

File created: 24-Sep-2023 13:59:34 {WMEDLEY}<sources>COMPILE.;5

edit by: rmk

changes to: (VARS COMPILECOMS)
(FNS COMPSET)

previous date: 5-Jul-2021 13:46:39 {WMEDLEY}<sources>COMPILE.;4

Read Table: INTERLISP

Package: INTERLISP

Format: XCCS

```
::  
:: Copyright (c) 1984-1990, 2021 by Venue & Xerox Corporation.  
:: The following program was created in 1984 but has not been published  
:: within the meaning of the copyright law, is furnished under license,  
:: and may not be used, copied and/or disclosed except in accordance  
:: with the terms of said license.
```

(RPAQQ **COMPILECOMS**

```
[ (FNS BCOMPL BCOMPL.BODY PRINT-COMPILE-HEADER RESETOPENFILES BCOMPL1A BCOMPL2 BCOMPL3 BLOCK%: BRECOMPILE  
BRECOMPILE1 BRECOMPILE2 BRECOMPILE3 BLOCKCOMPILE BLOCKCOMPILE1 COMPSET COMPSETREAD COMPSETY  
COMPSETF RCOMP3 TCOMPL RECOMPILE RECOMP? COMPILE COMPILE1 COMPILE1A SHOULD-BE-DWIMIFIED? COMPEM  
GETCFILE SPECVARS LOCALVARS GLOBALVARS)  
(ADDVARS (NOLINKFNS HELP ERRORX ERRORSET EVALV FAULTVEVAL INTERRUPT SEARCHPDL MAPDL BREAK1 EDITE EDITL)  
(LINKFNS)  
(FREEVARS)  
(SYSSPECVARS HELPCLOCK LISPXHIST RESETSTATE OLDVALUE UNDOSE0 SPECVARS LOCALVARS GLOBALVARS)  
(SYSLOCALVARS)  
(LOCALFREEVARS)  
(BLKLIBRARY)  
(RETFNS)  
(BLKAPPLYFNS)  
(DONTCOMPILEFNS)  
(NLAML)  
(NLAMA)  
(LAMS)  
(LAMA))  
(INITVARS (SPECVARS T)  
(LOCALVARS SYSLOCALVARS))  
(INITVARS (DWIMIFYCOMPFLG NIL)  
(COMPILEHEADER "compiled on ")  
(COMPSETLST ' (ST F STF S Y N 1 2 NIL T))  
[COMPSETKEYLST ' ((ST "ore and redefine " KEYLST (" (F . "orget exprs"))  
(S . "ame as last time")  
(F . "ile only")  
(T . "o terminal")  
(1)  
(2)  
(Y . "es")  
(N . "o")  
[COMPSETDEFAULTKEYLST ' ((Y . "es")  
(N . "o")  
(BCOMPL.SCRATCH '{CORE}BCOMPL.SCRATCH)  
(RECOMPILEDEFAULT 'CHANGES)  
(COUTFILE T)  
(SVFLG T)  
(STRF T)  
(LSTFIL T)  
(LCFIL)  
(LAPFLG T))  
(DECLARE%: DONTCOPY (RECORDS COMPFILEDESCR)  
(MACROS DIGITCHARP)  
(GLOBALVARS SYSSPECVARS SYSLOCALVARS RECOMPILEDEFAULT COMPILE.EXT NOTCOMPILEDFILES BYTECOMPFLG  
COMPILEHEADER COMPVERSION BCOMPL.SCRATCH LINKEDFNS NOFIXVARSLST0 NOFIXFNSLST0 CLISPTRANFLG  
CLISPARRAY COMPSETKEYLST REREADFLG HISTSTR0 LISPXHISTORY COMPSETDEFAULTKEYLST FILERDTBL  
DWIMFLG DWIMWAIT))  
[P (MOVD? 'NIL 'FILECHANGES)  
(CL:PROCLAIM ' (CL:SPECIAL COMPVARMACROHASH))  
(CL:PROCLAIM ' (GLOBAL SYSSPECVARS SYSLOCALVARS COMPILE.EXT NOTCOMPILEDFILES CLISPARRAY FILERDTBL  
DWIMFLG DWIMWAIT LISPXHISTORY)  
(COMS ; COMPILEMODE  
(PROP VARTYPE COMPILEMODELST)  
(FNS COMPILEMODE))  
(DECLARE%: DONTVEVAL@LOAD DOEVAL@COMPILE DONTCOPY COMPILERVARS (ADDVARS (NLAMA GLOBALVARS LOCALVARS  
SPECVARS BLOCK%:)  
(NLAML BCOMPL3)  
(LAMA]))
```

(DEFINEQ

(**BCOMPL**

[LAMBDA (FILES CFILE NOBLOCKSFLG OPTIONSSET)

; Edited 5-Jul-2021 13:39 by rmk:

;; BCOMPL is like TCOMPL, except that it reads in all of FILES before starting any compilations, so that a BLOCK can contain functions in several
;; FILES. BLOCKS are set up using a DECLARE statement of the form

;; (DECLARE (BLOCK: BLKNAME BLKFN1 BLKFN2 ... (VAR1 VALUE) (VAR2 VALUE) ...) (BLOCK: BLKNAME ...) ...)

;; where BLKFN1 ... are the functions in the BLOCK, and VAR1 ... are values for ENTRIES, RETFNS, SPECVARS, etc. A variable setting of the form
;; (VAR .list) sets variable to UNION of the list with the variable's top level value. A variable setting of the form (VAR .ATOM) simply sets the variable
;; to that atom, e.g. (NOLINKFLG .T)

```
(RESETLST
  (LET ((NLAML NLAML)
        (NLAMA NLAMA)
        (LAMS LAMS)
        (LAMA LAMA)
        (DWIMIFYCOMPFLG DWIMIFYCOMPFLG)
        (EXPRSLSLST NIL)
        (NOFIXVARSLST NOFIXVARSLST)
        (NOFIXFNSLSLST NOFIXFNSLSLST)
        (*PRINT-ARRAY* T)
        (*PRINT-LENGTH* NIL)
        (*PRINT-LEVEL* NIL))
    (DECLARE (SPECVARS NLAML NLAMA LAMS LAMA DWIMIFYCOMPFLG EXPRSLSLST NOFIXVARSLST NOFIXFNSLSLST
                  *PRINT-ARRAY* *PRINT-LEVEL* *PRINT-LENGTH*))
```

;; Checks that all FILES are there, and if not, attempts spelling correction. Opens them for input, too, and returns the input stream

```
(BCOMPL.BODY (RESETOOPENFILES FILES)
  (FILE NOBLOCKSFLG OPTIONSSET)))
```

(BCOMPL.BODY

```
[LAMBDA (STREAMS CFILE NOBLOCKSFLG OPTIONSSET)
```

; Edited 5-Jul-2021 13:46 by rmk:

;; STREAMS is a list of streams. Compile everything on them, dumping to CFILE (default first stream.dcom). NOBLOCKSFLG means TCOMPL
;; instead of BCOMPL. OPTIONSSET is true if the Listing? question has already been asked.

;; RMK: Apply each input streams \EXTERNALFORMAT

```
(DECLARE (SPECVARS CFILE))
(PROG ((SPECVARS T)
      (LOCALVARS SYSLOCALVARS)
      DEFS CHANGES OTHERS FIRST BLOCKS BLKFNS FILEROOT TEM SCRATCHFILE DESTINATIONENV UNPACKFILE)
  (DECLARE (SPECVARS SPECVARS LOCALVARS CHANGES OTHERS FIRST BLOCKS BLKFNS DESTINATIONENV DEFS))
  [OR OPTIONSSET (COMPSET NIL ' (F %
```

; OPTIONSSET is T on calls from TCOMPL. In this case, the
; first COMPSET has already been performed.

```
(COMPSET (OR CFILE (PACKFILENAME 'HOST [CADR (FMEMB 'HOST (SETQ UNPACKFILE (UNPACKFILENAME
  (CAR STREAMS)
  'DIRECTORY
  (CADR (FMEMB 'DIRECTORY UNPACKFILE))
  'NAME
  (SETQ FILEROOT (CADR (FMEMB 'NAME UNPACKFILE)))
  'EXTENSION COMPFILE.EXT))
```

;; Edited by TT(8-June-90 : for Fix AR#2999)

```
[COND
  (LCFIL (SETQ SCRATCHFILE (OPENSTREAM BCOMPL.SCRATCH 'BOTH 'NEW)
  (RESETSAVE NIL (LIST 'BCOMPL3 NIL STREAMS SCRATCHFILE))
```

;; BCOMPL3 will close and if necessary delete all the appropriate files when bcompl finishes, or control-d or control-e occurs.

```
[LET (DFNFLG)
```

;; if top level value of DFNFLG is PROP, still want to evaluate expressions in declarations etc as though it were T. i.e. make
;; BCOMPL1A equivalent to doing a LOADCOMP

```
(for STREAM in STREAMS
  do (RESETLST
      (RESETSAVE NIL (LIST 'CLOSEF STREAM))
      (RESETSAVE (INPUT STREAM)) ; Needs to be primary input for some of the filepkg expressions
      ; to work
      (WITH-READER-ENVIRONMENT *OLD-INTERLISP-READ-ENVIRONMENT*
        (until (OR (NULL (SETQ TEM (READ STREAM)))
                  (EQ TEM 'STOP))
          do (CL:WHEN (EQ (CAR (LISTP TEM))
                        'DEFINE-FILE-INFO)
                    (\EXTERNALFORMAT STREAM (OR (LISTGET (CDR TEM)
                                                          :FORMAT)
                                                  :XCCS)))
              (BCOMPL1A TEM 'DEFAULT 'DEFAULT 'DEFAULT STREAM))))]
  (SETQ NOFIXFNSLSLST (APPEND NLAMA NLAML LAMS (NCONC [MAPCAR DEFS (FUNCTION (LAMBDA (X)
  (RCOMP3 (CAR X)
  (CADR X)
  NOFIXFNSLSLST)))
```

;; The BCOMPL1 reads in FILES. It returns a list of variables set in the files. The RCOMP3 adds function to NLAMA, LAMS, etc., and returns a
;; list of functions. NOFIXFNSLSLST is reset in case there is any dwimifying to be done.

```
(WITH-READER-ENVIRONMENT (OR DESTINATIONENV (SETQ DESTINATIONENV *OLD-INTERLISP-READ-ENVIRONMENT*))
  (COND
```

```
(LCFIL (\EXTERNALFORMAT LCFIL (OR (FETCH (READER-ENVIRONMENT REFORMAT) OF DESTINATIONENV)
:XCSS))
(PRINT-COMPILE-HEADER STREAMS [LIST (COND
(NOBLOCKSFLG 'tcompl'd)
(T 'bcompl'd]
DESTINATIONENV)))
(COND
(SCRATCHFILE
;; writes others on a scratchfile so space can be freed up. will be copied onto lcfil aftr compilation.
(\EXTERNALFORMAT SCRATCHFILE (\EXTERNALFORMAT LCFIL))
(for X in OTHERS do (PRINT X SCRATCHFILE))
(PRINT NIL SCRATCHFILE)
(SETQ OTHERS NIL)))
[OR DWIMIFYCOMPFLG (SETQ DWIMIFYCOMPFLG (EQMEMB 'CLISP (GETPROP FILEROOT 'FILETYPE)
; The FILETYPE may have been set during the course of
; BCOMPL1.
[MAPC FIRST (FUNCTION (LAMBDA (X)
(PRINT X LCFIL))
(LISPXHIST)
(DECLARE (SPECVARS LISPXHIST))
;; compile blocks MAPC not used because BCOMPL2 checks BLOCKS. lispxhist rebound bcause no need to save information
;; when compiling from file
(AND NOBLOCKSFLG (GO NOBLOCKLP))
BLOCKLP
(COND
(BLOCKS (BCOMPL2 (CAR BLOCKS))
(SETQ BLOCKS (CDR BLOCKS))
(GO BLOCKLP)))
NOBLOCKLP
(COND
(DEFS (AND (NOT (FMEMB (CAAR DEFS)
DONTCOMPILEFNS))
(COMPIL1 (CAAR DEFS)
(CADAR DEFS)))
(SETQ DEFS (CDR DEFS))
(GO NOBLOCKLP))
(RETURN (FULLNAME LCFIL]))
; COMPIL1 other functions. done this way instead of MAPC to
; release the defs as soon as possible.
```

(PRINT-COMPILE-HEADER

```
[LAMBDA (STREAMS HOW ENV)
```

(* bvm%: "15-Sep-86 18:04")

(* "Prints the header at the start of a compiled file. First is a possible environment definition, then a FILECREATED expression naming the source files and what was compiled and how. STREAMS is list of sources, HOW is a list describing the details of how")

```
(WITH-READER-ENVIRONMENT *OLD-INTERLISP-READ-ENVIRONMENT*
(if ENV
then (PRINT-READER-ENVIRONMENT ENV LCFIL)
(SET-READER-ENVIRONMENT ENV))
(PRINT (NCONC (LIST 'FILECREATED (DATE)
(CONS COMPILEHEADER (MAPCAR STREAMS (FUNCTION FULLNAME)))
COMPVERSION)
(APPEND HOW (LIST 'in HERALDSTRING 'dated MAKESYSDATE)))
LCFIL))])
```

(RESETOPENFILES

```
[LAMBDA (FILES)
```

(* bvm%: " 2-Aug-86 17:00")

```
[RESETSAVE NIL (LIST [FUNCTION (LAMBDA (FILES)
(MAPC FILES (FUNCTION CLOSEF?])
(SETQ FILES (for F inside FILES collect (OPENSTREAM F 'INPUT]
FILES])])
```

(BCOMPL1A

```
[LAMBDA (X COMPCOPYFLG COMPEVALFLG FIRSTFLG)
```

; Edited 20-Jan-88 17:48 by jds

```
(PROG (TEM)
[SELECTQ (CAR (LISTP X))
(FILECREATED (SETQ CHANGES (NCONC (FILECHANGES X 'FNS)
CHANGES))
(SETQ FIRST (NCONC1 FIRST X)))
(DEFINE-FILE-INFO
(SETQ TEM (EVAL X))
(COND
((NOT DESTINATIONENV)
(SETQ DESTINATIONENV TEM)))
(DEFINEQ [COND
((EQ COMPCOPYFLG 'DEFAULT)
(SETQ DEFS (NCONC DEFS (CDR X]))
(DECLARE%:
; supercedes DECLARE, DEFLIST with third argument, and
; PROG with funny atom.
; Note that DECLARE: itself isn't copied to compiled file. nor is it
; evaluated at compile time.
```

; Will dump new dcom file in the environment of the first source
; DEFAULT means to copy the COMPILED definitions to the file.
; supercedes DECLARE, DEFLIST with third argument, and
; PROG with funny atom.
; Note that DECLARE: itself isn't copied to compiled file. nor is it
; evaluated at compile time.

```
(PROG ((DFNFLAG DFNFLAG)
      (FILEPKGFLAG)
      (COMPEVALFLAG0 COMPEVALFLAG)
      (COMPCOPYFLAG0 (EQ COMPCOPYFLAG 'DEFAULT))
      (FIRSTFLAG0 FIRSTFLAG)
      (X (CDR X))))
```

;; FLAG is the flag in effect when the DECLARE: started, FLAG0 the current flag. the use of two flags permits turning a
 ;; flag off and then back on and the same level, but prohibiting turning the flag on at a lower level, i.e. overriding a
 ;; higher command with a lower one

```
LP [COND
  ((NLISTP X)
   (RETURN))
  ((LISTP (CAR (LISTP X)))
   (BCOMPL1A (CAR (LISTP X))
              COMPCOPYFLAG0 COMPEVALFLAG0 FIRSTFLAG0))
  (T (SELECTQ (CAR (LISTP X))
             (DONTCOPY (SETQ COMPCOPYFLAG0 NIL))
             (DOCOPY COPY)

      ;; when a DECLARE: is encountered inside of a DECLARE: DOCOPY, want the entire DECLARE:
      ;; copied (because it may contain EVAL@LOADWHEN tags). in this case we don't want each
      ;; individual element also to be copied. starting compcopyflag0 off at NIL and only resetting when
      ;; compcopyflag is DEFAULT achieves this.

      (AND (EQ COMPCOPYFLAG 'DEFAULT)
            (SETQ COMPCOPYFLAG0 T)))
      (COPYWHEN (AND (EQ COMPCOPYFLAG 'DEFAULT)
                    (SETQ COMPCOPYFLAG0 (AND (EVAL (CADR (LISTP X)))
                                              T))))
               (SETQ X (CDR (LISTP X))))
      (FIRST ; these expressions are copied to the compiled file before any
             ; functions.
             (AND FIRSTFLAG (SETQ FIRSTFLAG0 T)))
      (NOTFIRST (SETQ FIRSTFLAG0 NIL))
      (DONTVAL@COMPILE
       (SETQ COMPEVALFLAG0 NIL))
      ((DOEVAL@COMPILE EVAL@COMPILE)
       (AND COMPEVALFLAG (SETQ COMPEVALFLAG0 T)))
      (EVAL@COMPILEWHEN
       (AND COMPEVALFLAG (SETQ COMPEVALFLAG0
                               (AND (EVAL (CADR (LISTP X)))
                                   T))))
      (SETQ X (CDR (LISTP X))))
  (COMPILEVAR ; From ADDVARS NLAMA and NLAML in prettydef. The resetting of dnfllg
              ; will suppress the (NLAMA RESET) message.
              (SETQ DFNFLAG T))
  (EVAL@LOADWHEN
   (SETQ X (CDR (LISTP X))))
  ((DONTVAL@LOAD DOEVAL@LOAD EVAL@LOAD)
   NIL)
  (CL:FORMAT (\GETSTREAM COUTFILE 'OUTPUT)
             "(~S unrecognized DECLARE tag)~%"
             (CAR (LISTP X))
             (SETQ X (CDR (LISTP X))))
  (GO LP)))
((SETQ SETQQ RPAQ RPAQQ RPAQ?)
 (push NOFIXVARSLST (CADR X))
 (AND (EQ COMPCOPYFLAG 'DEFAULT)
      (SETQ COMPCOPYFLAG T)))
(COND
 ((EQ (CAR (LISTP X))
      COMMENTFLAG)
  (RETURN))
 ((EQ COMPCOPYFLAG 'DEFAULT)
  (SETQ COMPCOPYFLAG T]))
[COND
 ((EQ COMPCOPYFLAG T)
  ;; OTHERS is a list of expressions to be written out later. FIRST is a list of expressions to be written out before the compiled code
  ;; goes out
  (COND
   ((EQ FIRSTFLAG T)
    (SETQ FIRST (NCONC1 FIRST X)))
   (T (SETQ OTHERS (NCONC1 OTHERS X))))
 (COND
  ((EQ COMPEVALFLAG T)
   (EVAL X)))
 (RETURN))
```

BCOMPL2

```
[LAMBDA (BLOCK FILEMAPLST COREOK)
```

; Edited 6-Dec-86 03:59 by lmm
 ; This function processes a single block.
 ; FILEMAPLST is given when recompiling.

```
(RESETLST
 (PROG ((GLOBALVARS GLOBALVARS)
```

```

(RETFNS RETFNS)
(BLKLIBRARY BLKLIBRARY)
(NOLINKFNS NOLINKFNS)
(LINKFNS LINKFNS)
(DONTCOMPILEFNS DONTCOMPILEFNS)
(SPECVARS SPECVARS)
(LOCALVARS LOCALVARS)
(BLKNAME (CAR BLOCK))
BLKAPPLYFNS ENTRIES LOCALFREEVARS X TEM LST (BNDLEV 0)
(TEM2))
(DECLARE (SPECVARS GLOBALVARS RETFNS BLKLIBRARY NOLINKFNS LINKFNS DONTCOMPILEFNS))
(COND
  ((NULL BLKNAME) ; BLKNAME NIL means regular compiling unless declared
                    ; otherwise
   (SPECVARS . T))
  (T (LOCALVARS . T)))
(GO LP1)

```

LP

;; Loop through BLOCK making assignments for non-atomic expressions and gathering on LST the definitions for the atoms.

```

(COND
  ((LISTP (SETQ X (CAR BLOCK))) ; A declaration
   [SETQ TEM (COND
     ((EQ (CADR X)
          '*)
      (EVAL (CADDR X)))
     (T (CDR X))
    ])
   [SELECTQ (CAR X)
     ((SPECVARS LOCALVARS)
      (EVAL X))
     (SET (CAR X)
          (COND
            ((NLISTP (CDR X))
             (CDR X))
            ([LISTP (SETQ TEM2 (EVAL (CAR X))
                  (APPEND TEM TEM2))
              (T TEM)]
            (GO LP1))
     ((AND FILEMAPLST (NULL BLKNAME)
            (NOT (RECOMP? X FNS)))
      ;; Function is not going to be compiled, so no point in looking up its definition. Note that BRECOMPILE never calls BCOMPL2 on
      ;; a block (other than one with a NIL name) unless the entire block is going to have to be recompiled.
      (SETQ TEM (LIST X)))
     [FILEMAPLST (COND
       ((NULL (SETQ TEM (BRECOMPILE3 X FILEMAPLST COREOK)))
        [COMPEM (CONS X '(not compileable)]
        (GO LP1)
       ((SETQ TEM (FASSOC X DEFS))
        (AND [NOTANY (CDR BLOCKS)
                    (FUNCTION (LAMBDA (X)
                              (FMEMB (CAR TEM)
                                      (CDR X))
                              (SETQ DEFS (DREMOVE TEM DEFS)))
                    ; This is done primarily to release the space for recompilation as
                    ; soon as possible.
        ])
     [AND COREOK (EXPRP (SETQ TEM (VIRGINFN X T))
      ;; this is a new feature. it is designed to allow the user to have definitions for functions in a library file and to load them in to the
      ;; files that need them at compile time by doing a loadfns. thus he doesn't have to have a definition or blklibrarydef for the function
      ;; in each file that uses it. note that when recompiling, this feature falls out because BRECOMPILE3 checks for an incore
      ;; definition before it goes to the file anyway, and doesn't distinguish between definitions of functions in the file, and those that just
      ;; happen to have in core definitions.
      (SETQ TEM (LIST X TEM T))
      (COMPEM (CONS X '(not on file, compiling in core definition)]
      (T [COMPEM (CONS X '(not compileable)]
          (GO LP1))))
      (SETQ LST (NCONC1 LST TEM))
      (SETQ BLKFNS (CONS X BLKFNS)) ; A list of those functions contained in blocks. All others will be
                                   ; compiled separately.
    ])
  LP1 (COND
    ((SETQ BLOCK (CDR BLOCK))
     (GO LP))
    ((AND (NULL LST)
          (NULL BLKNAME)
     ;; BLOCK consists of single function: BLKNAME, e.g. (FOO) or (FOO (SPECVARS --) (globavars --)) have to go back through
     ;; loop to look up definition on defs.
     (SETQ BLOCK (LIST BLKNAME))
     (GO LP)))
  (COND
    ((NULL BLKNAME)
     ;; By using NIL for BLOCK name, user indicates this is a non-block compilation. However, he can set LINKFLG to T, thereby
     ;; causing all calls to be linked, even though he is not compiling a BLOCK. He can also set NOLINKFNS and GLOBALVARS.
     (PROG NIL

```

```

L1 (COND
  ((NULL LST)
   (RETURN))
  [(OR (NULL FILEMAPLST)
        (AND (CADAR LST)
              (RECOMP? (CAAR LST)
                       FNS)))
        (AND (NOT (MEMB (CAAR LST)
                       DONTCOMPILEFNS))
              (COMPILE1 (CAAR LST)
                       (CADAR LST)
                       (CADDAR LST)
                       (T (BRECOMPILE1 (CAAR LST)
                                         T)))
                       (SETQ LST (CDR LST))
                       (GO L1)))]
  (T (BLOCKCOMPILE1 BLKNAME (PROG1 LST (SETQ LST NIL))
                    ENTRIES)))
(RETURN)))]

```

; BLOCKCOMPILE1 will also make some checks on ENTRIES.

(BCOMPL3

```

[NLAMBDA (CFILE FILES SCRATCHFILE)
; Edited 15-Sep-89 11:23 by bvm
; Cleans up after brecompile and bcompl have finished
; operating,
[COND
  (SCRATCHFILE [COND
    ((NULL RESETSTATE) ; finished successfully.
     (COPYBYTES SCRATCHFILE LCFIL 0 (GETFILEPTR SCRATCHFILE)
      (CLOSEF? SCRATCHFILE)
      (DELFILE (FULLNAME SCRATCHFILE))
    ]
  [COND
    (LCFIL (CLOSEF? LCFIL)
     (AND RESETSTATE (DELFILE (FULLNAME LCFIL))
    ]
  [COND
    ((AND LSTFIL LSTFIL1)
     (CLOSEF? LSTFIL1))
  [COND
    (CFILE (CLOSEF? CFILE)))
  (for FILE in FILES unless RESETSTATE do ; Finished successfully--remove files from NOTCOMPILEDFILES
    (/SETATOMVAL 'NOTCOMPILEDFILES (REMOVE (ROOTFILENAME FILE)
      NOTCOMPILEDFILES)))
  (AND (NULL RESETSTATE)
        (NEQ (POSITION COUTFILE)
              0)
        (TERPRI COUTFILE])

```

(BLOCK%:

```

[NLAMBDA X ; (* wt%: "26-FEB-78 20:27")
(* Used in DECLARE%: expressions to set up blocks. See comment in BCOMPL.
probably this should be implemented by having bcompl1 specifically check for BLOCK%: rather than simply having EVAL
called, because that way can distinguish between block declarations in file being compiled from block declarations in a file
being LOADCOMP'ed. for now this is handled by having LOADCOMP rebind BLOCKS.)
(SETQ BLOCKS (NCONC1 BLOCKS X))

```

(BRECOMPILE

```

[LAMBDA (FILES CFILE FNS NOBLOCKSFLG) ; Edited 5-Jul-2021 09:28 by rmk:
;;; FNS is a list of functions to be recompiled. The object is to make a file that looks exactly like that produced by BCOMPL except to greatly reduce the
;;; work by copying from CFILE the compiled definitions those functions not being recompiled.
;;; BRECOMPILE is driven by the source file(s). The algorithm is whenever a DEFINEQ is encountered, process all of the functions in the DEFINEQ as
;;; follows: COMPILE the definition of the function if it is on the list FNS, or if FNS is EXPRS and the function is currently defined as an EXPR. Otherwise
;;; copy its compiled definition from CFILE. Note that functions with compiled definitions in CFILE that do not appear in PFILE are NOT copied. This
;;; corresponds to the case where functions have been deleted from the source file.
;;; The value FNS = CHANGES means recompile anything marked changed in the file header.
;;; (RECOMPILE file cfile fns) is equivalent to (BRECOMPILE file cfile fns T).
;;; Note that CFILE=NIL is interpreted as meaning file.dcom even when FNS supplied.

```

```

(RESETLST
  (PROG ((*PRINT-ARRAY* T)
         (*PRINT-LENGTH* NIL)
         (*PRINT-LEVEL* NIL)
         (NLAMA NLAMA)
         (NLAML NLAML)
         (LAMS LAMS)
         (LAMA LAMA)
         (DWIMIFYCOMPFLG DWIMIFYCOMPFLG)
         (EXPRSLSLST NIL)

```

```

(NOFIXFNSLST NOFIXFNSLST)
(NOFIXVARSLST NOFIXVARSLST)
(BUILDMAPFLG T)
(SPECVARS T)
(LOCALVARS SYSLOCALVARS)
(AUXFILECOM T)
CHANGES OTHERS FIRST FILEMAPLST FNLST BLKFNS BLOCKS FILE FILE.COM TEM ADRLST SCRATCHFILE COREOK
DESTINATIONENV MSG)
(DECLARE (SPECVARS *PRINT-ARRAY* *PRINT-LENGTH* *PRINT-LEVEL* NLAMA NLAML LAMS LAMA
DWIMIFYCOMPLG EXPRSLST NOFIXFNSLST NOFIXVARSLST BUILDMAPFLG SPECVARS LOCALVARS
CHANGES OTHERS FIRST BLKFNS BLOCKS DESTINATIONENV ADRLST FILEMAPLST CFILE FNS
FILE))
(COND
  ((AND (NULL CFILE)
        (NULL FNS))
    (SETQ FNS RECOMPILEDEFAULT)))
(RESETSAVE (INPUT))
(SETQ FILES (RESETOPENFILES FILES))
(COND
  ((SETQ TEM (for FILE in FILES when (NOT (RANDACCESSP FILE)) collect (FULLNAME FILE)))
    (GO NONRAND)))
  (SETQ FILE (UNPACKFILENAME (CAR FILES)))
  (SETQ FILE.COM (PACKFILENAME 'HOST (CADR (FMEMB 'HOST FILE))
    'DIRECTORY
    (CADR (FMEMB 'DIRECTORY FILE))
    'NAME
    (SETQ FILE (CADR (FMEMB 'NAME FILE)))
    'EXTENSION COMPILE.EXT))

```

:: Edited by TT (8-June-90 : for fix AR#2999)

```

(COND
  ((EQ FNS 'ALL)
    (GO BRECALL)))
CFILERETRY
(COND
  ([NLSETQ (SETQ CFILE (OPENSTREAM (OR CFILE FILE.COM)
    'INPUT
    'OLD NIL ' ((TYPE BINARY]
    (COND
      ((NOT (RANDACCESSP CFILE))
        (SETQ TEM (CLOSEF CFILE))
        (GO NONRAND))
      ([OR [NULL (SETQ DESTINATIONENV (GET-ENVIRONMENT-AND-FILEMAP (CAR FILES]
        (CL:MULTIPLE-VALUE-BIND (ENV DUMMY START)
          (\PARSE-FILE-HEADER CFILE)
            (COND
              ((OR (NULL ENV)
                    (NOT (EQUAL-READER-ENVIRONMENT ENV DESTINATIONENV)))
                T)
              (T
                (SETFILEPTR CFILE START) (* "Position cfile back to start")
                NIL))))
        (SETQ TEM (CLOSEF CFILE))
        (SETQ MSG " has different reader environment than the new file")
        (GO NONREC))
      (GO BREC))
    ((OR (AND (EQ AUXFILECOM T)
      [SETQ AUXFILECOM (SPELLFILE (ROOTFILENAME (OR CFILE FILE.COM]
        (SETQ CFILE AUXFILECOM)
        (GO CFILERETRY))
      (EQ (ASKUSER DWIMWAIT 'Y (LIST (OR CFILE FILE.COM)
        "not found;" " compile all functions on "
        (FULLNAME (CAR FILES))
        "instead"))
        'Y)) ; Edited by TT(8-June-90 : for Fix AR#8017)
    (GO BRECALL))
  ((EQ [ASKUSER DWIMWAIT 'Y (CONS "Just forget about compiling" (MAPCAR FILES
    (FUNCTION FULLNAME]
    'Y)
    (SELECTQ (CAR READBUF)
      ((ST F STF) (* "E.g. From CLEANUP.")
      (SETQ READBUF (CDR READBUF)))
    NIL)
    (RETFROM 'BRECOMPILE))
  (T (PRIN1 "File to use for CFILE (source of compiled definitions not being recompiled): " T)
    (SETQ CFILE (READ T T))
    (GO CFILERETRY)))
BRECALL
(SETQ FNS ALL)
(SETQ CFILE NIL)
BREC
(COMPSET NIL '(S T %
))
[SETQ LCFIL (OPENSTREAM FILE.COM 'OUTPUT 'NEW NIL ' ((TYPE BINARY]
(SETQ SCRATCHFILE (OPENSTREAM BCOMPL.SCRATCH 'BOTH 'NEW))
(RESETSAVE NIL (LIST 'BCOMPL3 CFILE FILES SCRATCHFILE))

```

:: BCOMPL3 will close and if necessary delete all the appropriate files when brecompile finishes, or control-d or control-e occurs. Note that

;; this call differs from the call for bcompl in that cfile is also specified. this corresponds to the fact that recompile has an extra file open.

```
[SETQ COREOK (for X in FILES always (AND (EQ (CDAR (GETPROP (SETQ TEM (ROOTFILENAME X))
'FILEDATES))
X)
(FMEMB (CDAR (GETPROP TEM 'FILE))
' (LOADFNS T])
```

```
[SETQ FILEMAPLST
(for STREAM in FILES
collect (LET ((LDFLG 'EXPRESSIONS)
(VARLST 'COMPILING)
DONELST FNLST)
(DECLARE (SPECVARS LDFLG VARLST DONELST FNLST))
; FNLST etc are used free in LOADFNSCAN
(SETFILEPTR STREAM 0)
(INPUT STREAM)
```

;; LOADFNSCAN scans the file, building a map if one not already there. Value is the map. In addition, sets
;; DONELST to a list of all non-definex expressions.

```
(CL: MULTIPLE-VALUE-BIND (ENV MAP FILECREATEDLOC)
(GET-ENVIRONMENT-AND-FILEMAP STREAM)
(DECLARE (CL: SPECIAL FILECREATEDLOC))
; used by LOADFNSCAN
(WITH-READER-ENVIRONMENT ENV
(create COMPFILEDESCR
COMPFILESTREAM _ STREAM
COMPFILEENV _ ENV
COMPFILEMAP _ (LOADFNSCAN MAP)
COMPFILEXPRS _ (DREVERSE DONELST))))]
```

```
[SETQ FNLST (for DESCR in FILEMAPLST join (for DEFQ in (CDR (fetch COMPFILEMAP of DESCR))
join (for X in (CDDR DEFQ) collect (CAR X]
```

;; FILEMAPLST is a list of information about each file, including its name, filemap and non-definex expressions. The first entry on the filemap
;; is NIL. We start mapping down CDR of the filemap, and each element therein corresponds to a single DEFINEX, in the form (start stop .
;; fnEntries). fnEntries is a list of (FN start . stop), so the inner MAPCAR gathers up the names of the functions. The reason for not asking
;; LOADFNS to do this is in most cases the map will already have been built, so LOADFNS won't even go inside of the definex.

```
[for DESCR in FILEMAPLST do (for FORM in (fetch COMPFILEXPRS of DESCR)
do (BCOMPL1A FORM 'DEFAULT 'DEFAULT 'DEFAULT])
```

;; BCOMPL1A adds VARS set in the files to NOFIXVARSLST. NOFIXFNLST and NOFIXVARSLST are reset in case there is any
;; dwimifying to be done BCOMPL1 also sets free variable OTHERS to list of expressions to be printed on compiled file when all is done.

```
(SETQ NOFIXFNLST (APPEND NLAMA NLAML LAMS FNLST NOFIXFNLST))
(WITH-READER-ENVIRONMENT (SETQ DESTINATIONENV (fetch COMPFILEENV of (CAR FILEMAPLST)))
; Start writing the compiled file. Use environment of one of the
; source files--usually the only one
```

```
(if LCFIL
then (\EXTERNALFORMAT LCFIL (OR (LISTGET DESTINATIONENV :FORMAT)
:XCCS))
(PRINT-COMPILER-HEADER FILES
[CONS (if NOBLOCKSFLG
then 'recompiled
else 'brecompiled)
(if (EQ FNS 'ALL)
then (LIST 'ALL)
else (CONS (SELECTQ FNS
(CHANGES 'changes%)
((EXPRS T)
'exprs%)
'explicitly%)
(OR [SUBSET FNLST (FUNCTION (LAMBDA (X)
(RECOMP? X FNS])
(LIST 'nothing]
```

```
DESTINATIONENV))
[MAPC FNLST (FUNCTION (LAMBDA (X)
(RCOMP3 X (VIRGINFN X)
```

```
(if SCRATCHFILE
then ;; writes others on a scratchfile so space can be freed up. will be copied onto lcfil aftr compilation.
```

```
(\EXTERNALFORMAT SCRATCHFILE (\EXTERNALFORMAT LCFIL))
(for X in OTHERS do (PRINT X SCRATCHFILE))
(PRINT NIL SCRATCHFILE)
(SETQ OTHERS NIL)
```

```
(for X in (PROGN FIRST) do (PRINT X LCFIL))
[OR DWIMIFYCOMPFLG (SETQ DWIMIFYCOMPFLG (EQMEMB 'CLISP (GETPROP FILE 'FILETYPE))
(OR (EQ FNS 'ALL)
(INPUT CFILE))
```

```
[if (NOT NOBLOCKSFLG)
then (for BLOCK in BLOCKS do (if (NULL (CAR BLOCK))
then (BCOMPL2 BLOCK FILEMAPLST)
elseif (for X in BLOCK thereis (AND (LITATOM X)
(RECOMP? X FNS)))
then ; If any function in the BLOCK is to be recompiled, the whole
; BLOCK must be recompiled.
(BCOMPL2 BLOCK FILEMAPLST COREOK)
else (BRECOMPILE1 BLOCK]
```

;; NOBLOCKSFLG is T for calls from RECOMPILE. In this case, even if there were any blocks, ignore them.
;; Now COMPILE rest of functions.


```

    (for x in FNLST do (if (OR (FMEMB X BLKFNS)
                              (FMEMB X DONTCOMPILEFNS))
                          elseif (RECOMP? X FNS)
                          then ;; The HELP is bcause if X is on FNS, then it follows X is in the file map, and brecompile3
                               ;; should be able to produce its definition.
                               (COMPILE1 X (CADR (SETQ TEM (BRECOMPILE3 X FILEMAPLST COREOK)))
                                           (CADDR TEM))
                               else (BRECOMPILE1 X T))))
    (RETURN (FULLNAME LCFIL))
NONRAND
    (SETQ MSG " is not RANDACCESSP")
NONREC
    (printout T TEM MSG ", using " (if NOBLOCKSFLG
                                       then 'TCOMPL
                                       else 'BCOMPL)
              " instead." T)
    (RETURN (BCOMPL.BODY FILES NIL NOBLOCKSFLG))))))

```

(BRECOMPILE1

[LAMBDA (FN/BLOCK NOBLOCKSFLG)

(* bvm%: "29-Aug-86 22:41")

(* Looks for FN/BLOCK and its subfunctions on CFILE, skipping

till found.)

(COND

[(AND (NULL NOBLOCKSFLG)

BYTECOMPFLG

(NEQ BYTECOMPFLG 'NOBLOCK))

(PROG [(LST (APPEND (CDR (ASSOC 'ENTRIES FN/BLOCK))

(CDR (ASSOC 'RETFNS FN/BLOCK))

(CDR (ASSOC 'BLKAPPLYFNS FN/BLOCK))

(LISTP (CDR (ASSOC 'NOLINKFNS FN/BLOCK)))

(LIST (CAR FN/BLOCK)

(for X in (CDR FN/BLOCK) when (LITATOM X) do (SETQ BLKFNS (CONS X BLKFNS))

(BRECOMPILE1 (if (NOT (MEMB X LST))

then

(* functions that are normally "visible" are compiled with the same names.

othrs use this naming convention)

(SETQ X (PACK* '\ (CAR FN/BLOCK

)

/' X))

else X)

T]

(T

(PROG ((NAME (if NOBLOCKSFLG

then FN/BLOCK

else (CAR FN/BLOCK)))

X ADR (ADRLST0 ADRLST))

LP (SETQ ADR (GETFILEPTR CFILE))

(if [NULL (ATOM (SETQ X (READ CFILE]

elseif (OR (EQ X NAME)

(BRECOMPILE2 X NAME))

then

(PRIN2 X COUTFILE T)

(PRIN1 ' ", " COUTFILE)

(OUTPUT LCFIL)

(LCSKIP X T)

(* copy the function)

(if (EQ X NAME)

then

[if (NULL NOBLOCKSFLG)

then

(SETQ BLKFNS (CONS X BLKFNS))

(for X in (CDR FN/BLOCK)

do (if (NLISTP X)

then (SETQ BLKFNS (CONS X BLKFNS))

elseif (EQ (CAR X)

'ENTRIES)

then (for X in (CDR X)

do (if (EQ X NAME)

then

(* already copied, e.g. NAME is block name as well as an entry)

elseif (PROGN (SETQ ADR (GETFILEPTR CFILE))

(NEQ (READ CFILE)

X))

then [COMPEN (CONS X '(not found)

(SETFILEPTR CFILE ADR)

else (PRIN2 X COUTFILE T)

(PRIN1 ' ", " COUTFILE)

(LCSKIP X T]

(RETURN))

elseif (AND (EQ ADRLST0 ADRLST)

(for Y in ADRLST thereis

(* NAME is not the next function on the file. Before skipping this function, see if NAME has been encountered earlier by scanning ADRLST. This saves skipping all the way down to the end of the file in the case that NAME is simply out of order. Only do this the first time, i.e. once you hve determined that NAME is not on ADRLST, and skipped X, then no reason to recheck ADRLST.)

```

      (if (OR (EQ (CAR Y)
                NAME)
            (BRECOMPILE2 (CAR Y)
                         NAME))
          then (* NAME was previously encountered and skipped over, e.g.
                out of order.)
              (SETFILEPTR CFILE (CDR Y))
              (BRECOMPILE1 FN/BLOCK NOBLOCKSFLG)
              (SETFILEPTR CFILE ADR)
              (* Reset filepointer back to where it was.)
              T))
    then (RETURN)
  elseif (OR (NULL X)
            (EQ X 'STOP))
    then (if (SETQ X (BRECOMPILE3 NAME FILEMAPLST))
           then (COMPILE1 NAME (CADR X)
                    (CADDR X))
           else [COMPEN (CONS NAME '(not found)

(* The only way i can see the COMPEN happening is if a function is included in a block declaration but is not in one of the
files, since the list of functions used to drive brecompile/recompile is precisely all of the functions on the file.)
)
      (SETFILEPTR CFILE ADR) (* So next read wont hit end of file.)
      (RETURN)
    else (SETQ ADRLST (NCONC1 ADRLST (CONS X ADR)))
          (LCSKIP X)
          (GO LP])

```

(BRECOMPILE2
[LAMBDA (X FN) (* bvm%: "22-OCT-82 15:45")

(* True if X is a sub-function of FN, i.e. X is FN followed by one or more Annnn substrings.)

```

(AND (STRPOS FN X 1 NIL T)
     (PROG [(NX (ADD1 (NCHARS X)))
           (N (ADD1 (NCHARS FN))
            LP (COND
                ([AND (ILEQ (IPLUS N 5)
                           NX)
                  (EQ (NTHCHARCODE X N)
                     (CHARCODE A))
                 (from 1 to 4 always (DIGITCHARP (NTHCHARCODE X (add N 1]
                (COND
                    ((EQ (add N 1)
                       NX)
                     (RETURN T))
                    (T (GO LP])

```

(BRECOMPILE3
[LAMBDA (FN FILEMAPLST COREOK) (* bvm%: "29-Aug-86 22:07")

(* "returns definition of FN, either from in core, or from the file.")

```

(LET (DEF STREAM)
  (COND
    ([AND COREOK (EXPRP (SETQ DEF (VIRGINFN FN T))
                       (* "Value is of the form (FN DEF FLG) where FLG=T means the
                          definition was obtained from in core, so that it is ok to do spelling
                          correction.")
      (LIST FN DEF T))
    (T (for FILEDESCR in FILEMAPLST when [PROGN (SETQ STREAM (fetch COMPFILESTREAM of FILEDESCR))
        (for Y in (CDR (fetch COMPFILEMAP of FILEDESCR))
          thereis (SETQ DEF (FASSOC FN (CDDR Y))
        do (SETFILEPTR STREAM (CADR DEF))
          (SETQ DEF (WITH-READER-ENVIRONMENT (fetch COMPFILEENV of FILEDESCR)
        (READ STREAM))) (* "TEM is an arg to DEFINEQ, of the form (fn def)")
      (COND
        ((NEQ FN (CAR DEF))
         (ERROR ' "filemap does not agree with contents of" (FULLNAME STREAM)
                T)))
      (RETURN DEF])

```

(BLOCKCOMPILE
[LAMBDA (BLKNAME BLKFNS ENTRIES FLG) ; Edited 6-Dec-86 03:59 by lmm
(RESETLST

```

[PROG ((NLAMA NLAMA)
      (NLAML NLAML)
      (LAMS LAMS)
      (LAMA LAMA)
      (NOFIXFNSLST NOFIXFNSLST)
      (NOFIXVARSLST NOFIXVARSLST)
      (EXPRSLST NIL)
      (LOCALVARS T)
      (SPECVARS SYSSPECVARS))

```



```
[SETQ COREFLG (MAPCAR CALLTAGS (FUNCTION (LAMBDA (X)
                                         (CONS (CAR X)
                                               (CADDR X]
                                         ; for use by compileuserfn, so can tell which functions are core
                                         ; and which are from on the file

[SETQ TEM (COND
          (BYTECOMPFLG
           ;; rrb dont know who uses COREFLG but need the room for the byte compiler that the definitions are taking up
           ;; so reset it here.
           (BYTEBLOCKCOMPILE2 BLKNAME CALLTAGS ENTRIES))
          (T (BLOCKCOMPILE2 BLKNAME CALLTAGS ENTRIES))
          (COND
           (STRF ; Store and redefine
            (AND (NOT (FMEMB BLKNAME LINKEDFNS))
                 (SETQ LINKEDFNS (CONS BLKNAME LINKEDFNS)))
            [MAPC COREFLG (FUNCTION (LAMBDA (X)
                                   (COND
                                    ((EXPRP (CAR X))
                                     (SAVEDEF (CAR X))
                                     (/PUTD (CAR X))
                                     (SETQ EXPRSLST (CONS (CAR X)
                                                         EXPRSLST))
                                    ;; so that if this function appears more than once in block declaration, will be compiled
                                    ;; more than once
                                    ]
                                   )
                                   ;; All of the entries would now be compiled. the other function should have their definitions be removed from definition cell, so
                                   ;; that subsequent recompile will do the right thing.

            (RETURN (OR TEM BLKNAME]))
          ))
```

(COMPSET

```
[LAMBDA (FILE FLG) ; Edited 24-Sep-2023 13:59 by rmk
                    (* bvm%: "2-Aug-86 16:58")
```

;; If FILE is not NIL, COMPSET doesn't ask any questions but simply initializes the output FILE, LCFIL. If FLG is T (AND FILE IS NIL) COMPSET
 ;; doesn't ask for an output FILE, but does set up LAPFLG, STRF, SVFLG, and LSTFIL. --- BCOMPL and BRECOMPILE both call COMPSET
 ;; twice, once with FILE NIL and FLG T, and once with FILE set to their output FILE. --- COMPILE calls COMPSET only once, with both arguments
 ;; NIL.

```
(PROG (OLDO)
      (COND
       (FILE (GO NT)))
      [SELECTQ [SETQ FILE (COMPSETREAD ' "listing? " COMPSETKEYLST (OR FLG ' (S T %
      ]
      (S [COND
        (LAPFLG (PRIN1 ' "file: " T)
                (SETQ LSTFIL (COMPSETF (COMPSETREAD)
                (GO NOCHANGE))
        ((ST STF)
         (SETQ LAPFLG NIL)
         (SETQ STRF T)
         (SETQ SVFLG (EQ FILE 'ST))
         (GO NOCHANGE))
        (F (SETQ LAPFLG NIL)
           (SETQ STRF NIL)
           (SETQ SVFLG NIL)
           (GO NOCHANGE))
        (COND
         ((SETQ LAPFLG (COMPSETY FILE))
          (SELECTQ FILE
                   ((Y YES 1 2)
                    (PRIN1 ' "file: " T)
                    (SETQ FILE (COMPSETREAD)))
                    NIL)
          (SETQ LSTFIL (COMPSETF FILE)
          [COND
           ([SETQ STRF (COMPSETY (COMPSETREAD ' "redefine? ")
            (SETQ SVFLG (COMPSETY (COMPSETREAD ' "save exprs? ")
            NOCHANGE
            (COND
             ([AND LAPFLG (NEQ LSTFIL 'T)
              (NOT (OPENP LSTFIL 'OUTPUT)
              [SETQ LSTFIL1 (SETQ LSTFIL (OPENSTREAM LSTFIL 'OUTPUT 'NEW ' ((TYPE TEXT)
              ;; LSTFIL1 is set when the file is opened for this compilation. in this case it will be closed when the compilation is finished or aborttd.
              )
              (T (SETQ LSTFIL1 NIL)))
            (COND
             ([AND (NULL FLG)
              (COMPSETY (COMPSETREAD ' "output file? " NIL ' (N %
              (PRIN1 ' "file name: " T)
              (SETQ FILE (COMPSETREAD)))
```

```

(T (SETQ FILE NIL))
NT [COND
  ((AND (SETQ LCFIL (COMPSETF FILE))
        (NEQ LCFIL T))
   (SETQ LCFIL (OR (OPENP LCFIL 'OUTPUT)
                   (OPENSTREAM LCFIL 'OUTPUT 'NEW ' ((TYPE BINARY]
   (RETURN 'DONE]))

```

(COMPSETREAD

```

[LAMBDA (MESS KEYLST DEFAULT) (* wt%: "23-AUG-80 01:29")
  (PROG (X)
    (COND
      ((OR (NULL DWIMFLG)
           (AND READBUF (SETQ READBUF (LISPXREADBUF READBUF)))
           (NULL MESS))
       (AND MESS (PRIN1 MESS T))
       (SETQ X (LISPXREAD T))
       (AND (NULL REREADFLG)
            (EQ (PEEK T)
                '%
              )
            (READC T))
      )
    (T (SETQ REREADFLG NIL)
       (SETQ X (ASKUSER DWIMWAIT DEFAULT MESS (OR KEYLST COMPSETDEFAULTKEYLST)
                  T T))
    )
  )

```

(* so that askuser doesnt echo the carriage return again when it is called for next question.)

(* COMPSETDEFAULTKEYLST is a Y or N list with conformation required. user can make no confirmation by simply setting it to NIL, and letting askuser default to its list.)

```

))
(RETURN (COND
  ((NULL LISPXHISTORY)
   X)
  (REREADFLG (PRINT X T T))
  (T (NCONC (CAAR LISPXHISTORY)
            (LIST HISTSTR0 X))
   X))

```

(COMPSETY

```

[LAMBDA (A)
  (SELECTQ A
    ((Y YES)
     T)
    ((N NO)
     NIL)
  A)]

```

(COMPSETF

```

[LAMBDA (A)
  (SELECTQ A
    (T T)
    (N NIL)
  A)]

```

(* lmm " 5-NOV-82 00:13")

(RCOMP3

```

[LAMBDA (FN DEF)
  (PROG (TYPE TEM1 TEM2)
    (SELECTQ (SETQ TYPE (ARGTYPE DEF))
      (NIL)
      (1 (COND
          ((NOT (MEMB FN NLAML))
           (SETQ NLAML (CONS FN NLAML))
           (SETQ TEM1 'NLAML)
           (GO ERROR))
          ([MEMB FN (CL:SYMBOL-VALUE (SETQ TEM1 'NLAMA)
                                     (GO ERROR1))]))
      (3 (COND
          ((NOT (MEMB FN NLAMA))
           (SETQ NLAMA (CONS FN NLAMA))
           (SETQ TEM1 'NLAMA)
           (GO ERROR))
          ([MEMB FN (CL:SYMBOL-VALUE (SETQ TEM1 'NLAML)
                                     (GO ERROR1))]))
      ((0 2)
       [COND
        ([OR [MEMB FN (CL:SYMBOL-VALUE (SETQ TEM1 'NLAMA)
                                       (MEMB FN (CL:SYMBOL-VALUE (SETQ TEM1 'NLAML)
                                                                 (GO ERROR1))
                                       (NEQ (ARGTYPE FN)

```

; Edited 18-Dec-86 19:34 by Pavel

```

      TYPE)
      ;; Situation can occur when TCOMPLING a file which contains a LAMBDA definition for a function, but for which the
      ;; incore definition is an NLAMBDA.
      (SETQ LAMS (CONS FN LAMS])
      (HELP))
      (RETURN FN)
      ERROR1
      (/SET TEM1 (REMOVE FN (CL:SYMBOL-VALUE TEM1)))
      (SETQ TEM2 " was incorrectly on ")
      ERROR
      (PRIN1 ' "***note: " COUTFILE)
      (PRIN2 FN COUTFILE T)
      (PRIN1 (OR TEM2 ' " was not on " )
             COUTFILE)
      (PRINT TEM1 COUTFILE)
      (RETURN FN])

```

```

(TCOMPL
 [LAMBDA (FILES)
  (COMPSET NIL ' (F %
))
  (for FILE inside FILES collect (OR (CAR (ERSETQ (BCOMPL FILE NIL T T)))
                                     (CONS FILE '(not compiled))
                                     (* Imm "19-NOV-82 12:07")

```

```

(RECOMPILE
 [LAMBDA (PFILE CFIL FNS)
  (BRECOMPILE PFILE CFIL FNS T])

```

```

(RECOMP?
 [LAMBDA (X FNS)
  (SELECTQ FNS
   (ALL T)
   (CHANGES (FMEMB X CHANGES))
   ((T EXPRS)
    (OR (MEMB X EXPRSLST)
        (EXPRP (OR (GETPROP X 'ADVISED)
                   (GETPROP X 'BROKEN)
                   X))))
   (COND
    ((NLISTP FNS)
     (EQ X FNS))
    (T (FMEMB X FNS]))
  (* rmk%: "24-MAY-82 21:14")

```

```

(COMPILE
 [LAMBDA (X FLG)
  ; Edited 11-Jan-88 14:51 by jds

```

;; Compile a list of functions, optionally storing them on a file.

```

(RESETLST
 (PROG ( (NLAMA NLAMA)
        (NLAML NLAML)
        (LAMS LAMS)
        (LAMA LAMA)
        (NOFIXFNSLST NOFIXFNSLST)
        (NOFIXVARSLST NOFIXVARSLST)
        (SPECVARS SPECVARS)
        (LOCALVARS LOCALVARS))
  (DECLARE (SPECVARS NLAMA NLAML LAMS LAMA NOFIXFNSLST NOFIXVARSLST SPECVARS LOCALVARS))
  (COMPSET)
  (COND
   (LCFIL ;; We're saving the results on a file, so put out the file header...
    (PRINT-COMPILE-HEADER X 'compile'd NIL))
   (WITH-READER-ENVIRONMENT *OLD-INTERLISP-READ-ENVIRONMENT*
    ;; So that the file is printed with the correct reader environment, so it'll read back in.
    [SETQ X (MAPCAR (COND
                    ((ATOM X)
                     (LIST X))
                    (T X))
                   (FUNCTION (LAMBDA (FN)
                              (COMPILE1 FN (VIRGINFN FN T)
                              T))
                    (COND
                     ((AND (NULL FLG)
                          LCFIL)
                      (PRINT NIL LCFIL FILERDTBL)
                      (CLOSEF LCFIL)))
                    (COND
                     ((AND (NULL FLG)
                          LAPFLG LSTFIL)
                      (CLOSEF LSTFIL)))
                    (RETURN X)) ]))

```

(COMPILE1

```
[LAMBDA (FN DEF COREFLG)
  ;; COREFLG is used by COMPILE1A to reset NOSPELLFLG so that spelling correction not attempted when DWIMIFYING definitions from the file.
  ;; COREFLG IS ALSO USED BY COMPILEUSERFN FOR SAME PURPOSE
  (SETQ DEF (COMPILE1A FN DEF COREFLG))
  (PROG ((FREEVARS FREEVARS))
    (RETURN (COND
      (BYTECOMPFLG (BYTECOMPILE2 FN DEF))
      (T (COMPILE2 FN DEF))
    )
  )
]; Edited 23-Nov-86 17:00 by Pavel
```

(COMPILE1A

```
[LAMBDA (FN DEF COREFLG)
  (COND
    [(EXPRP DEF)
     (PROG NIL
      ;; Used by compile1 and blockcompile1. dwimifies def where appropriate and also checks to see if it has a remote clisptranslation (e.g. for
      ;; sri qlisp.)
      [COND
        ((OR DWIMIFYCOMPFLG (SHOULD-BE-DWIMIFIED? DEF))
         ;; Needs to be dwimified. Tell them it's happening, and do it.
         (CL:FORMAT (\GETSTREAM COUTFILE 'OUTPUT)
                    "(dwimifying ~S)~%%" FN)
          (LET ((NOSPELLFLG (OR NOSPELLFLG (NULL COREFLG)))
                (FILEPKGFLG (AND FILEPKGFLG COREFLG)))
            (SETQ NOFIXFNSLST0 NOFIXFNSLST)
            (SETQ NOFIXVARSLST0 NOFIXVARSLST)
            (DWIMIFY0 DEF FN)
            (COND
              ((TAILP NOFIXFNSLST NOFIXFNSLST0)
               (SETQ NOFIXFNSLST NOFIXFNSLST0))
              (COND
                ((TAILP NOFIXVARSLST NOFIXVARSLST0)
                 (SETQ NOFIXVARSLST NOFIXVARSLST0)
                )
              )
            )
            (AND (NEQ (POSITION COUTFILE)
                     0)
                 (TERPRI COUTFILE))
            (RETURN (COND
              ((AND CLISPTRANFLG (EQ (CAR DEF)
                                     CLISPTRANFLG))
               (CADR DEF))
              ((AND CLISPARRAY (GETHASH DEF CLISPARRAY))
               (T DEF))
            )
          )
        )
      ]
    )
  ]
]; Edited 20-Jan-88 17:48 by jds
```

(SHOULD-BE-DWIMIFIED?

```
[LAMBDA (LAMBDA-FORM)
  ;; Check to see if this LAMBDA-FORM contains a DECLARATIONS: comment or the first form is a CLISP: call. Such functions are to be automatically
  ;; dwimified.
  (FOR FORM IN (CDDR LAMBDA-FORM) WHILE (EQ (CAR FORM)
                                             '*))
  DO (IF (EQ (CADR FORM)
             'DECLARATIONS%)
        THEN (RETURN T))
  FINALLY (RETURN (EQ (CAR FORM)
                      'CLISP%:))
]; Edited 10-Apr-87 13:14 by Pavel
```

(COMPEM

```
[LAMBDA (X Y ERRORFLG FL)
  (AND (NULL FL)
        (SETQ FL COUTFILE))
  (COND
    [(NULL ERRORFLG)
     (PRIN1 ' "
      *****" FL)
     (PRIN1 X FL T)
     (COND
       (Y (SPACES 1 FL)
           (PRIN1 Y FL)))
     (TERPRI FL)
     (COND
       ((NEQ FL T)
        (COMPEM X Y NIL T))
       (T (PRIN1 ' "*****" T)
           (ERROR X Y T]))
    )
  ]
]; Edited 7-JUL-78 13:07"
(* ERRORFLG is NIL when called from COMP.
Just prints X and goes on.)
(* so error message printed both places)
```

(GETCFILE

(* bvm%: " 2-Aug-86 17:13")

```
[LAMBDA (FILES CFILE)
  (PROG (X STR)
    RETRY
```

```
  (COND
    ([NLSETQ (SETQ X (OPENSTREAM CFILE 'INPUT 'OLD]
```

(* The reason it is done this way instead of with an INFILEP is that the user may have specified corrective action when INFILE fails via ERRORFNS, e.g. check another directory, spelling correct, etc.)

```
  (COND
    ((RANDACCESSP X)
     (RETURN X))
    (CLOSEF X)
    (SETQ STR "is not a random access file,")
    (T (SETQ STR "not found,")))
  (TERPRI T)
  (COND
```

```
    ((EQ (ASKUSER DWIMWAIT 'Y (LIST (if X
                                     then (FULLNAME X)
                                     else CFILE)
                                     STR " compile all functions on " (FULLNAME (CAR FILES))
                                     ' "instead"))
      'Y)
     (RETURN)))
  (COND
```

```
    ((EQ [ASKUSER DWIMWAIT 'Y (LIST ' "Shall I just forget about compiling"
                                     (MAPCAR FILES (FUNCTION FULLNAME]
```

```
      'Y)
       [COND
        ((OR (EQ (CAR READBUF)
                 'ST)
              (EQ (CAR READBUF)
                 'F))
         (SETQ READBUF (CDR READBUF)
                 (ERROR!)))
        (PRIN1 ' "Then what shall I use for CFILE ? " T)
        (SETQ CFILE (READ T T))
        (GO RETRY])
```

(* E.g. From CLEANUP.)

(SPECVARS

(* Imm " 8-APR-82 21:49")

```
[NLAMBDA A
  (SETQ SPECVARS (COND
    ((LISTP A)
     (COND
      ((LISTP SPECVARS)
       (APPEND A SPECVARS))
      ((EQ SPECVARS T)
       T)
      (T A)))
    (T (SETQ LOCALVARS (UNION (LISTP LOCALVARS)
                              SYSLOCALVARS))
      T]))
```

(LOCALVARS

(* Imm " 8-APR-82 21:49")

```
[NLAMBDA A
  (SETQ LOCALVARS (COND
    ((LISTP A)
     (COND
      ((LISTP LOCALVARS)
       (APPEND A LOCALVARS))
      ((EQ LOCALVARS T)
       T)
      (T A)))
    (T (SETQ SPECVARS (UNION (LISTP SPECVARS)
                              SYSSPECVARS))
      T)))
  NIL])
```

(GLOBALVARS

; Edited 4-Dec-86 15:25 by Pavel

```
[NLAMBDA A
  (IF (LISTP A)
    THEN (SETQ GLOBALVARS (UNION A GLOBALVARS]))
  )
```

(ADDTOVAR **NOLINKFNS** HELP ERRORX ERRORSET EVALV FAULTEVAL INTERRUPT SEARCHPDL MAPDL BREAK1 EDITE EDITL)

(ADDTOVAR **LINKFNS**)

(ADDTOVAR **FREEVARS**)


```

(ADDTOVAR SYSSPECVARS HELPCLOCK LISPXHIST RESETSTATE OLDVALUE UNDOSE0 SPECVARS LOCALVARS GLOBALVARS)
(ADDTOVAR SYSLOCALVARS )
(ADDTOVAR LOCALFREEVARS )
(ADDTOVAR BLKLIBRARY )
(ADDTOVAR RETFNS )
(ADDTOVAR BLKAPPLYFNS )
(ADDTOVAR DONTCOMPILEFNS )
(ADDTOVAR NLAML )
(ADDTOVAR NLAMA )
(ADDTOVAR LAMS )
(ADDTOVAR LAMA )
(RPAQ? SPECVARS T)
(RPAQ? LOCALVARS SYSLOCALVARS)
(RPAQ? DWIMIFYCOMPFLG NIL)
(RPAQ? COMPILEHEADER "compiled on ")
(RPAQ? COMPSETLST '(ST F STF S Y N 1 2 NIL T))
(RPAQ? COMPSETKEYLST '((ST "ore and redefine " KEYLST (" (F . "orget exprs"))
(S . "ame as last time")
(F . "ile only")
(T . "o terminal")
(1)
(2)
(Y . "es")
(N . "o")))
(RPAQ? COMPSETDEFAULTKEYLST '((Y . "es")
(N . "o")))
(RPAQ? BCOMPL.SCRATCH '{CORE}BCOMPL.SCRATCH)
(RPAQ? RECOMPILEDEFAULT 'CHANGES)
(RPAQ? COUFILE T)
(RPAQ? SVFLG T)
(RPAQ? STRF T)
(RPAQ? LSTFIL T)
(RPAQ? LCFIL )
(RPAQ? LAPFLG T)
(DECLARE%: DONTCOPY
(DECLARE%: EVAL@COMPILE
(RECORD COMPFILEDESCR (COMPFILESTREAM COMPFILEENV COMPFILEMAP . COMPFILEXPRS)
)
(DECLARE%: EVAL@COMPILE
(PUTPROPS DIGITCHARP MACRO [LAMBDA (CHAR)
(AND (IGE Q CHAR (CHARCODE 0))
(ILE Q CHAR (CHARCODE 9))
)
(DECLARE%: DOEVAL@COMPILE DONTCOPY
(GLOBALVARS SYSSPECVARS SYSLOCALVARS RECOMPILEDEFAULT COMPILE.EXT NOTCOMPILEDFILES BYTECOMPFLG COMPILEHEADER
COMPVERSION BCOMPL.SCRATCH LINKEDFNS NOFIXVARSLST0 NOFIXFNSLST0 CLISPTRANFLG CLISPARRAY COMPSETKEYLST
REREADFLG HISTSTR0 LISPXHISTORY COMPSETDEFAULTKEYLST FILERDTBL DWIMFLG DWIMWAIT)
)
)
(MOVD? 'NILL 'FILECHANGES)
(CL:PROCLAIM '(CL:SPECIAL COMPVARMACROHASH))
(CL:PROCLAIM '(GLOBAL SYSSPECVARS SYSLOCALVARS COMPILE.EXT NOTCOMPILEDFILES CLISPARRAY FILERDTBL DWIMFLG

```

DWIMWAIT LISPXHISTORY))

:: COMPILEMODE

(PUTPROPS **COMPILEMODELST** **VARTYPE** ALIST)

(DEFINEQ

COMPILEMODE

[LAMBDA (MODE)

(* Imm%: "22-JUL-77 03:53")

(* returns current compile mode. If given a mode (one of ALTO MAXC or PDP10) looks it up on COMPILEMODELST and sets values appropriately.)

(PROG1 COMPILEMODE

(COND

(MODE [MAPC [CDR (OR (ASSOC MODE COMPILEMODELST)
(ERROR MODE '?]

(FUNCTION (LAMBDA (X)

(COND

((LISTP (CAR X))

(EVAL (CAR X)))

(T (SET (CAR X)

(CDR X]

(SETQ COMPILEMODE MODE))))])

)

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

(ADDTOVAR **NLAMA** GLOBALVARS LOCALVARS SPECVARS BLOCK%:)

(ADDTOVAR **NLAML** BCOMPL3)

(ADDTOVAR **LAMA**)

)

(PUTPROPS **COMPILE COPYRIGHT** ("Venue & Xerox Corporation" T 1984 1985 1986 1987 1988 1989 1990 2021))

FUNCTION INDEX

BCOMPL	1	BRECOMPILE	6	COMPILEMODE	18	PRINT-COMPILE-HEADER	3
BCOMPL.BODY	2	BRECOMPILE1	9	COMPSET	12	RCOMP3	13
BCOMPL1A	3	BRECOMPILE2	10	COMPSETF	13	RECOMP?	14
BCOMPL2	4	BRECOMPILE3	10	COMPSETREAD	13	RECOMPILE	14
BCOMPL3	6	COMPSEM	15	COMPSETY	13	RESETOPENFILES	3
BLOCK%:	6	COMPILE	14	GETCFILE	16	SHOULD-BE-DWIMIFIED?	15
BLOCKCOMPILE	10	COMPILE1	15	GLOBALVARS	16	SPECVARS	16
BLOCKCOMPILE1	11	COMPILE1A	15	LOCALVARS	16	TCOMPL	14

VARIABLE INDEX

BCOMPL.SCRATCH	17	COMPSETLST	17	LAPFLG	17	RECOMPILEDEFAULT	17
BLKAPPLYFNS	17	COUTFILE	17	LCFIL	17	REFNS	17
BLKLIBRARY	17	DONTCOMPILEFNS	17	LINKFNS	16	STRF	17
COMPILEHEADER	17	DWIMIFYCOMPFLG	17	LOCALFREEVARS	17	SVFLG	17
COMPSETDEFAULTKEYLST	17	FREEVARS	16	LSTFIL	17	SYSLOCALVARS	17
COMPSETKEYLST	17	LAMS	17	NOLINKFNS	16	SYSSPECVARS	17

PROPERTY INDEX

COMPILEMODELST	18
----------------------	----

MACRO INDEX

DIGITCHARP	17
------------------	----

RECORD INDEX

COMPFILEDESCR	17
---------------------	----
