EMACS.CXCW

```
  [LAMBDA (STREAM)                                                    (* kbr%: "24-Jul-85 16:36")
                                                                      (* Write buffer out to file. *)
    (PROG NIL
          (TEDIT.PUT (fetch (EMACSSTREAM TEXTOBJ) of STREAM])
```

(EMACS.CXCZ

```
  [LAMBDA (STREAM)                                                    (* kbr%: "24-Jul-85 16:36")
                                                                      (* Eval FORM in front of caret in lisp EXEC process.
                                                                      *)
    (PROG (FORM)
          (SETQ FORM (READ STREAM EMACS.READTABLE))
          (PROCESS.EVAL 'EXEC FORM])
```

(EMACS.FWD.SEXPR

```
  [LAMBDA (STREAM)                                                    (* kbr%: "24-Jul-85 16:36")
                                                                      (* Go forward a sexpr. *)
    (PROG NIL
          (RESETLST                                                   (* Accept uncaught BQUOTE commas.
                                                                      *)
              (RESETSAVE \BQUOTELEVEL (IQUOTIENT MAX.FIXP 2))
              (READ STREAM EMACS.READTABLE))])
```

(EMACS.BACK.DELETE.BYTE

```
  [LAMBDA (STREAM)                                                    (* kbr%: "19-Feb-85 15:12")
                                                                      (* Delete byte in backwards direction.
                                                                      *)
```

```
                                                                    (* TBW%: Delete selection if there is a selection.
                                                                    *)
    (PROG (PTR)
          (SETQ PTR (GETFILEPTR STREAM))
          (EMACS.DELETE.BYTES STREAM (SUB1 PTR)
                 (SUB1 PTR])
```

(EMACS.GOTO.BOD

```
  [LAMBDA (STREAM)                                                  (* kbr%: "24-Jul-85 16:36")
                                                                    (* Go to top of definition. *)
    (PROG (CODE)                                                    (* Find non-WS immediately preceded by CR.
                                                                    *)
          (EMACS.BCHAR STREAM)
          (DO (COND
                 ((EMACS.BOFP STREAM)
                  (RETURN)))
              (EMACS.BSKIP STREAM EMACS.NONCR)
              (COND
                 ((BITTEST (ELT EMACS.SYNTAX (EMACS.CCHAR STREAM))
                           EMACS.NONWS)
                  (RETURN)))
              (EMACS.BCHAR STREAM])
```

(EMACS.BOD

```
  [LAMBDA (STREAM)                                                  (* kbr%: "19-Feb-85 15:12")
                                                                    (* Determine top of definition. *)
    (PROG (PTR ANSWER)
          (SETQ PTR (GETFILEPTR STREAM))                            (* Find lparen preceded by CR. *)
          (EMACS.BSKIP STREAM EMACS.CR)
          (DO (COND
                 ((EMACS.BOFP STREAM)
                  (RETURN)))
              (EMACS.BSKIP STREAM EMACS.NONCR)
              (COND
                 ((EMACS.BOFP STREAM)
                  (RETURN)))
              (COND
                 ((OR (EMACS.BOFP STREAM)
                      (EQ (\PEEKBIN STREAM)
                          (CHARCODE "(")))
                  (RETURN)))
              (EMACS.BBYTE STREAM))
          (SETQ ANSWER (GETFILEPTR STREAM))
          (SETFILEPTR STREAM PTR)
          (RETURN ANSWER])
```

(EMACS.GOTO.EOD

```
  [LAMBDA (STREAM)                                                  (* kbr%: "24-Jul-85 16:36")
                                                                    (* Go to top of next definition. *)
    (PROG (CODE)                                                    (* Find non-WS immediately preceded by CR.
                                                                    *)
          (EMACS.FCHAR STREAM)
          (DO (COND
                 ((EMACS.EOFP STREAM)
                  (RETURN)))
              (EMACS.FSKIP STREAM EMACS.NONCR)
              (EMACS.FCHAR STREAM)
              (COND
                 ((BITTEST (ELT EMACS.SYNTAX (EMACS.CCHAR STREAM))
                           EMACS.NONWS)
                  (RETURN])
```

(EMACS.EOD

```
  [LAMBDA (STREAM)                                                  (* kbr%: "19-Feb-85 15:12")
                                                                    (* Determine top of next definition.
                                                                    *)
    (PROG (PTR ANSWER)
          (SETQ PTR (GETFILEPTR STREAM))                            (* Find two CRs. *)
          (EMACS.FSKIP STREAM EMACS.CR)
          [DO (COND
                 ((EMACS.EOFP STREAM)
                  (RETURN)))
              (EMACS.FSKIP STREAM EMACS.NONCR)
              (COND
                 ((EMACS.EOFP STREAM)
                  (RETURN)))
              (EMACS.FBYTE STREAM)
              (COND
                 ((OR (EMACS.EOFP STREAM)
                      (EQ (\PEEKBIN STREAM)
                          (CHARCODE CR)))
                  (\BACKBIN STREAM)
                  (RETURN]
```

```
          (SETQ ANSWER (GETFILEPTR STREAM))
          (SETFILEPTR STREAM PTR)
          (RETURN ANSWER])
```

(EMACS.KILL.SEXPR

```
  [LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
                                                          (* Delete expression. *)
    (PROG (PTR1 PTR2)
          (SETQ PTR1 (GETFILEPTR STREAM))
          (READ STREAM EMACS.READTABLE)
          (SETQ PTR2 (GETFILEPTR STREAM))
          (EMACS.DELETE.CHARS STREAM PTR1 (SUB1 PTR2))
          (EMACS.SETFILEPTR STREAM PTR1])
```

(EMACS.GOTO.BOF

```
  [LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
                                                          (* Go to beginning of file. *)
    (PROG NIL
          (EMACS.SETFILEPTR STREAM 0])
```

(EMACS.GOTO.EOF

```
  [LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
                                                          (* Go to end of file. *)
    (PROG NIL
          (EMACS.SETFILEPTR STREAM (GETEOFPTR STREAM])
```

(EMACS.BACK.WORD

```
  [LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
                                                          (* Backward a word. *)
    (PROG NIL
          (EMACS.BWORD STREAM])
```

(EMACS.FWD.DELETE.WORD

```
  [LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
                                                          (* Delete word. *)
    (PROG (PTR1 PTR2)
          (SETQ PTR1 (GETFILEPTR STREAM))
          (EMACS.FSKIP STREAM EMACS.WS)
          (EMACS.FSKIP STREAM EMACS.NONWS)
          (SETQ PTR2 (GETFILEPTR STREAM))
          (EMACS.DELETE.CHARS STREAM PTR1 (SUB1 PTR2))
          (EMACS.SETFILEPTR STREAM PTR1])
```

(EMACS.EDIT

```
  [LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
                                                          (* DEDIT expression. *)
    (PROG (EXPR PTR1 PTR2)
          (SKIPSEPRS STREAM)
          (SETQ PTR1 (GETFILEPTR STREAM))
          (SETQ EXPR (READ STREAM EMACS.READTABLE))
          (SETQ PTR2 (GETFILEPTR STREAM))
          (EMACS.DELETE.CHARS STREAM PTR1 (SUB1 PTR2))
          (SETQ EXPR (EDITE EXPR))
          (PRINTDEF EXPR NIL NIL NIL NIL STREAM])
```

(EMACS.FWD.WORD

```
  [LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
                                                          (* Forward a word. *)
    (PROG NIL
          (EMACS.FWORD STREAM])
```

(EMACS.GRIND

```
  [LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
                                                          (* Grind expression. *)
    (PROG (EXPR PTR1 PTR2)
          (SKIPSEPRS STREAM)
          (SETQ PTR1 (GETFILEPTR STREAM))
          (SETQ EXPR (READ STREAM EMACS.READTABLE))
          (SETQ PTR2 (GETFILEPTR STREAM))
          (EMACS.DELETE.CHARS STREAM PTR1 (SUB1 PTR2))
          (PRINTDEF EXPR NIL NIL NIL NIL STREAM])
```

(EMACS.SNARF

```
  [LAMBDA (STREAM)                                        (* kbr%: "24-Jul-85 16:36")
                                                          (* Snarf expression from DEDIT window.
                                                          *)
    (PROG (EXPR)
          (SETQ EXPR (CAR (TOPSELECTION)))
```

```
          (PRINTDEF EXPR NIL NIL NIL NIL STREAM])
```

(EMACS.MT

```
  [LAMBDA (STREAM)                                          (* kbr%: "24-Jul-85 16:36")
                                                            (* Transpose words. *)
    (PROG (PTR BPTR1 BPTR2 FPTR1 FPTR2)
          (SETQ PTR (GETFILEPTR STREAM))
          (EMACS.BSKIP STREAM EMACS.WS)
          (SETQ BPTR2 (GETFILEPTR STREAM))
          (EMACS.BWORD)
          (SETQ BPTR1 (GETFILEPTR STREAM))
          (EMACS.SETFILEPTR STREAM PTR)
          (EMACS.FSKIP STREAM EMACS.WS)
          (SETQ FPTR1 (GETFILEPTR STREAM))
          (EMACS.FWORD STREAM)
          (SETQ FPTR2 (GETFILEPTR STREAM))                  (* How do I move? *)
      ])
```

(EMACS.PREVIOUS.SCREENFULL

```
  [LAMBDA (STREAM)                                          (* kbr%: "24-Jul-85 16:36")
                                                            (* Backwards a screenfull. *)
    (PROG (WINDOW DELTAX DELTAY)
          (SETQ WINDOW (fetch (EMACSSTREAM WINDOW) of STREAM))
          (SETQ DELTAX 0)
          [SETQ DELTAY (IDIFFERENCE (FONTPROP (DSPFONT NIL WINDOW)
                                              'HEIGHT)
                                    (WINDOWPROP WINDOW 'HEIGHT]
          (replace (TEXTOBJ EDITOPACTIVE) of (fetch (EMACSSTREAM TEXTOBJ) of STREAM) with NIL)
          (\TEDIT.SCROLLFN WINDOW DELTAX DELTAY])
```

(EMACS.JOIN.LINES

```
  [LAMBDA (STREAM)                                          (* kbr%: "24-Jul-85 16:36")
                                                            (* Move current line up *)
    (PROG (PTR BOL EOL PBOL PEOL PTR1 PTR2)
          (SETQ PTR (GETFILEPTR STREAM))
          (SETQ BOL (EMACS.BOL STREAM PTR))
          (SETQ EOL (EMACS.EOL STREAM PTR))
          (COND
             ((ZEROP BOL)
              (RETURN)))
          (SETQ PEOL (SUB1 BOL))
          (SETQ PBOL (EMACS.BOL STREAM PEOL))
          (EMACS.SETFILEPTR STREAM BOL)
          (EMACS.BSKIP STREAM EMACS.WS)
          (SETQ PTR1 (IMAX (GETFILEPTR STREAM)
                           PBOL))
          (EMACS.SETFILEPTR STREAM BOL)
          (EMACS.FSKIP STREAM EMACS.WS)
          (SETQ PTR2 (IMIN (GETFILEPTR STREAM)
                           (ADD1 EOL)))
          (EMACS.SETFILEPTR STREAM PTR1)
          (EMACS.DELETE.CHARS STREAM PTR1 (SUB1 PTR2))
          (\BOUT STREAM (CHARCODE SP))
          (EMACS.SETFILEPTR STREAM (ADD1 PTR1])
```

(EMACS.BACK.DELETE.WORD

```
  [LAMBDA (STREAM)                                          (* kbr%: "24-Jul-85 16:36")
                                                            (* Delete backward a word. *)
    (PROG (PTR1 PTR2)
          (SETQ PTR1 (GETFILEPTR STREAM))
          (EMACS.BWORD STREAM)
          (SETQ PTR2 (GETFILEPTR STREAM))
          (EMACS.DELETE.CHARS STREAM PTR2 (SUB1 PTR1])
```

(NEW.TEDIT.SELECT.LINE.SCANNER

```
  [LAMBDA (X Y TEXTOBJ LINE.LIST REGION WORDSELFLG SELOPERATION WINDOW)
                                                            (* kbr%: "24-Jul-85 16:49")
    (PROG (SELECTION PTR)
          (SETQ SELECTION (OLD.TEDIT.SELECT.LINE.SCANNER X Y TEXTOBJ LINE.LIST REGION WORDSELFLG SELOPERATION
                                WINDOW))
          [COND
             ((EQ (TYPENAME SELECTION)
                  'SELECTION)
              (replace (SELECTION POINT) of SELECTION with 'LEFT)
              (EMACS.SETFILEPTR (fetch (TEXTOBJ STREAMHINT) of TEXTOBJ)
                     (SUB1 (fetch (SELECTION CH#) of SELECTION]
          (EMACS.FLUSH.CACHE)
          (RETURN SELECTION])

)
```

```
          (* * BALANCE *)

(PUTPROPS ACCESSFNS EMACS.TAB 2)

(PUTPROPS DATATYPE EMACS.TAB 2)

(PUTPROPS DEFEXPR EMACS.TAB 2)

(PUTPROPS DEFFEXPR EMACS.TAB 2)

(PUTPROPS DEFVAR EMACS.TAB 2)

(PUTPROPS DO EMACS.TAB 1)

(PUTPROPS FOR EMACS.TAB 1)

(PUTPROPS LAMBDA EMACS.TAB 2)

(PUTPROPS PROG EMACS.TAB 2)

(PUTPROPS RECORD EMACS.TAB 2)

(PUTPROPS SELECT EMACS.TAB 2)

(PUTPROPS SELECTQ EMACS.TAB 2)

(PUTPROPS UNTIL EMACS.TAB 1)

(PUTPROPS WHILE EMACS.TAB 1)

(RPAQ? EMACS.DELIMS NIL)

(RPAQ? EMACS.SDELIMS NIL)

(RPAQ? EMACS.LDELIMS NIL)

(RPAQ? EMACS.RDELIMS NIL)

(RPAQ? EMACS.SCACHE NIL)

(RPAQ? EMACS.BCACHE NIL)

(RPAQ? EMACS.SYNTAX NIL)

(RPAQ? EMACS.CR 1)

(RPAQ? EMACS.WS 2)

(RPAQ? EMACS.SD 4)

(RPAQ? EMACS.NONCR 8)

(RPAQ? EMACS.NONWS 16)

(RPAQ? EMACS.NONSD 32)

(RPAQ? EMACS.BQ 64)

(RPAQ? EMACS.ALPHA 128)

(RPAQ? EMACS.BD 256)

(RPAQ? EMACS.SPACE 512)

(DEFINEQ

(EMACS.DELIMS
  [LAMBDA (LCHARCODE RCHARCODE)                                    (* kbr%: "19-Feb-85 15:13")

          (* Make LCHARCODE & RCHARCODE into delimiters. If LCHARCODE = RCHARCODE, then string style.
          Otherwise paren style. *)

    (PROG (BUCKET)
          (SETQ BUCKET (CONS LCHARCODE RCHARCODE))
          (COND
             ((MEMBER BUCKET EMACS.DELIMS)                         (* Already there. *)
              (RETURN)))
          (PUSH EMACS.DELIMS BUCKET)
          (COND
             ((IEQP LCHARCODE RCHARCODE)
              (SETSYNTAX LCHARCODE 'STRINGDELIM EMACS.READTABLE)
              (SETA EMACS.SYNTAX LCHARCODE (LOGOR EMACS.NONWS EMACS.NONCR EMACS.SD))
              [SETA EMACS.COMMANDS LCHARCODE (EMACS.SDELIM.COMMAND (MKSTRING (CHARACTER LCHARCODE]
              (PUSH EMACS.SDELIMS LCHARCODE))
             (T (SETSYNTAX LCHARCODE 'LEFTPAREN EMACS.READTABLE)
                (SETSYNTAX RCHARCODE 'RIGHTPAREN EMACS.READTABLE)
                (FOR I IN (LIST LCHARCODE RCHARCODE) DO (SETA EMACS.SYNTAX I (LOGOR EMACS.NONCR EMACS.NONWS
```

```
                                                                    EMACS.NONSD EMACS.BD)))
        [SETA EMACS.COMMANDS LCHARCODE (EMACS.LDELIM.COMMAND (MKSTRING (CHARACTER LCHARCODE]
        [SETA EMACS.COMMANDS RCHARCODE (EMACS.RDELIM.COMMAND (MKSTRING (CHARACTER RCHARCODE]
        (PUSH EMACS.LDELIMS LCHARCODE)
        (PUSH EMACS.RDELIMS RCHARCODE])
```

(EMACS.CR

```
[LAMBDA (STREAM)                                        (* kbr%: "19-Feb-85 15:14")
  (PROG (PTR)
        (SETQ PTR (GETFILEPTR STREAM))
        (TEDIT.INSERT STREAM (CHARACTER (CHARCODE CR)))
        (EMACS.SETFILEPTR STREAM (ADD1 PTR))
        (COND
           ((NOT (EQ (EMACS.SCACHE STREAM)
                     'OUTSIDE))
            (FLASHWINDOW (fetch (EMACSSTREAM WINDOW) of STREAM))
            (EMACS.SETCARETPTR STREAM EMACS.SCACHE)
            (DISMISS 1000)
            (EMACS.SETCARETPTR STREAM (ADD1 PTR))
            (SETQ EMACS.SCACHE 'OUTSIDE)
            (SETQ EMACS.BCACHE NIL])
```

(EMACS.RPAREN

```
[LAMBDA (STREAM)                                        (* kbr%: "19-Feb-85 15:14")
  (PROG (PTR)
        (SETQ PTR (GETFILEPTR STREAM))
        (TEDIT.INSERT STREAM ")")
        (EMACS.SETFILEPTR STREAM (ADD1 PTR))
        (COND
           ((EQ (EMACS.SCACHE STREAM)
                'OUTSIDE)
            (EMACS.CLOSE.BALANCE STREAM])
```

(EMACS.RBRACKET

```
[LAMBDA (STREAM)                                        (* kbr%: "19-Feb-85 15:14")
  (PROG (PTR)
        (SETQ PTR (GETFILEPTR STREAM))
        (TEDIT.INSERT STREAM "]")
        (EMACS.SETFILEPTR STREAM (ADD1 PTR))
        (COND
           ((EQ (EMACS.SCACHE STREAM)
                'OUTSIDE)
            (EMACS.CLOSE.BALANCE STREAM])
```

(EMACS.RBRACE

```
[LAMBDA (STREAM)                                        (* kbr%: "19-Feb-85 15:14")
  (PROG (PTR)
        (SETQ PTR (GETFILEPTR STREAM))
        (TEDIT.INSERT STREAM "}")
        (EMACS.SETFILEPTR STREAM (ADD1 PTR))
        (COND
           ((EQ (EMACS.SCACHE STREAM)
                'OUTSIDE)
            (EMACS.CLOSE.BALANCE STREAM])
```

(EMACS.RANGLE

```
[LAMBDA (STREAM)                                        (* kbr%: "19-Feb-85 15:14")
  (PROG (PTR)
        (SETQ PTR (GETFILEPTR STREAM))
        (TEDIT.INSERT STREAM ">")
        (EMACS.SETFILEPTR STREAM (ADD1 PTR))
        (COND
           ((EQ (EMACS.SCACHE STREAM)
                'OUTSIDE)
            (EMACS.CLOSE.BALANCE STREAM])
```

(EMACS.SDELIM.COMMAND

```
[LAMBDA (SDELIM)                                        ; Edited 19-Feb-2021 11:21 by rmk:
                                                        (* kbr%: "19-Feb-85 15:14")
                                                        (* Return sdelim fn to be inserted in EMACS.COMMANDS.
                                                        SDELIM = 1 letter string. *)

  `(LAMBDA (STREAM)
     (EMACS.SDELIM ,SDELIM STREAM])
```

(EMACS.LDELIM.COMMAND

```
[LAMBDA (LDELIM)                                        ; Edited 19-Feb-2021 11:20 by rmk:
                                                        (* kbr%: "19-Feb-85 15:14")
                                                        (* Return LDELIM fn to be inserted in EMACS.COMMANDS.
                                                        LDELIM = 1 letter string. *)

  `(LAMBDA (STREAM)
```

```
      (EMACS.LDELIM ,LDELIM STREAM])
```

(EMACS.RDELIM.COMMAND

```
  [LAMBDA (RDELIM)                                    ; Edited 19-Feb-2021 11:20 by rmk:
                                                      (* kbr%: "19-Feb-85 15:14")
                                                      (* Return RDELIM fn to be inserted in EMACS.COMMANDS.
                                                      RDELIM = 1 letter string. *)

    `(LAMBDA (STREAM)
       (EMACS.RDELIM ,RDELIM STREAM])
```

(EMACS.SDELIM

```
  [LAMBDA (SDELIM STREAM)                             (* kbr%: "19-Feb-85 15:14")
                                                      (* Insert string delimiter SDELIM & update caches.
                                                      SDELIM = 1 letter string *)

    (PROG (PTR)
          (SETQ PTR (GETFILEPTR STREAM))
          (TEDIT.INSERT STREAM SDELIM)
          (EMACS.SETFILEPTR STREAM (ADD1 PTR))
          (COND
             ((EMACS.BACK.ESCAPEDP STREAM)
              (RETURN)))
          (COND
             ((EQ (EMACS.SCACHE STREAM)
                  'OUTSIDE)
              (EMACS.OPEN.STRING STREAM))
             (T (EMACS.CLOSE.STRING STREAM])
```

(EMACS.LDELIM

```
  [LAMBDA (LDELIM STREAM)                             (* kbr%: "19-Feb-85 15:14")
                                                      (* Insert LDELIM & update caches.
                                                      LDELIM = 1 letter string *)

    (PROG (PTR)
          (SETQ PTR (GETFILEPTR STREAM))
          (TEDIT.INSERT STREAM LDELIM)
          (EMACS.SETFILEPTR STREAM (ADD1 PTR))
          (COND
             ((EMACS.BACK.ESCAPEDP STREAM)
              (RETURN)))
          (COND
             ((EQ (EMACS.SCACHE STREAM)
                  'OUTSIDE)
              (EMACS.OPEN.BALANCE STREAM])
```

(EMACS.RDELIM

```
  [LAMBDA (RDELIM STREAM)                             (* kbr%: "19-Feb-85 15:14")
                                                      (* Insert RDELIM & update caches.
                                                      RDELIM = 1 letter string *)

    (PROG (PTR)
          (SETQ PTR (GETFILEPTR STREAM))
          (TEDIT.INSERT STREAM RDELIM)
          (EMACS.SETFILEPTR STREAM (ADD1 PTR))
          (COND
             ((EMACS.BACK.ESCAPEDP STREAM)
              (RETURN)))
          (COND
             ((EQ (EMACS.SCACHE STREAM)
                  'OUTSIDE)
              (EMACS.CLOSE.BALANCE STREAM])
```

(EMACS.OPEN.STRING

```
  [LAMBDA (STREAM)                                    (* kbr%: "19-Feb-85 15:14")
    (PROG (LPTR)                                      (* We should be 1 char after left delim.
                                                      *)
          (SETQ LPTR (SUB1 (GETFILEPTR STREAM)))
          (SETQ EMACS.SCACHE LPTR])
```

(EMACS.CLOSE.STRING

```
  [LAMBDA (STREAM)                                    (* kbr%: "19-Feb-85 15:14")
    (PROG (LPTR RPTR LDELIM RDELIM MATCHED)           (* We should be 1 char after right delim.
                                                      *)
          (SETQ EMACS.SCACHE 'OUTSIDE)
          (SETQ RPTR (SUB1 (GETFILEPTR STREAM)))
          (EMACS.SETFILEPTR STREAM RPTR)
          (SETQ RDELIM (\PEEKBIN STREAM))
          (EMACS.BSKIP STREAM EMACS.NONSD)
          (EMACS.BBYTE STREAM)
          (SETQ LPTR (GETFILEPTR STREAM))
          (SETQ LDELIM (\PEEKBIN STREAM))
          (SETQ MATCHED (IEQP LDELIM RDELIM))
          (COND
             (MATCHED (EMACS.SETCARETPTR STREAM LPTR)
```

```
                (DISMISS 200))
            (T (FLASHWINDOW (fetch (EMACSSTREAM WINDOW) of STREAM))
               (EMACS.SETCARETPTR STREAM LPTR)
               (DISMISS 1000)))
         (EMACS.SETCARETPTR STREAM (ADD1 RPTR))
         (EMACS.SETFILEPTR STREAM (ADD1 RPTR])
```

(EMACS.OPEN.BALANCE

```
(LAMBDA (STREAM)                                        (* kbr%: "19-Feb-85 15:14")
  (PROG (LPTR)                                          (* We should be 1 char after left delim.
                                                        *)
        (SETQ LPTR (SUB1 (GETFILEPTR STREAM)))
        (COND
           ((NUMBERP EMACS.BCACHE)                      (* We were at top level. *)
            (SETQ EMACS.BCACHE (LIST LPTR)))
           (T (push EMACS.BCACHE LPTR])
```

(EMACS.CLOSE.BALANCE

```
(LAMBDA (STREAM)                                        (* kbr%: "19-Feb-85 15:14")
  (PROG (PTR LPTR RPTR LDELIM RDELIM BALANCED)          (* LPTR & RPTR point at balancing delims *)
        (SETQ PTR (GETFILEPTR STREAM))
        (SETQ RPTR (SUB1 PTR))
        (EMACS.SETFILEPTR STREAM RPTR)
        (SETQ RDELIM (\PEEKBIN STREAM))
        (EMACS.BCACHE STREAM)
        (SETQ BALANCED (NOT (NUMBERP EMACS.BCACHE)))
        (COND
           (BALANCED (SETQ LPTR (CAR EMACS.BCACHE))
                  (EMACS.SETFILEPTR STREAM LPTR)
                  (SETQ LDELIM (\PEEKBIN STREAM))
                  (EMACS.SETCARETPTR STREAM LPTR)
                  (COND
                     ((IEQP (CDR (ASSOC LDELIM EMACS.DELIMS))
                            RDELIM)                     (* Correct match *)
                      (DISMISS 200))
                     (T                                 (* Flash incorrect match. *)
                        (FLASHWINDOW (fetch (EMACSSTREAM WINDOW) of STREAM))
                        (DISMISS 1000)))
                  (pop EMACS.BCACHE))
           (T                                           (* Flash beginning of non-list def.
                                                        *)
              (EMACS.SETCARETPTR STREAM EMACS.BCACHE)
              (FLASHWINDOW (fetch (EMACSSTREAM WINDOW) of STREAM))
              (DISMISS 1000)))
        (EMACS.SETCARETPTR STREAM PTR)
        (EMACS.SETFILEPTR STREAM PTR])
```

(EMACS.FLUSH.CACHE

```
(LAMBDA NIL                                             (* kbr%: "19-Feb-85 15:14")
                                                        (* Lose cached info about string & paren balancing.
                                                        *)
  (PROG NIL

        (* Hopefully we can change things so that not all commands flush all of cache.
        *)

        (SETQ EMACS.SCACHE NIL)
        (SETQ EMACS.BCACHE NIL])
```

(EMACS.SCACHE

```
(LAMBDA (STREAM)                                        (* kbr%: "19-Feb-85 15:14")
                                                        (* Return <number> or OUTSIDE, computing if necessary.
                                                        *)
  (PROG (PTR ANSWER)
        (COND
           (EMACS.SCACHE (RETURN EMACS.SCACHE)))        (* Recompute. *)
        (SETQ PTR (GETFILEPTR STREAM))
        (EMACS.SETFILEPTR STREAM (EMACS.BOL STREAM PTR))
        (SETQ ANSWER 'OUTSIDE)
        (while (ILESSP (GETFILEPTR STREAM)
                       PTR)
           do                                           (* Find opening. *)
              (EMACS.FSKIP STREAM EMACS.NONSD PTR)
              (EMACS.FBYTE STREAM)
              (COND
                 ((IGEQ (GETFILEPTR STREAM)
                        PTR)
                  (RETURN)))
              (SETQ ANSWER (GETFILEPTR STREAM))         (* Find closing. *)
              (EMACS.FSKIP STREAM EMACS.NONSD PTR)
              (EMACS.FBYTE STREAM)
              (COND
                 ((IGEQ (GETFILEPTR STREAM)
```

```
                        PTR)
                    (RETURN)))
               (SETQ ANSWER 'OUTSIDE))                    (* Store ANSWER, restore fileptr, & return *)
          (SETQ EMACS.SCACHE ANSWER)
          (EMACS.SETFILEPTR STREAM PTR)
          (RETURN ANSWER])
```

(EMACS.BCACHE

```
  [LAMBDA (STREAM)                                        (* kbr%: "19-Feb-85 15:14")
                                                          (* Return (<number1> |...| <numberN>) or OUTSIDE *)
    (PROG (PTR SCACHE ANSWER)
          (COND
             ((EMACS.BCACHE (RETURN EMACS.BCACHE)))       (* Recompute. *)
          (SETQ PTR (GETFILEPTR STREAM))
          (SETQ SCACHE (EMACS.SCACHE STREAM))
          [COND
             ((NOT (EQ SCACHE 'OUTSIDE))                  (* Move off string. *)
              (EMACS.SETFILEPTR STREAM SCACHE)
              (COND
                 ((OR (EMACS.BOFP STREAM)
                      (IEQP (\BACKPEEKBIN STREAM)
                            (CHARCODE CR)))               (* A string def! *)
                  (SETQ ANSWER SCACHE)
                  (GO EXIT]
          [COND
             ((NULL (EMACS.SAFE.BACK.SEXPRS STREAM))      (* Unsuccessful read = unbalanced parens.
                                                          Treat as if top level. *)
              (SETQ ANSWER (GETFILEPTR STREAM)))
             ((OR (ZEROP (GETFILEPTR STREAM))
                  (IEQP (\BACKPEEKBIN STREAM)
                        (CHARCODE CR)))                   (* Top level. *)
              (SETQ ANSWER (GETFILEPTR STREAM)))
             (T                                           (* Opening delim present. *)
                (SETQ ANSWER (LIST (SUB1 (GETFILEPTR STREAM]
      EXIT
          (EMACS.SETFILEPTR STREAM PTR)
          (SETQ EMACS.BCACHE ANSWER)
          (RETURN ANSWER])
```

(EMACS.SAFE.BACK.SEXPRS

```
  [LAMBDA (STREAM)                                        (* kbr%: "19-Feb-85 15:14")

          (* Backwards read sexprs up to but not including opening delim.
          Return T if successful backwards read. Otherwise NIL & leave fileptr near failure point.
          *)

    (PROG (ANSWER)
          [DO (EMACS.BACK.SKIPSEPRS STREAM)
              (COND
                 ([OR (ZEROP (GETFILEPTR STREAM))
                      (IEQP (\BACKPEEKBIN STREAM)
                            (CHARCODE CR))
                      (AND (FMEMB (\BACKPEEKBIN STREAM)
                                  EMACS.LDELIMS)
                           (NOT (EMACS.BACK.ESCAPEDP STREAM]  (* Up against delimiter. *)
                  (SETQ ANSWER T)
                  (RETURN))
                 ((NULL (NLSETQ (EMACS.BACK.SEXPR STREAM)))  (* Error reading backwards. *)
                  (FLASHWINDOW STREAM)
                  (RETURN]
          (RETURN ANSWER])
```

(EMACS.SAFE.BACK.SEXPR

```
  [LAMBDA (STREAM)                                        (* kbr%: "19-Feb-85 15:14")

          (* Return T if successful backwards read. Otherwise NIL & leave fileptr near failure point.
          *)

    (PROG NIL
          (COND
             ((NULL (NLSETQ (EMACS.BACK.SEXPR STREAM)))   (* Error reading backwards. *)
              (FLASHWINDOW STREAM)
              (RETURN NIL)))
          (RETURN T])
```

(EMACS.BACK.SEXPR

```
  [LAMBDA (STREAM)                                        (* kbr%: "19-Feb-85 15:14")
    (PROG (RDELIM LDELIM)
          (EMACS.BACK.SKIPSEPRS STREAM)
          (COND
             ((EMACS.BOFP STREAM)
              (ERROR!))
             ((EMACS.BACK.ESCAPEDP STREAM)                (* Atom *)
```

```
            (EMACS.BACK.WORD STREAM)
            (RETURN)))
         (SETQ RDELIM (\BACKPEEKBIN STREAM))
         [SETQ LDELIM (for BUCKET in EMACS.DELIMS when (IEQP (CDR BUCKET)
                                                             RDELIM)
                         do (RETURN (CAR BUCKET]
         (COND
            ((NULL LDELIM)                                    (* Atom *)
             (EMACS.BACK.WORD STREAM))
            ((IEQP LDELIM RDELIM)                             (* String delimiters *)
             (\BACKBIN STREAM)
             (WHILE (AND (NOT (EMACS.BOFP STREAM))
                         (OR (NOT (IEQP (\BACKPEEKBIN STREAM)
                                        LDELIM))
                             (EMACS.BACK.ESCAPEDP STREAM)))
                DO (\BACKBIN STREAM))
             (COND
                ((EMACS.BOFP STREAM)
                 (ERROR!)))
             (\BACKBIN STREAM))
            (T                                                (* Left Right delimters *)
               (\BACKBIN STREAM)
               [do (EMACS.BACK.SKIPSEPRS STREAM)
                   (COND
                      ((EMACS.BOFP STREAM)
                       (ERROR!))
                      ((AND (FMEMB (\BACKPEEKBIN STREAM)
                                   EMACS.LDELIMS)
                            (NOT (EMACS.BACK.ESCAPEDP STREAM)))
                       (RETURN)))
                   (EMACS.BACK.SEXPR STREAM)
                   (COND
                      ((OR (EMACS.BOFP STREAM)
                           (IEQP (\BACKPEEKBIN STREAM)
                                 (CHARCODE CR)))              (* At top of definition in middle of read.
                                                              *)
                       (ERROR!]
               (\BACKBIN STREAM)
               (EMACS.BSKIP STREAM EMACS.BQ])
```

(EMACS.BACK.SKIPSEPRS

```
  [LAMBDA (STREAM)                                            (* kbr%: "19-Feb-85 15:14")
                                                              (* Backwards SKIPSEPRS. *)
    (PROG (SA CH SNX)
          (SETQ SA (fetch (READTABLEP READSA) of EMACS.READTABLE))
          (COND
             ((EMACS.BOFP STREAM)
              (RETURN)))
          (SETQ CH (\BACKPEEKBIN STREAM))
          (SETQ SNX (\GETBASEBYTE SA CH))
          (COND
             ((NOT (EQ SNX SEPRCHAR.RC))
              (RETURN)))
          (\BACKBIN STREAM)
          (do (COND
                 ((EMACS.BOFP STREAM)
                  (RETURN)))
              (SETQ CH (\BACKPEEKBIN STREAM))
              (SETQ SNX (\GETBASEBYTE SA CH))
              (COND
                 ((EQ SNX SEPRCHAR.RC)
                  (\BACKBIN STREAM))
                 ((EQ SNX ESCAPE.RC)
                  (\BIN STREAM)
                  (COND
                     ((NOT (EMACS.BACK.ESCAPEDP STREAM))
                      (\BACKBIN STREAM)))
                  (RETURN))
                 (T (RETURN])
```

(EMACS.BACK.ESCAPEDP

```
  [LAMBDA (STREAM)                                            (* kbr%: "19-Feb-85 15:14")
                                                              (* Is the previous byte escaped? *)
    (PROG (PTR SA CH SNX ANSWER)                              (* T if previous byte preceded by odd number of %%'s.
                                                              *)
          (SETQ PTR (GETFILEPTR STREAM))
          (COND
             ((ILEQ PTR 1)
              (RETURN NIL)))
          (SETQ SA (fetch (READTABLEP READSA) of EMACS.READTABLE))
          (\BACKBIN STREAM)
          [do (SETQ CH (\BACKBIN STREAM))
              (SETQ SNX (\GETBASEBYTE SA CH))
              (COND
                 ((EQ SNX ESCAPE.RC)
```

```
                (SETQ ANSWER (NOT ANSWER)))
               (T (RETURN)))
            (COND
               ((EMACS.BOFP STREAM)
                (RETURN]
         (SETFILEPTR STREAM PTR)
         (RETURN ANSWER])
```

(EMACS.TAB

```
  [LAMBDA (STREAM)                                                    (* kbr%: "19-Feb-85 15:14")
                                                                      (* Lisp indent. *)
    (PROG (PTR BOL EOL CODE INDENT OFFSET TABFLG)

          (* INDENT = how much we want to indent. OFFSET = how many chars to nonws.
          TABFLG = any tabs present at beginning of line. *)

          (SETQ PTR (GETFILEPTR STREAM))
          (SETQ INDENT (EMACS.TAB.INDENT STREAM))
          (SETQ BOL (EMACS.BOL STREAM PTR))                           (* Calc OFFSET. *)
          (SETQ EOL (EMACS.EOL STREAM PTR))
          (EMACS.SETFILEPTR STREAM BOL)
          (SETQ OFFSET 0)
          (for I from BOL to (SUB1 EOL) do (SETQ CODE (\BIN STREAM))
                                           (COND
                                              ((EQUAL CODE (CHARCODE TAB))
                                               (SETQ TABFLG T)))
                                           (COND
                                              ((NOT (BITTEST (ELT EMACS.SYNTAX (OR (NUMBERP CODE)
                                                                                   256))
                                                             EMACS.WS))
                                               (RETURN)))
                                           (SETQ OFFSET (ADD1 OFFSET)))
                                                                      (* Insert and/or delete whitespace.
                                                                      *)
          (COND
             [TABFLG (EMACS.DELETE.BYTES STREAM BOL (IPLUS BOL OFFSET -1))
                     (COND
                        ((NOT (ZEROP INDENT))
                         (TEDIT.INSERT STREAM (ALLOCSTRING INDENT " ")
                                (ADD1 BOL]
             ((IEQP OFFSET INDENT)                                    (* Do nothing. *)
              )
             ((IGREATERP OFFSET INDENT)
              (EMACS.DELETE.BYTES STREAM BOL (IPLUS BOL (IDIFFERENCE OFFSET INDENT)
                                                    -1)))
             ((ILESSP OFFSET INDENT)
              (TEDIT.INSERT STREAM (ALLOCSTRING (IDIFFERENCE INDENT OFFSET)
                                           " ")
                     (ADD1 BOL)))
             (T (SHOULDNT)))                                          (* Reposition fileptr. *)
          (COND
             ((ILEQ PTR (IPLUS BOL OFFSET))
              (EMACS.SETFILEPTR STREAM (IPLUS BOL INDENT)))
             (T (EMACS.SETFILEPTR STREAM (IPLUS PTR (IDIFFERENCE INDENT OFFSET])
```

(EMACS.TAB.INDENT

```
  [LAMBDA (STREAM)                                                    (* kbr%: "19-Feb-85 15:14")
                                                                      (* Amount to indent for Lisp indent.
                                                                      *)
    (PROG (PTR BOD SISTER1PTR SISTER2PTR LDELIMFLG SISTER1 SISTERPTR OFFSET BOL ANSWER)
          (SETQ PTR (GETFILEPTR STREAM))                              (* SETQ BOD (EMACS.BOD STREAM PTR))
          (SETQ BOD 0)
          (EMACS.SETFILEPTR STREAM (EMACS.BOL STREAM PTR))
          (EMACS.BSKIP STREAM EMACS.WS BOD)                           (* Get SISTER1PTR, SISTER2PTR, & LDELIMFLG *)
          (do (EMACS.BSKIP STREAM EMACS.SPACE)
              (COND
                 ((ILEQ (GETFILEPTR STREAM)
                        BOD)
                  (RETURN)))
              (COND
                 ((AND (FMEMB (\BACKPEEKBIN STREAM)
                              EMACS.LDELIMS)
                       (NOT (EMACS.BACK.ESCAPEDP STREAM)))
                  (SETQ LDELIMFLG T)
                  (RETURN)))
              (EMACS.SAFE.BACK.SEXPR STREAM)
              (SETQ SISTER2PTR SISTER1PTR)
              (SETQ SISTER1PTR (GETFILEPTR STREAM)))                  (* Get SISTER1. *)
          [COND
             (SISTER1PTR (EMACS.SETFILEPTR STREAM SISTER1PTR)
                    (SETQ SISTER1 (RATOM STREAM]                      (* Get SISTERPTR & OFFSET. *)
          (SETQ SISTERPTR (OR SISTER1PTR (GETFILEPTR STREAM)))
          [COND
             ((AND SISTER1 (LITATOM SISTER1))
              (SETQ OFFSET (GETPROP SISTER1 'EMACS.TAB]
```

```
          (COND
             (OFFSET (SETQ OFFSET (SUB1 OFFSET)))
             ((NULL SISTER1)
              (SETQ OFFSET 1))
             ((NULL LDELIMFLG)
              (SETQ OFFSET 0))
             ((NULL SISTER2PTR)
              (SETQ OFFSET 0))
             (T (SETQ SISTERPTR SISTER2PTR)
                (SETQ OFFSET 0)))                              (* Get ANSWER. *)
          (SETQ BOL (EMACS.BOL STREAM SISTERPTR))
          (EMACS.SETFILEPTR STREAM BOL)
          (SETQ ANSWER OFFSET)
          [for I from BOL to (SUB1 SISTERPTR) do (COND
                                                    ((IEQP (\BIN STREAM)
                                                           (CHARCODE TAB))
                                                     (SETQ ANSWER (IPLUS ANSWER 8)))
                                                    (T (SETQ ANSWER (ADD1 ANSWER]
      EXIT
          (EMACS.SETFILEPTR STREAM PTR)
          (RETURN ANSWER])
```

(EMACS.INIT.SYNTAX

```
  [LAMBDA NIL                                                  (* kbr%: "19-Feb-85 15:14")
    (PROG NIL                                                  (* "Character" 256 is used to handle IMAGEOBJs.
                                                               *)
          (SETQ EMACS.SYNTAX (ARRAY 257 'WORD 0 0))
          (FOR I FROM 0 TO 256 DO (SETA EMACS.SYNTAX I (LOGOR EMACS.NONCR EMACS.NONWS EMACS.NONSD EMACS.ALPHA)
                                                              ))
          (FOR I IN (CHARCODE (TAB LF SP)) DO (SETA EMACS.SYNTAX I (LOGOR EMACS.WS EMACS.NONCR EMACS.NONSD
                                                                          EMACS.SPACE)))
          (SETA EMACS.SYNTAX (CHARCODE CR)
                (LOGOR EMACS.WS EMACS.CR EMACS.NONSD))
          (FOR I IN '(39 44 64 96) DO (SETA EMACS.SYNTAX I (LOGOR EMACS.NONWS EMACS.NONCR EMACS.NONSD EMACS.BQ
                                                                  EMACS.ALPHA)))
          (SETQ EMACS.DELIMS NIL)
          (SETQ EMACS.SDELIMS NIL)
          (SETQ EMACS.LDELIMS NIL)
          (SETQ EMACS.RDELIMS NIL)
          (EMACS.DELIMS (CHARCODE "(")
                (CHARCODE ")"))
          (EMACS.DELIMS (CHARCODE "[")
                (CHARCODE "]"))
          (EMACS.DELIMS (CHARCODE "{")
                (CHARCODE "}"))
          (EMACS.DELIMS 34 34])

)

(DECLARE%: DONTEVAL@LOAD DOCOPY

(EMACS.INIT)

(MOVD? %'TEDIT.SELECT.LINE.SCANNER %' OLD.TEDIT.SELECT.LINE.SCANNER)

(MOVD %'NEW.TEDIT.SELECT.LINE.SCANNER %' TEDIT.SELECT.LINE.SCANNER)

(MOVD %'EMACS %'TEDIT)
)

(PUTPROPS EMACS COPYRIGHT ("Xerox Corporation" 1985 1986 2021))
```

FUNCTION INDEX

VARIABLE INDEX

PROPERTY INDEX

RECORD INDEX