*File created:* 16-May-90 12:27:02 **{DSK}<usr>local>lde>lispcore>sources>CLISP.;2**

*changes to:* (VARS CLISPCOMS)

*previous date:* 26-Nov-86 12:32:58 {DSK}<usr>local>lde>lispcore>sources>CLISP.;1

*Read Table:* INTERLISP

*Package:* INTERLISP

*Format:* XCCS

```
(RPAQQ CLISPCOMS
       [(COMS                                                  ; DWIM stuff
              [INITVARS (NOFIXFNSLST0)
                     (NOFIXVARSLST0)
                     (NOSPELLFLG)
                     (LPARKEY 9)
                     (RPARKEY 0)
                     (WTFIXCHCONLST '(NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL
                                          NIL NIL))
                     (WTFIXCHCONLST1 '(NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL
                                           NIL NIL]
              (USERMACROS FIX8 FIX9)
              (ADDVARS (DWIMUSERFORMS)
                     (LAMBDASPLST LAMBDA NLAMBDA)
                     (OKREEVALST AND OR PROGN SAVESETQ CAR CDR ADD1 SUB1 CONS LIST EQ EQUAL PRINT PRIN1 APPEND
                            NEQ NOT NULL)
                     (NOFIXFNSLST)
                     (NOFIXVARSLST)
                     (GLOBALVARS)
                     (LOCALVARS)
                     (SPECVARS)
                     (NLAMA)
                     (NLAML)
                     (LAMA)
                     (LAMS))
              (P (MOVD? 'NILL 'FREEVARS))
              (PROP FILEDEF BREAKDOWN CALLS CLISPRECORD SETUPHASHARRAY MAKEMATCH)
              (VARS (DWIMIFYFLG 'EVAL)
                    (COMPILEUSERFN 'COMPILEUSERFN)
                    (CLISPTRANFLG 'CLISP% )
                    (DWIMESSGAG))
              (INITVARS (DWIMCHECK#ARGSFLG T)
                     (DWIMCHECKPROGLABELSFLG T)
                     (%#CLISPARRAY 250)
                     (RECORDHASHFLG T)
                     (CLISPRETRANFLG))
              (ADDVARS (DWIMEQUIVLST))
              (USERMACROS DW !DW CLISP%: NOCLISP PPT))
        (COMS (* CLISP props)
              (PROP CLISPTYPE %')
              [E (SETQQ CLISPCHARS
                        (^ * / + - = _ %: %' ~ +- ~= < > @ ! ← ↑))
                (CLISPDEC '(STANDARD MIXED]
              [VARS (CLISPFLG T)
                    (CLISPCHARS '(^ * / + - = _ %: %' ~ +- ~= < > @ ! ← ↑]
              (INITVARS (CLISPHELPFLG T)
                     (TREATASCLISPFLG)
                     (CLISPINFIXSPLST)
                     (CLISPCHARRAY (MAKEBITTABLE CLISPCHARS))
                     [LEFT.ARROWS.BITTABLE (MAKEBITTABLE '(_ ←]
                     (LEFT.ARROW '_)
                     (CLISPISWORDSPLST)
                     (CLISPLASTSUB (CONS))
                     (CHECKCARATOMFLG)
                     (CLISPARITHOPLST '(+ - * / +- LT GT lt gt GEQ LEQ GE LE geq leq ge le))
                     (CLISPARITHCLASSLST '(INTEGER FIXED MIXED FLOATING))
                     (DWIMINMACROSFLG NIL))
              (IFPROP (CLISPTYPE LISPFN UNARYOP CLISPCLASS CLISPCLASSDEF CLISPNEG CLISPBRACKET)
                     ↑ ^ * / + - = _ ← %: %' ~ +- ~= < > @ !)
              (VARS DECLWORDS)
              (IFPROP (CLISPTYPE LISPFN UNARYOP CLISPINFIX CLISPCLASS CLISPCLASSDEF CLISPNEG BROADSCOPE)
                     *
                     (PROGN DECLWORDS))
              (IFPROP (CLISPTYPE LISPFN UNARYOP CLISPINFIX CLISPCLASS CLISPCLASSDEF CLISPNEG BROADSCOPE)
                     LT lt GT gt LE le GE ge LEQ leq GEQ geq EQ NEQ EQP EQUAL EQUALS NOT AND OR and or NOR nor
                     MEMBER SETQ IPLUS IMINUS IDIFFERENCE ITIMES IQUOTIENT ILESSP IGREATERP FPLUS FMINUS
                     FDIFFERENCE FTIMES FQUOTIENT FGTP PLUS MINUS DIFFERENCE TIMES QUOTIENT LESSP GREATERP EXPT
                     -> =>)
              (PROP SETFN ELT SETA)
              (OPTIMIZERS CLISP% ))
        (PROP CLISPWORD AND OR and or ! !! CLISP clisp MATCH match)
        (COMS (* IF)
              (VARS CLISPIFWORDSPLST)
              (INITVARS (CLISPIFTRANFLG T))
              (PROP CLISPWORD IF THEN ELSE ELSEIF if then else elseif))
        (COMS (* I.S.OPR)
              (VARS (CLISPI.S.GAG))
              (PROP CLISPWORD * INITISOPRS)
              (IFPROP I.S.OPR * (PROGN INITISOPRS))
              [ADDVARS * (LIST (CONS 'I.S.OPRLST INITISOPRS)
                               (CONS 'CLISPFORWORDSPLST (SUBSET INITISOPRS 'U-CASEP]
              [VARS (CLISPDUMMYFORVARS '($$TEM0 $$TEM1 $$TEM2 $$TEM3 $$TEM4 $$TEM5 $$TEM6]
              (ADDVARS * (LIST (CONS 'SYSLOCALVARS CLISPDUMMYFORVARS)
                               (CONS 'INVISIBLEVARS CLISPDUMMYFORVARS)))
              (ADDVARS (SYSLOCALVARS $$VAL $$TEM $$LST1 $$LST2 $$LST3 $$LST4 $$LST5 $$LST6 $$END $$EXTREME)
                     (INVISIBLEVARS $$VAL $$END $$TEM $$LST1 $$LST2 $$LST3 $$LST4 $$LST5 $$LST6 $$EXTREME))
              (FILEPKGCOMS I.S.OPRS)
              (FNS DUMPI.S.OPRS GETDEF.I.S.OPR))
        (COMS (* forDuration)
              (ADDVARS (DURATIONCLISPWORDS (TIMERUNITS timerUnits timerunits)
                              (USINGBOX usingBox usingbox)
                              (USINGTIMER usingTimer usingtimer)
                              (FORDURATION forDuration forduration DURING during)
                              (RESOURCENAME resourceName resourcename)
                              (UNTILDATE untilDate untildate)))
              (IFPROP (CLISPWORD \DURATIONTRAN)
                     *
                     (APPLY 'APPEND DURATIONCLISPWORDS))
              (RESOURCES \ForDurationOfBox))
        (COMS
              ;; Currently there are four possible entries for the INFO property: EVAL, BINDS, LABELS, PROGN, or a list containg any or all of
              ;; these.

              ;; EVAL is used to indicate that an nlambda evaluates its arguments.  EVAL affects DWIMIFY and CLISPIFY: neither will touch an
              ;; nlambda that does not have this property.

              ;; BINDS tells clispify and dwimify that CADR of the form is a list of variables being bound, a la prog.

              ;; PROGN says that only the last top level expression is being used for value. This affects the way OR's and AND's are clispified, for
              ;; example.

              ;; Finally, LABELS indicates that top level atoms in this expression are not being evaluated. This tells clispify not to create atoms out
              ;; of lists at the top level.  LABELS also implies that none of the top level expressions are being used for value.

              ;; For example, FOR has info property just BINDS, (EVAL is unnecssary since FOR is not a function and its dwimifying and clispifying
              ;; affected by its clispword property), whereas PROG has (BINDS EVAL LABELS), and LAMBDA has (EVAL BINDS PROGN)

              (PROP INFO PROG PROG* RESETVARS RESETBUFS RESETLST ADV-PROG ADV-SETQ AND ARG COND ERSETQ NLSETQ OR
                    PROG1 PROG2 PROGN RESETFORM RESETSAVE RESETVAR RPAQ RPTQ FRPTQ SAVESETQ SETN SETQ UNDONLSETQ
                    XNLSETQ SETARG LET LET* RETURN))
        (PROP FILETYPE CLISP)
        (DECLARE%: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS (ADDVARS (NLAMA DUMPI.S.OPRS)
                                                                              (NLAML)
                                                                              (LAMA])
```

```
;; DWIM stuff

(RPAQ? NOFIXFNSLST0 )

(RPAQ? NOFIXVARSLST0 )

(RPAQ? NOSPELLFLG )

(RPAQ? LPARKEY 9)

(RPAQ? RPARKEY 0)

(RPAQ? WTFIXCHCONLST '(NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL))

(RPAQ? WTFIXCHCONLST1 '(NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL NIL))

(ADDTOVAR EDITMACROS
          (FIX9 (X N)
                (BIND (E (SETQ %#1 (EDITFPAT 'X))
                         T)
                      (IF (NOT (ATOM (%##)))
                          (1))
                      (COMS (SPLIT89 RPARKEY N))
                      (I F RPARKEY T)
                      (E [SETQ %#2 (ADD1 (LENGTH (CAR L]
                         T)
                      !0 MARK (LPQ [IF (OR (NULL %#1)
                                           (NOT (EDIT4E %#1 (%## 1]
                                   UP
                                   (E (SETQ %#3 (LENGTH (CAR L)))
                                      T)
                                   (I RI 1 (MINUS %#2))
```

```
                                    (E (SETQ %#2 %#3)
                                       T)
                                    1 !0)
                      __
                      (DELETE NX)))
          (FIX9 NIL (FIX9))
          (FIX8 (X N)
                (BIND (E (SETQ %#1 (EDITFPAT 'X))
                         T)
                      (IF (LISTP (%##))
                          (1))
                      (COMS (SPLIT89 LPARKEY N))
                      (I F LPARKEY T)
                      (1)
                      (LI 1)
                      (IF (TAILP (CAR L)
                                 (CADR L))
                          (!0)
                          NIL)
                      (LPQ [IF (OR (NULL %#1)
                                   (NOT (EDIT4E %#1 (%## 1]
                           UP
                           (RO 1)
                           !0)))
          (FIX8 NIL (FIX8)))

(ADDTOVAR DWIMUSERFORMS )

(ADDTOVAR LAMBDASPLST LAMBDA NLAMBDA)

(ADDTOVAR OKREEVALST AND OR PROGN SAVESETQ CAR CDR ADD1 SUB1 CONS LIST EQ EQUAL PRINT PRIN1 APPEND NEQ NOT NULL
)

(ADDTOVAR NOFIXFNSLST )

(ADDTOVAR NOFIXVARSLST )

(ADDTOVAR GLOBALVARS )

(ADDTOVAR LOCALVARS )

(ADDTOVAR SPECVARS )

(ADDTOVAR NLAMA )

(ADDTOVAR NLAML )

(ADDTOVAR LAMA )

(ADDTOVAR LAMS )

(MOVD? 'NILL 'FREEVARS)

(PUTPROPS BREAKDOWN FILEDEF BRKDWN)

(PUTPROPS CALLS FILEDEF MSANALYZE)

(PUTPROPS CLISPRECORD FILEDEF RECORD)

(PUTPROPS SETUPHASHARRAY FILEDEF (RECORD SETUPHASHARRAY))

(PUTPROPS MAKEMATCH FILEDEF MATCH)

(RPAQQ DWIMIFYFLG EVAL)

(RPAQQ COMPILEUSERFN COMPILEUSERFN)

(RPAQQ CLISPTRANFLG CLISP% )

(RPAQQ DWIMESSGAG NIL)

(RPAQ? DWIMCHECK#ARGSFLG T)

(RPAQ? DWIMCHECKPROGLABELSFLG T)

(RPAQ? %#CLISPARRAY 250)

(RPAQ? RECORDHASHFLG T)

(RPAQ? CLISPRETRANFLG )

(ADDTOVAR DWIMEQUIVLST )

(ADDTOVAR EDITMACROS
          (DW NIL (BIND (E (PROGN (SETQ %#1 (%##))
                                  (AND (CDR L)
                                       (%## !0 (E (SETQ %#2 L)
                                                 T)))
                                  (AND [SETQ %#3 (DWIMIFY %#1 T (OR %#2 '(NIL]
                                       EDITCHANGES
                                       (RPLACA (CDR EDITCHANGES)
                                              T)))
                          T)
                        (IF (NLISTP %#1)
                            ((I %: %#3)
                             (IF (LISTP %#3)
                                 (1)
                                 NIL))
                           NIL)))
          (PPT NIL (RESETVAR PRETTYTRANFLG T PP))
          (!DW NIL (RESETVAR CLISPRETRANFLG T DW))
          (NOCLISP NIL (NOCLISP TTY%:))
          (NOCLISP COMS (RESETVAR CLISPTRANFLG NIL . COMS))
          (CLISP%: NIL (BIND (E (COND ((SETQ %#1 (AND CLISPARRAY (GETHASH (%##)
                                                                         CLISPARRAY)))
                                       (SETQQ COM CLISP%:)
                                       (EDITE %#1))
                                      (T (PRIN1 '"not translated.
                                                 " T)))
                               T))))

(ADDTOVAR EDITCOMSA PPT DW !DW CLISP%:)

          (* * CLISP props)

(PUTPROPS %' CLISPTYPE 15)

(RPAQQ CLISPFLG T)

(RPAQQ CLISPCHARS (^ * / + - = _ %: %' ~ +- ~= < > @ ! ← ↑))

(RPAQ? CLISPHELPFLG T)

(RPAQ? TREATASCLISPFLG )

(RPAQ? CLISPINFIXSPLST )

(RPAQ? CLISPCHARRAY (MAKEBITTABLE CLISPCHARS))

(RPAQ? LEFT.ARROWS.BITTABLE (MAKEBITTABLE '(_ ←)))

(RPAQ? LEFT.ARROW '_)

(RPAQ? CLISPISWORDSPLST )

(RPAQ? CLISPLASTSUB (CONS))

(RPAQ? CHECKCARATOMFLG )

(RPAQ? CLISPARITHOPLST '(+ - * / +- LT GT lt gt GEQ LEQ GE LE geq leq ge le))

(RPAQ? CLISPARITHCLASSLST '(INTEGER FIXED MIXED FLOATING))

(RPAQ? DWIMINMACROSFLG NIL)

(PUTPROPS ↑ CLISPTYPE 6)

(PUTPROPS ^ CLISPTYPE 6)

(PUTPROPS * CLISPTYPE 4)

(PUTPROPS / CLISPTYPE 4)

(PUTPROPS + CLISPTYPE 2)

(PUTPROPS - CLISPTYPE 7)

(PUTPROPS = CLISPTYPE -20)

(PUTPROPS _ CLISPTYPE (8 . -12))

(PUTPROPS ← CLISPTYPE (8 . -12))

(PUTPROPS %: CLISPTYPE (14 . 13))

(PUTPROPS %' CLISPTYPE 15)

(PUTPROPS ~ CLISPTYPE 7)

(PUTPROPS +- CLISPTYPE 2)

(PUTPROPS < CLISPTYPE BRACKET)
```

```
(PUTPROPS > CLISPTYPE BRACKET)

(PUTPROPS ↑ LISPFN EXPT)

(PUTPROPS ^ LISPFN EXPT)

(PUTPROPS * LISPFN TIMES)

(PUTPROPS / LISPFN QUOTIENT)

(PUTPROPS + LISPFN PLUS)

(PUTPROPS - LISPFN MINUS)

(PUTPROPS = LISPFN EQ)

(PUTPROPS _ LISPFN SETQ)

(PUTPROPS ← LISPFN SETQ)

(PUTPROPS %' LISPFN QUOTE)

(PUTPROPS ~ LISPFN NOT)

(PUTPROPS +- LISPFN DIFFERENCE)

(PUTPROPS - UNARYOP T)

(PUTPROPS %' UNARYOP T)

(PUTPROPS ~ UNARYOP T)

(PUTPROPS < UNARYOP T)

(PUTPROPS > UNARYOP T)

(PUTPROPS * CLISPCLASS *)

(PUTPROPS / CLISPCLASS /)

(PUTPROPS + CLISPCLASS +)

(PUTPROPS - CLISPCLASS -)

(PUTPROPS +- CLISPCLASS +-)

(PUTPROPS * CLISPCLASSDEF (ARITH ITIMES FTIMES TIMES))

(PUTPROPS / CLISPCLASSDEF (ARITH IQUOTIENT FQUOTIENT QUOTIENT))

(PUTPROPS + CLISPCLASSDEF (ARITH IPLUS FPLUS PLUS))

(PUTPROPS - CLISPCLASSDEF (ARITH IMINUS FMINUS MINUS))

(PUTPROPS +- CLISPCLASSDEF (ARITH IDIFFERENCE FDIFFERENCE DIFFERENCE))

(PUTPROPS = CLISPNEG ~=)

(PUTPROPS < CLISPBRACKET (< > SEPARATOR ! DWIMIFY CLISPANGLEBRACKETS CLISPIFY SHRIEKIFY))

(PUTPROPS > CLISPBRACKET (< > SEPARATOR ! DWIMIFY CLISPANGLEBRACKETS CLISPIFY SHRIEKIFY))

(RPAQQ DECLWORDS
       (FLOATING FAST FFETCHFIELD FETCHFIELD REPLACEFIELD FREPLACEFIELD /REPLACEFIELD /LISTPUT /LISTPUT1 /MAPCON
              /MAPCONC /NCONC /NCONC1 /PUT /PUTASSOC /PUTHASH /PUTPROP /RPLACA /RPLACD /RPLNODE /RPLNODE2 /SETA
              ASSOC CLISPIFY FASSOC FIXED FLAST FMEMB FNTH FRPLACA FRPLACD FRPLNODE FRPLNODE2 INTEGER LAST
              LISTPUT LISTPUT1 MAPCON MAPCONC MEMB MIXED NCONC NCONC1 NTH PUT PUTASSOC PUTHASH PUTPROP RPLACA
              RPLACD RPLNODE RPLNODE2 SETA STANDARD UNDOABLE))

(PUTPROPS FMEMB CLISPTYPE -20)

(PUTPROPS MEMB CLISPTYPE -20)

(PUTPROPS FETCHFIELD LISPFN FETCHFIELD)

(PUTPROPS REPLACEFIELD LISPFN REPLACEFIELD)

(PUTPROPS FREPLACEFIELD LISPFN FREPLACEFIELD)

(PUTPROPS ASSOC LISPFN ASSOC)

(PUTPROPS LAST LISPFN LAST)

(PUTPROPS LISTPUT LISPFN LISTPUT)

(PUTPROPS LISTPUT1 LISPFN LISTPUT1)
```

```
(PUTPROPS MAPCON LISPFN MAPCON)

(PUTPROPS MAPCONC LISPFN MAPCONC)

(PUTPROPS MEMB LISPFN MEMB)

(PUTPROPS NCONC LISPFN NCONC)

(PUTPROPS NCONC1 LISPFN NCONC1)

(PUTPROPS NTH LISPFN NTH)

(PUTPROPS PUT LISPFN PUT)

(PUTPROPS PUTASSOC LISPFN PUTASSOC)

(PUTPROPS PUTHASH LISPFN PUTHASH)

(PUTPROPS PUTPROP LISPFN PUTPROP)

(PUTPROPS RPLACA LISPFN RPLACA)

(PUTPROPS RPLACD LISPFN RPLACD)

(PUTPROPS RPLNODE LISPFN RPLNODE)

(PUTPROPS RPLNODE2 LISPFN RPLNODE2)

(PUTPROPS SETA LISPFN SETA)

(PUTPROPS FLOATING CLISPCLASS (ARITH . 2))

(PUTPROPS FAST CLISPCLASS (ACCESS . 3))

(PUTPROPS FFETCHFIELD CLISPCLASS FETCHFIELD)

(PUTPROPS FETCHFIELD CLISPCLASS FETCHFIELD)

(PUTPROPS REPLACEFIELD CLISPCLASS REPLACEFIELD)

(PUTPROPS FREPLACEFIELD CLISPCLASS REPLACEFIELD)

(PUTPROPS /REPLACEFIELD CLISPCLASS REPLACEFIELD)

(PUTPROPS /LISTPUT CLISPCLASS LISTPUT)

(PUTPROPS /MAPCON CLISPCLASS MAPCON)

(PUTPROPS /MAPCONC CLISPCLASS MAPCONC)

(PUTPROPS /NCONC CLISPCLASS NCONC)

(PUTPROPS /NCONC1 CLISPCLASS NCONC1)

(PUTPROPS /PUT CLISPCLASS PUT)

(PUTPROPS /PUTASSOC CLISPCLASS PUTASSOC)

(PUTPROPS /PUTHASH CLISPCLASS PUTHASH)

(PUTPROPS /PUTPROP CLISPCLASS PUTPROP)

(PUTPROPS /RPLACA CLISPCLASS RPLACA)

(PUTPROPS /RPLACD CLISPCLASS RPLACD)

(PUTPROPS /RPLNODE CLISPCLASS RPLNODE)

(PUTPROPS /RPLNODE2 CLISPCLASS RPLNODE2)

(PUTPROPS /SETA CLISPCLASS SETA)

(PUTPROPS ASSOC CLISPCLASS ASSOC)

(PUTPROPS FASSOC CLISPCLASS ASSOC)

(PUTPROPS FIXED CLISPCLASS (ARITH . 1))

(PUTPROPS FLAST CLISPCLASS LAST)

(PUTPROPS FMEMB CLISPCLASS MEMB)

(PUTPROPS FNTH CLISPCLASS NTH)

(PUTPROPS FRPLACA CLISPCLASS RPLACA)

(PUTPROPS FRPLACD CLISPCLASS RPLACD)
```

```
(PUTPROPS FRPLNODE CLISPCLASS RPLNODE)

(PUTPROPS FRPLNODE2 CLISPCLASS RPLNODE2)

(PUTPROPS INTEGER CLISPCLASS (ARITH . 1))

(PUTPROPS LAST CLISPCLASS LAST)

(PUTPROPS LISTPUT CLISPCLASS LISTPUT)

(PUTPROPS LISTPUT1 CLISPCLASS LISTPUT1)

(PUTPROPS MAPCON CLISPCLASS MAPCON)

(PUTPROPS MAPCONC CLISPCLASS MAPCONC)

(PUTPROPS MEMB CLISPCLASS MEMB)

(PUTPROPS MIXED CLISPCLASS (ARITH . 3))

(PUTPROPS NCONC CLISPCLASS NCONC)

(PUTPROPS NCONC1 CLISPCLASS NCONC1)

(PUTPROPS NTH CLISPCLASS NTH)

(PUTPROPS PUT CLISPCLASS PUT)

(PUTPROPS PUTASSOC CLISPCLASS PUTASSOC)

(PUTPROPS PUTHASH CLISPCLASS PUTHASH)

(PUTPROPS PUTPROP CLISPCLASS PUTPROP)

(PUTPROPS RPLACA CLISPCLASS RPLACA)

(PUTPROPS RPLACD CLISPCLASS RPLACD)

(PUTPROPS RPLNODE CLISPCLASS RPLNODE)

(PUTPROPS RPLNODE2 CLISPCLASS RPLNODE2)

(PUTPROPS SETA CLISPCLASS SETA)

(PUTPROPS STANDARD CLISPCLASS (ACCESS . 1))

(PUTPROPS UNDOABLE CLISPCLASS (ACCESS . 2))

(PUTPROPS FETCHFIELD CLISPCLASSDEF (ACCESS FETCHFIELD NIL FFETCHFIELD))

(PUTPROPS REPLACEFIELD CLISPCLASSDEF (ACCESS REPLACEFIELD /REPLACEFIELD FREPLACEFIELD))

(PUTPROPS ASSOC CLISPCLASSDEF (ACCESS ASSOC NIL FASSOC))

(PUTPROPS LAST CLISPCLASSDEF (ACCESS LAST NIL FLAST))

(PUTPROPS LISTPUT CLISPCLASSDEF (ACCESS LISTPUT /LISTPUT))

(PUTPROPS LISTPUT1 CLISPCLASSDEF (ACCESS LISTPUT1 /LISTPUT1))

(PUTPROPS MAPCON CLISPCLASSDEF (ACCESS MAPCON /MAPCON))

(PUTPROPS MAPCONC CLISPCLASSDEF (ACCESS MAPCONC /MAPCONC))

(PUTPROPS MEMB CLISPCLASSDEF (ACCESS MEMB NIL FMEMB))

(PUTPROPS NCONC CLISPCLASSDEF (ACCESS NCONC /NCONC))

(PUTPROPS NCONC1 CLISPCLASSDEF (ACCESS NCONC1 /NCONC1))

(PUTPROPS NTH CLISPCLASSDEF (ACCESS NTH NIL FNTH))

(PUTPROPS PUT CLISPCLASSDEF (ACCESS PUT /PUT))

(PUTPROPS PUTASSOC CLISPCLASSDEF (ACCESS PUTASSOC /PUTASSOC))

(PUTPROPS PUTHASH CLISPCLASSDEF (ACCESS PUTHASH /PUTHASH))

(PUTPROPS PUTPROP CLISPCLASSDEF (ACCESS PUTPROP /PUTPROP))

(PUTPROPS RPLACA CLISPCLASSDEF (ACCESS RPLACA /RPLACA FRPLACA))

(PUTPROPS RPLACD CLISPCLASSDEF (ACCESS RPLACD /RPLACD FRPLACD))

(PUTPROPS RPLNODE CLISPCLASSDEF (ACCESS RPLNODE /RPLNODE FRPLNODE))
```

```
(PUTPROPS RPLNODE2 CLISPCLASSDEF (ACCESS RPLNODE2 /RPLNODE2 FRPLNODE2))

(PUTPROPS SETA CLISPCLASSDEF (ACCESS SETA /SETA))

(PUTPROPS FMEMB CLISPNEG ~FMEMB)

(PUTPROPS MEMB CLISPNEG ~MEMB)

(PUTPROPS FMEMB BROADSCOPE T)

(PUTPROPS MEMB BROADSCOPE T)

(PUTPROPS LT CLISPTYPE -20)

(PUTPROPS lt CLISPTYPE -20)

(PUTPROPS GT CLISPTYPE -20)

(PUTPROPS gt CLISPTYPE -20)

(PUTPROPS LE CLISPTYPE -20)

(PUTPROPS le CLISPTYPE -20)

(PUTPROPS GE CLISPTYPE -20)

(PUTPROPS ge CLISPTYPE -20)

(PUTPROPS LEQ CLISPTYPE -20)

(PUTPROPS leq CLISPTYPE -20)

(PUTPROPS GEQ CLISPTYPE -20)

(PUTPROPS geq CLISPTYPE -20)

(PUTPROPS EQ CLISPTYPE -20)

(PUTPROPS NEQ CLISPTYPE -20)

(PUTPROPS EQP CLISPTYPE -20)

(PUTPROPS EQUAL CLISPTYPE -20)

(PUTPROPS EQUALS CLISPTYPE -20)

(PUTPROPS AND CLISPTYPE -25)

(PUTPROPS OR CLISPTYPE -26)

(PUTPROPS and CLISPTYPE -25)

(PUTPROPS or CLISPTYPE -26)

(PUTPROPS NOR CLISPTYPE -25)

(PUTPROPS nor CLISPTYPE -25)

(PUTPROPS MEMBER CLISPTYPE -20)

(PUTPROPS ILESSP CLISPTYPE -20)

(PUTPROPS IGREATERP CLISPTYPE -20)

(PUTPROPS FGTP CLISPTYPE -20)

(PUTPROPS MINUS CLISPTYPE 8)

(PUTPROPS LESSP CLISPTYPE -20)

(PUTPROPS GREATERP CLISPTYPE -20)

(PUTPROPS -> CLISPTYPE 7)

(PUTPROPS => CLISPTYPE 7)

(PUTPROPS LT LISPFN LESSP)

(PUTPROPS lt LISPFN LESSP)

(PUTPROPS GT LISPFN GREATERP)

(PUTPROPS gt LISPFN GREATERP)

(PUTPROPS LE LISPFN LEQ)

(PUTPROPS le LISPFN LEQ)
```

```
(PUTPROPS GE LISPFN GEQ)

(PUTPROPS ge LISPFN GEQ)

(PUTPROPS LEQ LISPFN LEQ)

(PUTPROPS leq LISPFN LEQ)

(PUTPROPS GEQ LISPFN GEQ)

(PUTPROPS geq LISPFN GEQ)

(PUTPROPS EQUALS LISPFN EQUAL)

(PUTPROPS AND LISPFN AND)

(PUTPROPS OR LISPFN OR)

(PUTPROPS and LISPFN AND)

(PUTPROPS or LISPFN OR)

(PUTPROPS NOR LISPFN AND)

(PUTPROPS nor LISPFN AND)

(PUTPROPS NOT UNARYOP T)

(PUTPROPS MINUS UNARYOP T)

(PUTPROPS LEQ CLISPINFIX le)

(PUTPROPS GEQ CLISPINFIX ge)

(PUTPROPS EQ CLISPINFIX =)

(PUTPROPS NOT CLISPINFIX ~)

(PUTPROPS AND CLISPINFIX and)

(PUTPROPS OR CLISPINFIX or)

(PUTPROPS SETQ CLISPINFIX _)

(PUTPROPS IPLUS CLISPINFIX +)

(PUTPROPS IMINUS CLISPINFIX -)

(PUTPROPS IDIFFERENCE CLISPINFIX +-)

(PUTPROPS ITIMES CLISPINFIX *)

(PUTPROPS IQUOTIENT CLISPINFIX /)

(PUTPROPS ILESSP CLISPINFIX lt)

(PUTPROPS IGREATERP CLISPINFIX gt)

(PUTPROPS PLUS CLISPINFIX +)

(PUTPROPS MINUS CLISPINFIX -)

(PUTPROPS DIFFERENCE CLISPINFIX +-)

(PUTPROPS TIMES CLISPINFIX *)

(PUTPROPS QUOTIENT CLISPINFIX /)

(PUTPROPS LESSP CLISPINFIX lt)

(PUTPROPS GREATERP CLISPINFIX gt)

(PUTPROPS EXPT CLISPINFIX ^)

(PUTPROPS LT CLISPCLASS LT)

(PUTPROPS lt CLISPCLASS LT)

(PUTPROPS GT CLISPCLASS GT)

(PUTPROPS gt CLISPCLASS GT)

(PUTPROPS LE CLISPCLASS LEQ)

(PUTPROPS le CLISPCLASS LEQ)
```

```
(PUTPROPS GE CLISPCLASS GEQ)

(PUTPROPS ge CLISPCLASS GEQ)

(PUTPROPS LEQ CLISPCLASS LEQ)

(PUTPROPS leq CLISPCLASS LEQ)

(PUTPROPS GEQ CLISPCLASS GEQ)

(PUTPROPS geq CLISPCLASS GEQ)

(PUTPROPS IPLUS CLISPCLASS +)

(PUTPROPS IMINUS CLISPCLASS -)

(PUTPROPS IDIFFERENCE CLISPCLASS +-)

(PUTPROPS ITIMES CLISPCLASS *)

(PUTPROPS IQUOTIENT CLISPCLASS /)

(PUTPROPS ILESSP CLISPCLASS LT)

(PUTPROPS IGREATERP CLISPCLASS GT)

(PUTPROPS FPLUS CLISPCLASS +)

(PUTPROPS FMINUS CLISPCLASS -)

(PUTPROPS FDIFFERENCE CLISPCLASS +-)

(PUTPROPS FTIMES CLISPCLASS *)

(PUTPROPS FQUOTIENT CLISPCLASS /)

(PUTPROPS FGTP CLISPCLASS GT)

(PUTPROPS PLUS CLISPCLASS +)

(PUTPROPS MINUS CLISPCLASS -)

(PUTPROPS DIFFERENCE CLISPCLASS +-)

(PUTPROPS TIMES CLISPCLASS *)

(PUTPROPS QUOTIENT CLISPCLASS /)

(PUTPROPS LESSP CLISPCLASS LT)

(PUTPROPS GREATERP CLISPCLASS GT)

(PUTPROPS LT CLISPCLASSDEF (ARITH ILESSP LESSP LESSP))

(PUTPROPS GT CLISPCLASSDEF (ARITH IGREATERP FGTP GREATERP))

(PUTPROPS LE CLISPCLASSDEF (ARITH ILEQ LEQ LEQ))

(PUTPROPS GE CLISPCLASSDEF (ARITH IGEQ GEQ GEQ))

(PUTPROPS LEQ CLISPCLASSDEF (ARITH ILEQ LEQ LEQ))

(PUTPROPS GEQ CLISPCLASSDEF (ARITH IGEQ GEQ GEQ))

(PUTPROPS LT CLISPNEG GEQ)

(PUTPROPS GT CLISPNEG LEQ)

(PUTPROPS EQUALS CLISPNEG ~EQUAL)

(PUTPROPS MEMBER CLISPNEG ~MEMBER)

(PUTPROPS LT BROADSCOPE T)

(PUTPROPS lt BROADSCOPE T)

(PUTPROPS GT BROADSCOPE T)

(PUTPROPS gt BROADSCOPE T)

(PUTPROPS LE BROADSCOPE T)

(PUTPROPS le BROADSCOPE T)

(PUTPROPS GE BROADSCOPE T)

(PUTPROPS ge BROADSCOPE T)
```

```
(PUTPROPS LEQ BROADSCOPE T)

(PUTPROPS leq BROADSCOPE T)

(PUTPROPS GEQ BROADSCOPE T)

(PUTPROPS geq BROADSCOPE T)

(PUTPROPS EQ BROADSCOPE T)

(PUTPROPS NEQ BROADSCOPE T)

(PUTPROPS EQP BROADSCOPE T)

(PUTPROPS EQUAL BROADSCOPE T)

(PUTPROPS EQUALS BROADSCOPE T)

(PUTPROPS NOT BROADSCOPE T)

(PUTPROPS AND BROADSCOPE T)

(PUTPROPS OR BROADSCOPE T)

(PUTPROPS and BROADSCOPE T)

(PUTPROPS or BROADSCOPE T)

(PUTPROPS NOR BROADSCOPE T)

(PUTPROPS nor BROADSCOPE T)

(PUTPROPS MEMBER BROADSCOPE T)

(PUTPROPS ILESSP BROADSCOPE T)

(PUTPROPS IGREATERP BROADSCOPE T)

(PUTPROPS FGTP BROADSCOPE T)

(PUTPROPS LESSP BROADSCOPE T)

(PUTPROPS GREATERP BROADSCOPE T)

(PUTPROPS ELT SETFN SETA)

(PUTPROPS SETA SETFN (ELT))

(DEFOPTIMIZER CLISP% (X &REST Y)
                      X)

(PUTPROPS AND CLISPWORD T)

(PUTPROPS OR CLISPWORD T)

(PUTPROPS and CLISPWORD T)

(PUTPROPS or CLISPWORD T)

(PUTPROPS ! CLISPWORD T)

(PUTPROPS !! CLISPWORD T)

(PUTPROPS CLISP CLISPWORD (PREFIXFN . clisp))

(PUTPROPS clisp CLISPWORD (PREFIXFN . clisp))

(PUTPROPS MATCH CLISPWORD (MATCHWORD . match))

(PUTPROPS match CLISPWORD (MATCHWORD . match))

          (* * IF)

(RPAQQ CLISPIFWORDSPLST (THEN ELSE ELSEIF IF))

(RPAQ? CLISPIFTRANFLG T)

(PUTPROPS IF CLISPWORD (IFWORD . if))

(PUTPROPS THEN CLISPWORD (IFWORD . then))

(PUTPROPS ELSE CLISPWORD (IFWORD . else))

(PUTPROPS ELSEIF CLISPWORD (IFWORD . elseif))
```

```
(PUTPROPS if CLISPWORD (IFWORD . if))

(PUTPROPS then CLISPWORD (IFWORD . then))

(PUTPROPS else CLISPWORD (IFWORD . else))

(PUTPROPS elseif CLISPWORD (IFWORD . elseif))



          (* * I.S.OPR)

(RPAQQ CLISPI.S.GAG NIL)

(RPAQQ INITISOPRS
       (ALWAYS AS BIND BY COLLECT COUNT DECLARE DECLARE%: DO EACHTIME FCOLLECT FINALLY FIND FIRST FOR FROM IN
              INSIDE ISTHERE JOIN LARGEST NEVER OLD ON ORIGINAL REPEATUNTIL REPEATWHILE SMALLEST SUCHTHAT SUM
              THEREIS THRU TO UNLESS UNTIL WHEN WHERE WHILE always as bind by collect count declare declare%: do
              eachtime fcollect finally find first for from in inside isthere join largest never old on original
              repeatuntil repeatwhile smallest suchthat sum thereis thru to unless until when where while))

(PUTPROPS ALWAYS CLISPWORD (FORWORD . always))

(PUTPROPS AS CLISPWORD (FORWORD . as))

(PUTPROPS BIND CLISPWORD (FORWORD . bind))

(PUTPROPS BY CLISPWORD (FORWORD . by))

(PUTPROPS COLLECT CLISPWORD (FORWORD . collect))

(PUTPROPS COUNT CLISPWORD (FORWORD . count))

(PUTPROPS DECLARE CLISPWORD (FORWORD . declare))

(PUTPROPS DECLARE%: CLISPWORD (FORWORD declare%: DECLARE))

(PUTPROPS DO CLISPWORD (FORWORD . do))

(PUTPROPS EACHTIME CLISPWORD (FORWORD . eachtime))

(PUTPROPS FCOLLECT CLISPWORD (FORWORD . fcollect))

(PUTPROPS FINALLY CLISPWORD (FORWORD . finally))

(PUTPROPS FIND CLISPWORD (FORWORD find FOR))

(PUTPROPS FIRST CLISPWORD (FORWORD . first))

(PUTPROPS FOR CLISPWORD (FORWORD . for))

(PUTPROPS FROM CLISPWORD (FORWORD . from))

(PUTPROPS IN CLISPWORD (FORWORD . in))

(PUTPROPS INSIDE CLISPWORD (FORWORD . inside))

(PUTPROPS ISTHERE CLISPWORD (FORWORD isthere THEREIS))

(PUTPROPS JOIN CLISPWORD (FORWORD . join))

(PUTPROPS LARGEST CLISPWORD (FORWORD . largest))

(PUTPROPS NEVER CLISPWORD (FORWORD . never))

(PUTPROPS OLD CLISPWORD (FORWORD . old))

(PUTPROPS ON CLISPWORD (FORWORD . on))

(PUTPROPS ORIGINAL CLISPWORD (FORWORD . original))

(PUTPROPS REPEATUNTIL CLISPWORD (FORWORD . repeatuntil))

(PUTPROPS REPEATWHILE CLISPWORD (FORWORD . repeatwhile))

(PUTPROPS SMALLEST CLISPWORD (FORWORD . smallest))

(PUTPROPS SUCHTHAT CLISPWORD (FORWORD suchthat THEREIS))

(PUTPROPS SUM CLISPWORD (FORWORD . sum))

(PUTPROPS THEREIS CLISPWORD (FORWORD . thereis))

(PUTPROPS THRU CLISPWORD (FORWORD thru TO))

(PUTPROPS TO CLISPWORD (FORWORD . to))
```

```
(PUTPROPS UNLESS CLISPWORD (FORWORD . unless))

(PUTPROPS UNTIL CLISPWORD (FORWORD . until))

(PUTPROPS WHEN CLISPWORD (FORWORD . when))

(PUTPROPS WHERE CLISPWORD (FORWORD where WHEN))

(PUTPROPS WHILE CLISPWORD (FORWORD . while))

(PUTPROPS always CLISPWORD (FORWORD . always))

(PUTPROPS as CLISPWORD (FORWORD . as))

(PUTPROPS bind CLISPWORD (FORWORD . bind))

(PUTPROPS by CLISPWORD (FORWORD . by))

(PUTPROPS collect CLISPWORD (FORWORD . collect))

(PUTPROPS count CLISPWORD (FORWORD . count))

(PUTPROPS declare CLISPWORD (FORWORD . declare))

(PUTPROPS declare%: CLISPWORD (FORWORD declare%: DECLARE))

(PUTPROPS do CLISPWORD (FORWORD . do))

(PUTPROPS eachtime CLISPWORD (FORWORD . eachtime))

(PUTPROPS fcollect CLISPWORD (FORWORD . fcollect))

(PUTPROPS finally CLISPWORD (FORWORD . finally))

(PUTPROPS find CLISPWORD (FORWORD find FOR))

(PUTPROPS first CLISPWORD (FORWORD . first))

(PUTPROPS for CLISPWORD (FORWORD . for))

(PUTPROPS from CLISPWORD (FORWORD . from))

(PUTPROPS in CLISPWORD (FORWORD . in))

(PUTPROPS inside CLISPWORD (FORWORD . inside))

(PUTPROPS isthere CLISPWORD (FORWORD isthere thereis))

(PUTPROPS join CLISPWORD (FORWORD . join))

(PUTPROPS largest CLISPWORD (FORWORD . largest))

(PUTPROPS never CLISPWORD (FORWORD . never))

(PUTPROPS old CLISPWORD (FORWORD . old))

(PUTPROPS on CLISPWORD (FORWORD . on))

(PUTPROPS original CLISPWORD (FORWORD . original))

(PUTPROPS repeatuntil CLISPWORD (FORWORD . repeatuntil))

(PUTPROPS repeatwhile CLISPWORD (FORWORD . repeatwhile))

(PUTPROPS smallest CLISPWORD (FORWORD . smallest))

(PUTPROPS suchthat CLISPWORD (FORWORD suchthat THEREIS))

(PUTPROPS sum CLISPWORD (FORWORD . sum))

(PUTPROPS thereis CLISPWORD (FORWORD . thereis))

(PUTPROPS thru CLISPWORD (FORWORD thru TO))

(PUTPROPS to CLISPWORD (FORWORD . to))

(PUTPROPS unless CLISPWORD (FORWORD . unless))

(PUTPROPS until CLISPWORD (FORWORD . until))

(PUTPROPS when CLISPWORD (FORWORD . when))

(PUTPROPS where CLISPWORD (FORWORD where WHEN))

(PUTPROPS while CLISPWORD (FORWORD . while))

(PUTPROPS always I.S.OPR ((COND ((NULL BODY)
                                (SETQ $$VAL NIL)
                                (GO $$OUT)))
                         BIND
                         (SETQ $$VAL T)))

(PUTPROPS collect I.S.OPR ((SETQ $$VAL (NCONC1 $$VAL BODY))))

(PUTPROPS count I.S.OPR ((AND BODY (SETQ $$VAL (ADD1 $$VAL)))
                         BIND
                         ($$VAL _ 0)))

(PUTPROPS do I.S.OPR (BODY))

(PUTPROPS fcollect I.S.OPR [(= SUBPAIR '(VAR1 VAR2)
                               (LIST (GETDUMMYVAR T)
                                     (GETDUMMYVAR T))
                               '(PROGN (SETQ VAR1 BODY)
                                       (COND [VAR2 (FRPLACD VAR2 (SETQ VAR2 (LIST VAR1]
                                             (T (SETQ $$VAL (SETQ VAR2 (LIST VAR1])

(PUTPROPS inside I.S.OPR [NIL = SUBST (GETDUMMYVAR)
                              'VAR
                              '(bind (VAR _ BODY)
                                     eachtime
                                     (COND ((NULL VAR)
                                            (GO $$OUT))
                                           ((NLISTP VAR)
                                            (SETQ I.V. VAR)
                                            (SETQ VAR NIL))
                                           (T (SETQ I.V. (CAR VAR))
                                              (SETQ VAR (CDR VAR])

(PUTPROPS join I.S.OPR ((SETQ $$VAL (NCONC $$VAL BODY))))

(PUTPROPS largest I.S.OPR [NIL = SUBST (GETDUMMYVAR)
                               '$$TEMP
                               '(BIND $$EXTREME $$TEMP DO (SETQ $$TEMP BODY)
                                      (COND ((OR (NULL $$EXTREME)
                                                 (GREATERP $$TEMP $$EXTREME))
                                             (SETQ $$EXTREME $$TEMP)
                                             (SETQ $$VAL I.V.])

(PUTPROPS never I.S.OPR ((COND (BODY (SETQ $$VAL NIL)
                                     (GO $$OUT)))
                         BIND
                         ($$VAL _ T)))

(PUTPROPS old I.S.OPR MODIFIER)

(PUTPROPS smallest I.S.OPR [NIL = SUBST (GETDUMMYVAR)
                                '$$TEMP
                                '(BIND $$EXTREME $$TEMP DO (SETQ $$TEMP BODY)
                                       (COND ((OR (NULL $$EXTREME)
                                                  (LESSP $$TEMP $$EXTREME))
                                              (SETQ $$EXTREME $$TEMP)
                                              (SETQ $$VAL I.V.])

(PUTPROPS sum I.S.OPR ((SETQ $$VAL (PLUS $$VAL BODY))
                       BIND
                       ($$VAL _ 0)))

(PUTPROPS thereis I.S.OPR [(COND (BODY (SETQ $$VAL (OR I.V. T))
                                       (GO $$OUT])

(ADDTOVAR I.S.OPRLST ALWAYS AS BIND BY COLLECT COUNT DECLARE DECLARE%: DO EACHTIME FCOLLECT FINALLY FIND FIRST
                     FOR FROM IN INSIDE ISTHERE JOIN LARGEST NEVER OLD ON ORIGINAL REPEATUNTIL REPEATWHILE
                     SMALLEST SUCHTHAT SUM THEREIS THRU TO UNLESS UNTIL WHEN WHERE WHILE always as bind by
                     collect count declare declare%: do eachtime fcollect finally find first for from in
                     inside isthere join largest never old on original repeatuntil repeatwhile smallest
                     suchthat sum thereis thru to unless until when where while)

(ADDTOVAR CLISPFORWORDSPLST ALWAYS AS BIND BY COLLECT COUNT DECLARE DECLARE%: DO EACHTIME FCOLLECT FINALLY
                            FIND FIRST FOR FROM IN INSIDE ISTHERE JOIN LARGEST NEVER OLD ON ORIGINAL
                            REPEATUNTIL REPEATWHILE SMALLEST SUCHTHAT SUM THEREIS THRU TO UNLESS UNTIL
                            WHEN WHERE WHILE)

(RPAQQ CLISPDUMMYFORVARS ($$TEM0 $$TEM1 $$TEM2 $$TEM3 $$TEM4 $$TEM5 $$TEM6))

(ADDTOVAR SYSLOCALVARS $$TEM0 $$TEM1 $$TEM2 $$TEM3 $$TEM4 $$TEM5 $$TEM6)

(ADDTOVAR INVISIBLEVARS $$TEM0 $$TEM1 $$TEM2 $$TEM3 $$TEM4 $$TEM5 $$TEM6)

(ADDTOVAR SYSLOCALVARS $$VAL $$TEM $$LST1 $$LST2 $$LST3 $$LST4 $$LST5 $$LST6 $$END $$EXTREME)

(ADDTOVAR INVISIBLEVARS $$VAL $$END $$TEM $$LST1 $$LST2 $$LST3 $$LST4 $$LST5 $$LST6 $$EXTREME)

[PUTDEF 'I.S.OPRS 'FILEPKGCOMS '((COM MACRO [X (DECLARE%: EVAL@COMPILE (P * (DUMPI.S.OPRS . X]
                                      CONTENTS NILL)
                                  (TYPE DESCRIPTION "i.s. operators" GETDEF GETDEF.I.S.OPR WHENCHANGED (
                                                                                                   CLEARCLISPARRAY
                                                                                                         ]

(DEFINEQ

(DUMPI.S.OPRS
  [NLAMBDA X                                                     (* lmm "14-Aug-84 18:34")
                                                                 (* Dump I.S.OPRS definitions. -
                                                                 redefined to dump out same case as given)

    (for Y in X collect (OR (GETDEF.I.S.OPR Y)
                            (PROG1 NIL
                                (LISPXPRINT (LIST 'I.S.OPR Y 'not 'defined)
                                       T T))])

(GETDEF.I.S.OPR
  [LAMBDA (Y)                                                    (* lmm "14-Aug-84 18:34")
    (PROG (TEM BODY EVALFLG)
          (RETURN (CONS 'I.S.OPR
                        (CONS (KWOTE Y)
                              (OR [AND [SETQ TEM (LISTP (GETPROP Y 'CLISPWORD]
                                        (EQ (CAR TEM)
                                            'FORWORD)
                                        (COND
                                           [[AND (NLISTP (CDR TEM))
                                                 (SETQ BODY (GETPROP (CDR TEM)
                                                                   'I.S.OPR]
                                            (COND
                                               [(LISTP BODY)
                                                (CONS [KWOTE (COND
                                                                ((EQ (CAR (LISTP (CAR BODY)))
                                                                     '=)
                                                                 (SETQ EVALFLG T)
                                                                 (CDAR BODY))
                                                                (T (CAR BODY]
                                                      (COND
                                                         ((EQ (CADR BODY)
                                                              '=)
                                                          (LIST (KWOTE (CDDR BODY))
                                                                T))
                                                         [(CDR BODY)
                                                          (COND
                                                             (EVALFLG (SHOULDNT)))

          (* somehow there was an = in front of the i.s.type and not in front of the others.
          this shouldnt happen)

                                                          (LIST (KWOTE (CDR BODY]
                                                         (EVALFLG '(NIL T]
                                               (T (LIST (KWOTE BODY]
                                           ((AND (LISTP (CDR TEM))
                                                 (CADDR TEM))
                                            (LIST (KWOTE (CADDR TEM]
                                   (RETURN])
)


          (* * forDuration)

(ADDTOVAR DURATIONCLISPWORDS (TIMERUNITS timerUnits timerunits)
                                (USINGBOX usingBox usingbox)
                                (USINGTIMER usingTimer usingtimer)
                                (FORDURATION forDuration forduration DURING during)
                                (RESOURCENAME resourceName resourcename)
                                (UNTILDATE untilDate untildate))

(PUTPROPS TIMERUNITS CLISPWORD (FORWORD . timerUnits))

(PUTPROPS timerUnits CLISPWORD (FORWORD . timerUnits))

(PUTPROPS timerunits CLISPWORD (FORWORD . timerUnits))

(PUTPROPS USINGBOX CLISPWORD (FORWORD . usingBox))

(PUTPROPS usingBox CLISPWORD (FORWORD . usingBox))

(PUTPROPS usingbox CLISPWORD (FORWORD . usingBox))

(PUTPROPS USINGTIMER CLISPWORD (FORWORD . usingTimer))

(PUTPROPS usingTimer CLISPWORD (FORWORD . usingTimer))

(PUTPROPS usingtimer CLISPWORD (FORWORD . usingTimer))
```

```
(PUTPROPS FORDURATION CLISPWORD (FORWORD . forDuration))

(PUTPROPS forDuration CLISPWORD (FORWORD . forDuration))

(PUTPROPS forduration CLISPWORD (FORWORD . forDuration))

(PUTPROPS DURING CLISPWORD (FORWORD . during))

(PUTPROPS during CLISPWORD (FORWORD . during))

(PUTPROPS RESOURCENAME CLISPWORD (FORWORD . resourceName))

(PUTPROPS resourceName CLISPWORD (FORWORD . resourceName))

(PUTPROPS resourcename CLISPWORD (FORWORD . resourceName))

(PUTPROPS UNTILDATE CLISPWORD (FORWORD . untildate))

(PUTPROPS untilDate CLISPWORD (FORWORD . untilDate))

(PUTPROPS untildate CLISPWORD (FORWORD . untildate))

(PUTPROPS timerUnits \DURATIONTRAN T)

(PUTPROPS usingBox \DURATIONTRAN T)

(PUTPROPS usingTimer \DURATIONTRAN T)

(PUTPROPS forDuration \DURATIONTRAN T)

(PUTPROPS during \DURATIONTRAN T)

(PUTPROPS resourceName \DURATIONTRAN T)

(PUTPROPS untilDate \DURATIONTRAN T)

(PUTPROPS untildate \DURATIONTRAN T)

(DECLARE%: EVAL@COMPILE

[PUTDEF '\ForDurationOfBox 'RESOURCES '(NEW (\TIMER.MAKETIMER]
)

;; Currently there are four possible entries for the INFO property: EVAL, BINDS, LABELS, PROGN, or a list containg any or all of these.

;; EVAL is used to indicate that an nlambda evaluates its arguments.  EVAL affects DWIMIFY and CLISPIFY: neither will touch an nlambda that does not
;; have this property.

;; BINDS tells clispify and dwimify that CADR of the form is a list of variables being bound, a la prog.

;; PROGN says that only the last top level expression is being used for value. This affects the way OR's and AND's are clispified, for example.

;; Finally, LABELS indicates that top level atoms in this expression are not being evaluated. This tells clispify not to create atoms out of lists at the top
;; level.  LABELS also implies that none of the top level expressions are being used for value.

;; For example, FOR has info property just BINDS, (EVAL is unnecssary since FOR is not a function and its dwimifying and clispifying affected by its
;; clispword property), whereas PROG has (BINDS EVAL LABELS), and LAMBDA has (EVAL BINDS PROGN)

(PUTPROPS PROG INFO (EVAL BINDS LABELS))

(PUTPROPS PROG* INFO (EVAL BINDS LABELS))

(PUTPROPS RESETVARS INFO (EVAL BINDS LABELS))

(PUTPROPS RESETBUFS INFO EVAL)

(PUTPROPS RESETLST INFO (EVAL PROGN))

(PUTPROPS ADV-PROG INFO (EVAL BINDS LABELS))

(PUTPROPS ADV-SETQ INFO EVAL)

(PUTPROPS AND INFO EVAL)

(PUTPROPS ARG INFO EVAL)

(PUTPROPS COND INFO EVAL)

(PUTPROPS ERSETQ INFO EVAL)

(PUTPROPS NLSETQ INFO EVAL)

(PUTPROPS OR INFO EVAL)

(PUTPROPS PROG1 INFO EVAL)

(PUTPROPS PROG2 INFO EVAL)

(PUTPROPS PROGN INFO (EVAL PROGN))
```

```
(PUTPROPS RESETFORM INFO EVAL)

(PUTPROPS RESETSAVE INFO EVAL)

(PUTPROPS RESETVAR INFO EVAL)

(PUTPROPS RPAQ INFO EVAL)

(PUTPROPS RPTQ INFO EVAL)

(PUTPROPS FRPTQ INFO EVAL)

(PUTPROPS SAVESETQ INFO EVAL)

(PUTPROPS SETN INFO EVAL)

(PUTPROPS SETQ INFO EVAL)

(PUTPROPS UNDONLSETQ INFO EVAL)

(PUTPROPS XNLSETQ INFO EVAL)

(PUTPROPS SETARG INFO EVAL)

(PUTPROPS LET INFO (BINDS EVAL))

(PUTPROPS LET* INFO (BINDS EVAL))

(PUTPROPS RETURN INFO EVAL)

(PUTPROPS CLISP FILETYPE CL:COMPILE-FILE)

(DECLARE%: DONTEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS

(ADDTOVAR NLAMA DUMPI.S.OPRS)

(ADDTOVAR NLAML )

(ADDTOVAR LAMA )
)

(PUTPROPS CLISP COPYRIGHT ("Venue & Xerox Corporation" T 1982 1983 1984 1985 1986 1990))
```

**{MEDLEY}<sources>CLISP.;1      28-Jun-2024 18:34:03**

-- Listed on 30-Jun-2024 13:15:26 --

## FUNCTION INDEX

## PROPERTY INDEX

## VARIABLE INDEX

## OPTIMIZER INDEX