*File created:* 8-Dec-91 15:13:17 **{DSK}<users>sybalsky>PUBS>IMTOOLS.;2**

*changes to:* (VARS IMTOOLSCOMS)
(FNS GRAB.IMPTR PRINT.INDEX.OBJECT REF.TO.PAGE)

*previous date:* 4-Sep-91 18:04:53 {DSK}<users>sybalsky>PUBS>IMTOOLS.;1

*Read Table:* INTERLISP

*Package:* INTERLISP

*Format:* XCCS

```
;;
;; Copyright (c) 1985, 1986, 1987, 1988, 1991 by Xerox Corporation.  All rights reserved.

(RPAQQ IMTOOLSCOMS
       (                                                    ; Maintainance Functions: generating indexes, etc
        (FNS ADD.IM.MENU COLLECT.MODIFIED.IM.CHAPTERS DO.INDEX DO.MANUAL GRAB.IM.MANUAL.PTRS GRAB.IMPTR
             IM.COMMAND.MENU IM.TEDIT.SELECTION INIT.INDEX.VARS INSERT.CHARS.AROUND.SEL
             INTERPRET.IM.MENU.COMMAND MAKE.IM.TITLE MAKE.IM.TOC NEWTO PRINT.INDEX.OBJECT PRINT.INDEX.SUBREFS
             PROCESS.IM.CHAPTERS REF.TO.PAGE REFS.TO.PAGES)
        (COMS (FNS MAKE.IM.INDEX))
        (VARS IM.MANUAL.CHAPTERS IM.MANUAL.VOLUMES)
        (VARS (IM.MANUAL.DIRECTORY '"{PELE:}<DOC>IRM>"))
        (FILES IMTEDIT FREEMENU)))

;; Maintainance Functions: generating indexes, etc

(DEFINEQ

(ADD.IM.MENU
  [LAMBDA (WINDOW)                                          (* mjs "12-Jul-85 15:20")
    (ATTACHMENU (create MENU
                        ITEMS _ '(fn var arg lisp FnDef VarDef Text Name Args lispcode U-CASE CLOSE)
                        WHENSELECTEDFN _ (FUNCTION INTERPRET.IM.MENU.COMMAND))
          WINDOW
          'RIGHT
          'TOP])

(COLLECT.MODIFIED.IM.CHAPTERS
  [LAMBDA NIL                                               (* mjs "29-Jul-85 17:09")
                                                            (* check which chapters have been updated since they were last
    processed)
    (for CHAPTER.FILE.LIST in IM.MANUAL.CHAPTERS
       when (PROG ((IPFILE (FINDFILE (PACKFILENAME 'EXTENSION 'IP 'BODY (CADR CHAPTER.FILE.LIST))
                                     T))
                   (IMPTRFILE (FINDFILE (PACKFILENAME 'EXTENSION 'IMPTR 'BODY (CADR CHAPTER.FILE.LIST))
                                        T))
                   EARLIEST.IDATE)
                  (printout T "checking " (CADR CHAPTER.FILE.LIST)
                            "...")
                  (if (OR (NULL IPFILE)
                          (NULL IMPTRFILE))
                      then                                  (* process if IP or IMPTR file doesn't exist)
                           (printout T "will reprocess" T)
                           (RETURN T))
                  [SETQ EARLIEST.IDATE (IMIN (GETFILEINFO IPFILE 'ICREATIONDATE)
                                             (GETFILEINFO IMPTRFILE 'ICREATIONDATE]
                  (if (for IMFILE in (CDR CHAPTER.FILE.LIST) bind FULLFILE
                         thereis (AND (SETQ FULLFILE (FINDFILE (PACKFILENAME 'EXTENSION 'IM 'BODY IMFILE)
                                                               T))
                                      (IGREATERP (GETFILEINFO FULLFILE 'ICREATIONDATE)
                                                 EARLIEST.IDATE)))
                      then (printout T "will reprocess" T)
                           (RETURN T)
                    else (printout T "OK" T)
                         (RETURN NIL)))
       collect (LIST (CAR CHAPTER.FILE.LIST)
                     (CADR CHAPTER.FILE.LIST])

(DO.INDEX
  [LAMBDA NIL                                               (* mjs "23-Sep-85 14:10")
    (printout T "creating Index" T)
    (MAKE.IM.INDEX '{DSK}ChapIndex.IP IM.MANUAL.VOLUMES)
    [if (INFILEP '{DSK}ChapIndex.IP)
        then (RENAMEFILE '{DSK}ChapIndex.IP (PACK* IM.MANUAL.DIRECTORY 'ChapIndex.IP]
    (for X in (CONS '(Master)
                    IM.MANUAL.VOLUMES)
       bind LOCALFILE REMOTEFILE do (SETQ LOCALFILE (PACK* '{DSK}ChapTOC- (CAR X)
                                                           '.IP))
                                    (SETQ REMOTEFILE (PACK* IM.MANUAL.DIRECTORY 'ChapTOC- (CAR X)
                                                            '.IP))
                                    (printout T "creating TOC for vol: " (CAR X)
```

```
                                        T)
                                    (MAKE.IM.TOC LOCALFILE (CDR X))
                                    (if (INFILEP LOCALFILE)
                                        then (RENAMEFILE LOCALFILE REMOTEFILE)))
    (for X in IM.MANUAL.CHAPTERS do (printout T "making TOC for chapter: " X T)
                                    (MAKE.IM.TOC (PACK* IM.MANUAL.DIRECTORY (CADR X)
                                                        '-TOC.IP)
                                                 (CAR X)))
    (for X in '((I "Volume I: Language" 3101272 "October, 1985")
                (II "Volume II: Environment" 3101273 "October, 1985")
                (III "Volume III: Input/Output" 3101274 "October, 1985"))
       do (MAKE.IM.TITLE (PACK* IM.MANUAL.DIRECTORY "ChapTitle-" (CAR X)
                                '.IP)
                         (CADR X)
                         (CADDR X)
                         (CADDDR X])
```

## (DO.MANUAL

```
  [LAMBDA (CHAPNAMES MAKE.INDEX.FLG GET.REFS NO.IP.FLG)          (* mjs "12-Oct-85 14:18")
    (PROG ((IM.NOTE.FLG NIL)
           (IM.CHECK.DEFS NIL)
           (IM.REF.FLG NIL)
           (IM.INDEX.FILE.FLG T)
           (IM.DRAFT.FLG NIL)
           (IM.EVEN.FLG T))
          (CNDIR '{DSK})
          (if GET.REFS
              then (INIT.INDEX.VARS)
                   (GRAB.IM.MANUAL.PTRS)
                   (SETQ IM.REF.FLG T))
          (PROCESS.IM.CHAPTERS CHAPNAMES NO.IP.FLG)
          (if MAKE.INDEX.FLG
              then (INIT.INDEX.VARS)
                   (GRAB.IM.MANUAL.PTRS)
                   (DO.INDEX])
```

## (GRAB.IM.MANUAL.PTRS

```
  [LAMBDA NIL                                                    (* mjs "25-Sep-85 15:47")
    (INIT.INDEX.VARS)
    (for X in IM.MANUAL.CHAPTERS unless (MEMB (U-CASE (CADR X))
                                              '(CHAPACK CHAT))
       do (GRAB.IMPTR (PACKFILENAME 'BODY (CADR X)
                                    'BODY IM.MANUAL.DIRECTORY])
```

## (GRAB.IMPTR

```
  [LAMBDA (IMPTR.FILE.NAME KEEP.IMPLICIT.REFS.FLG IMPTR.TYPES)   ; Edited 8-Dec-91 14:20 by jds
    ;; Reads the IMPTR file IMPTR.FILE.NAME, and puts the refs in the variables IMPTR.HASH IMPTR.TOC.LIST IMPTR.NAME.LIST.  If
    ;; KEEP.IMPLICIT.REFS.FLG is non-NIL, implicit references are kept.  If IMPTR.TYPES is non-NIL, it should be a list of reference types <as lists>
    ;; that should be retrieved.  For example 'IMPTR.TYPES=((Function)(Variable))' to retrieve only function and variable references.
    (PROG ((PTRFILE (OPENSTREAM (FINDFILE (PACKFILENAME 'BODY (U-CASE IMPTR.FILE.NAME)
                                                        'EXTENSION
                                                        'IMPTR)
                                          T)
                                'INPUT
                                'OLD))
           DATUM)
          (COND
             (KEEP.IMPLICIT.REFS.FLG (printout T T "  ***  WARNING:  implicit references will be included  ***"
                                           T T)))
          (COND
             ((NOT (BOUNDP 'IMPTR.HASH))
              (INIT.INDEX.VARS)))
          [while [SETQ DATUM (CAR (NLSETQ (READ PTRFILE]
             do (COND
                   ((NOT (type? IM.INDEX.DATA DATUM))
                    (printout T "bad datum -- " DATUM T))
                   ([MEMBER (MKLIST (fetch (IM.INDEX.DATA TYPE) of DATUM))
                            '((CHAPTER)
                              (SUBSEC)
                              (APPENDIX]
                    ;; map (CHAPTER) => CHAPTER.  We want to allow these words in a list, so that users can create TOC entries using
                    ;; IMINDEX.
                    [replace (IM.INDEX.DATA TYPE) of DATUM with (CAR (MKLIST (fetch (IM.INDEX.DATA TYPE)
                                                                                 of DATUM]
                    (SETQ IMPTR.TOC.LIST (CONS DATUM IMPTR.TOC.LIST)))
                   ((AND (NOT KEEP.IMPLICIT.REFS.FLG)
                         (MEMB '*IMPLICIT* (fetch (IM.INDEX.DATA INFO) of DATUM)))
                                                                  ; unless KEEP.IMPLICIT.REFS.FLG is true, flush implicit
                                                                  ; references
                    NIL)
                   ((AND IMPTR.TYPES (NOT (MEMBER (fetch (IM.INDEX.DATA TYPE) of DATUM)
                                                  IMPTR.TYPES)))  ; if IMPTR.TYPES is non-NIL, flush references that do not have
                                                                  ; one of these types.
```

```
                    NIL)
                   (T [COND
                         ((NULL (fetch (IM.INDEX.DATA TYPE) of DATUM))
                          (replace (IM.INDEX.DATA TYPE) of DATUM with 'TERM]
                      [COND
                         ((NOT (GETHASH (fetch (IM.INDEX.DATA NAME) of DATUM)
                                        IMPTR.HASH))
                          (SETQ IMPTR.NAME.LIST (CONS (fetch (IM.INDEX.DATA NAME) of DATUM)
                                                      IMPTR.NAME.LIST]
                      (PUTHASH (fetch (IM.INDEX.DATA NAME) of DATUM)
                             (CONS DATUM (GETHASH (fetch (IM.INDEX.DATA NAME) of DATUM)
                                                  IMPTR.HASH))
                             IMPTR.HASH]
          (CLOSEF PTRFILE])
```

## (IM.COMMAND.MENU

```
(* mjs "24-Jul-85 17:02")
[LAMBDA (MENU.POS)
  (ADDMENU (create MENU
                   ITEMS _ '[(|FORMAT TEDIT SELECTION| (IM.TEDIT.SELECTION))
                            (|Add IM Menu| (ADD.IM.MENU (WHICHW (GETPOSITION]
                   TITLE _ "IM commands")
           NIL
           (if MENU.POS
             else (GETPOSITION])
```

## (IM.TEDIT.SELECTION

```
(* mjs "30-Jul-85 12:01")
[LAMBDA NIL
  (PROG ((TEDITW (WHICHW (GETPOSITION)))
         TSTREAM TSTREAMSEL SELSTREAM FIRSTCHAR ENDCHAR)        (* make sure that error window won't pagehold)
        (WINDOWPROP (WFROMDS (TTYDISPLAYSTREAM))
               'PAGEFULLFN
               (FUNCTION NILL))
        (if [OR (NULL TEDITW)
                (NULL (WINDOWPROP TEDITW 'TEXTSTREAM]
            then (RETURN))
        (SETQ TSTREAM (TEXTSTREAM TEDITW))
        (SETQ TSTREAMSEL (TEDIT.GETSEL TSTREAM))
        (SETQ SELSTREAM (OPENTEXTSTREAM))
        (SETQ FIRSTCHAR (SUB1 (fetch (SELECTION CH#) of TSTREAMSEL)))
        (SETQ ENDCHAR (IPLUS FIRSTCHAR (fetch (SELECTION DCH) of TSTREAMSEL)))
        (SETFILEPTR TSTREAM FIRSTCHAR)
        [for X from 1 to (IDIFFERENCE ENDCHAR FIRSTCHAR) bind C
           do (SETQ C (BIN TSTREAM))
              (if (SMALLP C)
                  then (BOUT SELSTREAM C)
                elseif (IMAGEOBJP C)
                  then (TEDIT.INSERT.OBJECT C SELSTREAM (ADD1 (GETFILEPTR SELSTREAM)))
                       (SETFILEPTR SELSTREAM (GETEOFPTR SELSTREAM]
        (for X in '(FORMATSELDUMMY.IM FORMATSELDUMMY.IMERR FORMATSELDUMMY.IMPTR) when (INFILEP X)
           do (printout T "deleting " X T)
              (DELFILE X))
        (TEDIT.PUT SELSTREAM 'FORMATSELDUMMY.IM)
        (TEDIT.KILL SELSTREAM)
        (TEDIT (IM.TEDIT 'FORMATSELDUMMY.IM T])
```

## (INIT.INDEX.VARS

```
(* mjs " 6-Aug-86 09:49")
[LAMBDA NIL
  (SETQ IMPTR.HASH (HASHARRAY 5000))
  (SETQ IMPTR.NAME.LIST NIL)
  (SETQ IMPTR.TOC.LIST NIL])
```

## (INSERT.CHARS.AROUND.SEL

```
(* mjs " 8-May-85 11:46")
[LAMBDA (TEXTSTREAM BEFORETEXT AFTERTEXT)
  (PROG [(FIRSTCHAR (fetch (SELECTION CH#) of (TEDIT.GETSEL TEXTSTREAM)))
         (AFTERLASTCHAR (fetch (SELECTION CHLIM) of (TEDIT.GETSEL TEXTSTREAM]
        (TEDIT.INSERT TEXTSTREAM AFTERTEXT AFTERLASTCHAR)
        (TEDIT.INSERT TEXTSTREAM BEFORETEXT FIRSTCHAR])
```

## (INTERPRET.IM.MENU.COMMAND

```
(* mjs "12-Jul-85 15:19")
[LAMBDA (ITEM MENU MOUSEKEY)
  (PROG [(TS (TEXTSTREAM (MAINWINDOW (WFROMMENU MENU]
        (if (EQ ITEM 'CLOSE)
            then (DETACHWINDOW (WFROMMENU MENU))
                 (CLOSEW (WFROMMENU MENU))
          elseif (EQ ITEM 'U-CASE)
            then (PROG [(CH# (fetch (SELECTION CH#) of (TEDIT.GETSEL TS)))
                        (NEW (U-CASE (TEDIT.SEL.AS.STRING TS]
                       (TEDIT.DELETE TS (TEDIT.GETSEL TS))
                       (TEDIT.INSERT TS NEW)
                       (TEDIT.SETSEL TS CH# (NCHARS NEW)
                              'LEFT))
          else (INSERT.CHARS.AROUND.SEL TS (CONCAT "{" ITEM (if (EQ ITEM 'Text)
```

```
                                                    then (CHARACTER (CHARCODE CR))
                                                   else " "))
                        "}"])
```

## (MAKE.IM.TITLE

```
  [LAMBDA (OUTFILE.FLG SUBTITLE DOCNUMBER DOCDATE)              (* mjs "23-Sep-85 14:05")
    (PROG ((SUBSEC.COUNT.LIST (LIST (LIST NIL "" "ii" "iii" "iv")))
           (IM.EVEN.FLG NIL))
          (DECLARE (SPECVARS SUBSEC.COUNT.LIST IM.EVEN.FLG))
          (RETURN (MAKE.IM.DOCUMENT [LIST (FUNCTION (LAMBDA NIL
                                                    (DUMPOUT FONT IM.CHAPTER.TITLE.FONT PARALOOKS
                                                           `(SPECIALX 0 SPECIALY %, IM.TITLEPAGE.TITLE.Y)
                                                           DUMP.CHARS "Interlisp-D Reference Manual" CR CR
                                                           PARALOOKS `(SPECIALX 0 SPECIALY %, (DIFFERENCE

                                                                                                IM.TITLEPAGE.TITLE.Y
                                                                                                    20))
                                                           DUMP.CHARS SUBTITLE CR CR)
                                                    (DUMPOUT FONT IM.XEROX.LOGO.FONT PARALOOKS
                                                           `(LEFTMARGIN 0 1STLEFTMARGIN 0 SPECIALX 0 SPECIALY
                                                                  %, IM.TITLEPAGE.TITLE.Y)
                                                           DUMP.CHARS "XEROX" CR CR)
                                                    (DUMPOUT FONT IM.SUBSEC.THREE.TITLE.FONT PARALOOKS
                                                           `(SPECIALX 0 SPECIALY %, IM.TITLEPAGE.DOCNUMBER.Y)
                                                           DUMP.CHARS DOCNUMBER CR CR PARALOOKS
                                                           `(SPECIALX 0 SPECIALY %, (DIFFERENCE
                                                                                      IM.TITLEPAGE.DOCNUMBER.Y
                                                                                          12))
                                                           DUMP.CHARS DOCDATE CR CR)
                                                    (DUMPOUT FONT IM.SUBSEC.THREE.TITLE.FONT PARALOOKS
                                                           `(SPECIALX 0 SPECIALY %, IM.TITLEPAGE.TITLE.Y
                                                                  NEWPAGEBEFORE T)
                                                           DUMP.CHARS "Copyright (c) 1985 Xerox Corporation"
                                                           CR CR FONT NIL DUMP.CHARS "All rights reserved." CR
                                                           CR PARALOOKS '(QUAD JUSTIFIED)
                                                           DUMP.CHARS "Portions from %"Interlisp Reference
                                                           Manual%" Copyright (c) 1983 Xerox Corporation, and
                                                           %"Interlisp Reference Manual%" Copyright (c) 1974,
                                                           1975, 1978 Bolt, Beranek & Newman and Xerox
                                                           Corporation." CR CR DUMP.CHARS "This publication
                                                           may not be reproduced or transmitted in any form by
                                                           any means, electronic, microfilm, xerography, or
                                                           otherwise, or incorporated into any information
                                                           retrieval system, without the written permission of
                                                           Xerox Corporation." CR CR]
                                 OUTFILE.FLG NIL "Hardcopy of Title Page"])
```

## (MAKE.IM.TOC

```
  [LAMBDA (OUTFILE.FLG CHAPTER.NUMBERS IMPTR.FILES)             (* mjs "12-Aug-86 09:54")

          (* * CHAPTER.NUMBERS is either%: NIL, meaning to generate TOC of ALL data available;
          a single number, meaning to generate a chapter TOC for that chapter;
          or a list of numbers, meaning to generate a TOC for those chapters)

    (PROG* ([IMPTR.DATA (if IMPTR.FILES
                            then (PROG ((IMPTR.HASH (HASHARRAY 50))
                                        (IMPTR.TOC.LIST NIL)
                                        (IMPTR.NAME.LIST NIL))
                                       (DECLARE (SPECVARS IMPTR.HASH IMPTR.TOC.LIST IMPTR.NAME.LIST))
                                       [for X in (MKLIST IMPTR.FILES) do (printout T "Grabbing IMPTR file: " X T)


          (* * the IMPTR.TYPES arg to GRAB.IMPTR is a list containing no possible types, so that no index entries will be saved.)

                                                                                  (GRAB.IMPTR X NIL '(XXXXXXXXX]
                                       (RETURN (LIST IMPTR.HASH IMPTR.TOC.LIST IMPTR.NAME.LIST]
      (IMPTR.HASH (if IMPTR.FILES
                      then (CAR IMPTR.DATA)
                    else IMPTR.HASH))
      (IMPTR.TOC.LIST (if IMPTR.FILES
                          then (CADR IMPTR.DATA)
                        else IMPTR.TOC.LIST))
      (IMPTR.NAME.LIST (if IMPTR.FILES
                           then (CADDR IMPTR.DATA)
                         else IMPTR.NAME.LIST))
      (SINGLE.CHAP.TOC.FLG (NUMBERP CHAPTER.NUMBERS))
      (SUBSEC.COUNT.LIST '(TOC))
      (MAKE.IM.TOC.TITLE (if (NULL CHAPTER.NUMBERS)
                             then "MASTER TABLE OF CONTENTS"
                           else "TABLE OF CONTENTS"))
      MAKE.IM.TOC.LIST)
     (DECLARE (SPECVARS SINGLE.CHAP.TOC.FLG SUBSEC.COUNT.LIST MAKE.IM.TOC.TITLE MAKE.IM.TOC.LIST
                        IMPTR.HASH IMPTR.TOC.LIST IMPTR.NAME.LIST))
     (SETQ MAKE.IM.TOC.LIST (if (NULL CHAPTER.NUMBERS)
                                then (APPEND IMPTR.TOC.LIST)
```

```
                                    else (for X in IMPTR.TOC.LIST bind (CHLST _ (MKLIST CHAPTER.NUMBERS))
                                            when (MEMB (CAR (LAST (fetch (IM.INDEX.DATA SUBSEC) of X)))
                                                       CHLST)
                                            collect X)))
          [SORT MAKE.IM.TOC.LIST (FUNCTION (LAMBDA (A B)
                                             (LIST.ORDER (REVERSE (fetch (IM.INDEX.DATA SUBSEC) of A))
                                                    (REVERSE (fetch (IM.INDEX.DATA SUBSEC) of B]
          (RETURN
           (MAKE.IM.DOCUMENT
            [LIST
             (FUNCTION (LAMBDA NIL
                         (DUMP.HEADERS.FOOTERS MAKE.IM.TOC.TITLE MAKE.IM.TOC.TITLE)
                         (DUMPOUT FONT IM.CHAPTER.TITLE.FONT PARALOOKS
                                '(PARALEADING 0 LINELEADING 0 QUAD LEFT TABS %, IM.RIGHT.MARGIN.TABS)
                                TAB DUMP.CHARS MAKE.IM.TOC.TITLE CR CR)
                         (DUMP.HRULE 6)
                         (DUMPOUT START.PARA FONT NIL PARALOOKS '(PARALEADING 93 LINELEADING 0)
                                DUMP.CHARS " " CR CR)
                         (for TOC.PTR on MAKE.IM.TOC.LIST bind TOC.LINE
                            do
                            (SETQ TOC.LINE (CAR TOC.PTR))
                            (PROG ((SECTION.STRING "")
                                   (SEC.LIST (REVERSE (fetch (IM.INDEX.DATA SUBSEC) of TOC.LINE)))
                                   (PAGE.LIST (REF.TO.PAGE TOC.LINE))
                                   MAJOR.SECTION.FLG LINE.PARA LINE.INDENT)
                                  (SETQ MAJOR.SECTION.FLG (EQLENGTH SEC.LIST 1))
                                  [for X on SEC.LIST do (SETQ SECTION.STRING (CONCAT SECTION.STRING (CAR X)
                                                                                     (if (CDR X)
                                                                                         then "."
                                                                                       else ""]
                                  (SETQ LINE.INDENT (if MAJOR.SECTION.FLG
                                                        then 0
                                                      elseif (EQLENGTH SEC.LIST 2)
                                                        then IM.TOC.SUBSEC.ONE.LEFTMARGIN
                                                      else IM.TOC.SUBSEC.TWO.LEFTMARGIN))
                                  (SETQ LINE.PARA (LIST '1STLEFTMARGIN LINE.INDENT 'LEFTMARGIN (PLUS
                                                                                                 LINE.INDENT
                                                                                                 72)
                                                        'QUAD
                                                        'LEFT
                                                        'TABS IM.RIGHT.MARGIN.TABS))
                                  (DUMPOUT PARALOOKS (APPEND '(HEADINGKEEP ON)
                                                             LINE.PARA)
                                         FONT
                                         (if MAJOR.SECTION.FLG
                                             then IM.SUBSEC.ONE.TITLE.FONT
                                           else IM.SUBSEC.THREE.TITLE.FONT)
                                         DUMP.CHARS SECTION.STRING DUMP.CHARS ".   " DUMP.CHARS
                                         [PROG ((SAV (fetch (IM.INDEX.DATA SAV) of TOC.LINE)))
                                               (if (NLISTP SAV)
                                                   then (SETQ SAV (CHCON SAV)))
                                               (RETURN (CONS SAV (LAST SAV]
                                         TAB FONT NIL DUMP.CHARS (CONCAT (CAR PAGE.LIST)
                                                                         "."
                                                                         (CADR PAGE.LIST))
                                         CR CR)
                                  (DUMP.HRULE
                                   1 NIL (APPEND (if [AND MAJOR.SECTION.FLG
                                                          (EQP (CAR SEC.LIST)
                                                               (CAR (REVERSE (fetch (IM.INDEX.DATA SUBSEC)
                                                                                    of (CADR TOC.PTR]
                                                 then (* if this is a major heading, and there are subheadings, prevent
                                                      a pagebreak here)
                                                      '(HEADINGKEEP ON)
                                               else NIL)
                                             LINE.PARA]
              OUTFILE.FLG NIL "Hardcopy of Table of Contents"])
```

## (NEWTO

```
  [LAMBDA (NAME)                                          (* mjs " 8-APR-83 16:43")
    (SETQ NAME (U-CASE NAME))
    (PROG ((TO.PROG.NAME (PACK* NAME "#TOPROG")))
          [COND
             ((FMEMB NAME TO.NAME.LIST)
              (printout T "$$$ WARNING --- " NAME " already defined as TO --- will redefine " TO.PROG.NAME))
             (T (SETQ TO.NAME.LIST (CONS NAME TO.NAME.LIST]
          (replace TO.PROG of NAME with TO.PROG.NAME)
          (DEFINE (LIST (CONS TO.PROG.NAME '(NIL (STANDARD.DUMMY.TO.PROG])
```

## (PRINT.INDEX.OBJECT

```
  [LAMBDA (NAM LST VOLUME.INFO SUB.LEVEL)                 ; Edited  8-Dec-91 15:10 by jds
```

;;; LST is a list of index refs

;;; VOLUME.INFO is a list of elements, where each element describes the 'name' of a given volume, and what chapters go into it.  Example: ((I 1 2 3) (II
;;; 4 5) (III 6 7 8)) means that chapters 1-3 go into volume 'I' .  If NIL, the index entries are not divided into volumes

```
;;; SUB.LEVEL is NIL if called to print a top-level index entry, 1 if called recursively for first-level subentries, or 2 for second-level subentries.

;;; for now, as the type we will just yank the type of the first index reference.  Eventually may want to find the one which looks like it has the case-info
;;; most 'correct'

    (PROG ([TYP (COND
                   ((fetch (IM.INDEX.DATA TYPE) of (CAR LST)))
                   (T 'TERM]
           DEF.REFS OTHER.DEF.REFS REF.TEXT REF SUB.REFS)
          (COND
             ((EQ TYP 'TAG)                                          ; ignore TAG index entries
              (RETURN))
             ([NOT (OR (LISTP TYP)
                       (EQ TYP 'TERM]
              (SHOULDNT "bad TYP given to PRINT.INDEX.OBJECT")))

;;; temporary: throw out all *END* references

          [SETQ LST (for X in LST collect X unless (MEMB '*END* (fetch (IM.INDEX.DATA INFO) of X]
          (COND
             ((NULL LST)
              (RETURN)))

;;; seperate out all sub-index entries now, to be processed after this the main one

          (SETQ SUB.REFS (for X in LST when [PROG ((PLIST (fetch (IM.INDEX.DATA PROPLIST) of X)))
                                                  (RETURN (OR (LISTGET PLIST 'SUBNAME)
                                                              (LISTGET PLIST 'SUBTEXT]
                                collect X))
          (SETQ LST (LDIFFERENCE LST SUB.REFS))

;;; if all refs are subrefs, generate a dummy ref

          [COND
             ((NULL LST)
              (SETQ LST (LIST (create IM.INDEX.DATA
                                      INFO _ '(*NOPAGE*)
                                      PROPLIST _ NIL using (CAR SUB.REFS]

;;; DEF.REFS is all definition references, sorted by size of description, longest one first

          [SETQ DEF.REFS (SORT (for X in LST when (MEMB '*DEF* (fetch (IM.INDEX.DATA INFO) of X)) collect X)
                               (FUNCTION (LAMBDA (A B)
                                                 (ILEQ (LENGTH (fetch (IM.INDEX.DATA SAV) of A))
                                                       (LENGTH (fetch (IM.INDEX.DATA SAV) of B]

;;; for all but the longest definition reference, print as separate index lines, and remove from index list for this line

;;; note that any other defs that print the same as the longest def are also merged into this line

          (SETQ OTHER.DEF.REFS (for X in (CDR DEF.REFS) unless (EQUAL (fetch (IM.INDEX.DATA SAV) of X)
                                                                      (fetch (IM.INDEX.DATA SAV)
                                                                         of (CAR DEF.REFS)))
                                     collect X))
          [COND
             (OTHER.DEF.REFS (PRINT.INDEX.OBJECT NAM OTHER.DEF.REFS VOLUME.INFO SUB.LEVEL)
                    (SETQ LST (LDIFFERENCE LST OTHER.DEF.REFS]

;;; get the reference text from the definition ref, if there is one, otherwise from the first primary ref with text, otherwise from any ref, otherwise use the
;;; name

          (SETQ REF.TEXT (COND
                            ((CAR DEF.REFS)
                             (fetch (IM.INDEX.DATA SAV) of (CAR DEF.REFS)))
                            ((SETQ REF (for X in LST when (MEMB '*PRIMARY* (fetch (IM.INDEX.DATA INFO)
                                                                                  of X))
                                                  thereis (fetch (IM.INDEX.DATA SAV) of X)))
                             (fetch (IM.INDEX.DATA SAV) of REF))
                            ([AND (SETQ REF (for X in LST thereis (fetch (IM.INDEX.DATA SAV) of X)))
                                  (NULL DEF.REFS)
                                  (for X in LST never (MEMB '*PRIMARY* (fetch (IM.INDEX.DATA INFO) of X]
                                                                          ; only use text from non-def,non-primary index if there are NO
                                                                          ; primary or def index entries
                             (fetch (IM.INDEX.DATA SAV) of REF))
                            (T NAM)))

;;; if REF.TEXT is not a list <either because NAM is used, or the SAV data was an atom>, put it in a list, adding {lisp...} if it is not a term

          [COND
             ((NLISTP REF.TEXT)
              (SETQ REF.TEXT (CHCON REF.TEXT))
              (COND
                 ((NEQ TYP 'TERM)
                  (SETQ REF.TEXT (CONS '(FONT . LISP)
                                       REF.TEXT]
```

```
;;; Flush all nopage refs

        (SETQ LST (for X in LST unless (MEMB '*NOPAGE* (fetch (IM.INDEX.DATA INFO) of X)) collect X))

;;; print out the text of the index entry

        (DUMPOUT FONT NIL PARALOOKS (COND
                                    ((NULL SUB.LEVEL)
                                     NIL)
                                    ((EQ SUB.LEVEL 1)
                                     (LIST '1STLEFTMARGIN IM.INDEX.SUB.1STLEFTMARGIN 'LEFTMARGIN
                                           IM.INDEX.SUB.LEFTMARGIN))
                                    (T (LIST '1STLEFTMARGIN IM.INDEX.SUBSUB.1STLEFTMARGIN 'LEFTMARGIN
                                             IM.INDEX.SUBSUB.LEFTMARGIN)))
               DUMP.CHARS
               (CONS REF.TEXT (LAST REF.TEXT))
               DUMP.CHARS "  " FONT ITALIC DUMP.CHARS (COND
                                                       ((OR (EQ (U-CASE TYP)
                                                                'TERM)
                                                            (AND (EQUAL (U-CASE TYP)
                                                                        '(FUNCTION))
                                                                 DEF.REFS))
                                                        "")
                                                       (T (MKSTRING TYP)))
               DUMP.CHARS "   ")                                ; divide the index refs between the volumes.  If none in a given
                                                                ; volume, don't include it
        [SETQ VOLUME.REF.LISTS (COND
                                (VOLUME.INFO
                                 (for VOL in VOLUME.INFO bind VOL.REFS
                                    when (SETQ VOL.REFS
                                           (for REF in LST
                                              when (MEMBER (CAR (LAST (fetch (IM.INDEX.DATA SUBSEC)
                                                                              of REF)))
                                                           (CDR VOL))
                                              collect REF))
                                    collect (CONS (CAR VOL)
                                                  VOL.REFS)))
                                (T (LIST (CONS NIL LST]
        [for VOL.REF.LIST in VOLUME.REF.LISTS bind VOL.NAME VOL.REFS PRIMARY.PAGELST PAGELST
           do (SETQ VOL.NAME (CAR VOL.REF.LIST))
              (SETQ VOL.REFS (CDR VOL.REF.LIST))
              (DUMPOUT FONT NIL DUMP.CHARS (COND
                                           ((EQ VOL.REF.LIST (CAR VOLUME.REF.LISTS))
                                            "")
                                           (T "; "))
                       DUMP.CHARS
                       (COND
                          (VOL.NAME (CONCAT VOL.NAME ": "))
                          (T "")))
              ;; PRIMARY.PAGELST is a list of all primary or definition references, sorted by chapter and page, with no duplicates
              (SETQ PRIMARY.PAGELST (for X in VOL.REFS when (OR (MEMB '*DEF* (fetch (IM.INDEX.DATA INFO)
                                                                                of X))
                                                               (MEMB '*PRIMARY* (fetch (IM.INDEX.DATA INFO)
                                                                                   of X)))
                                           collect (REF.TO.PAGE X)))
              (SETQ PRIMARY.PAGELST (INTERSECTION PRIMARY.PAGELST PRIMARY.PAGELST))
              [SETQ PRIMARY.PAGELST (SORT PRIMARY.PAGELST
                                          (FUNCTION (LAMBDA (A B)
                                                      (COND
                                                         [(AND (NUMBERP (CAR A))
                                                               (NUMBERP (CAR B)))
                                                          ;; CHAPTER is numeric, so compare with numeric
                                                          (OR (LESSP (CAR A)
                                                                     (CAR B))
                                                              (AND (EQUAL (CAR A)
                                                                          (CAR B))
                                                                   (LESSP (CADR A)
                                                                          (CADR B]
                                                         (T ;; One or t'other chapter is alpha, so compare using ALPHORDER
                                                            (OR (ALPHORDER (CAR A)
                                                                           (CAR B))
                                                                (AND (STRING-EQUAL (MKSTRING (CAR A))
                                                                                   (MKSTRING (CAR B)))
                                                                     (LESSP (CADR A)
                                                                            (CADR B]
              ;; PAGELST is: ((chap1 page1) ... (chapN pageN)) EXCEPT for primary references
              (SETQ PAGELST (for X in VOL.REFS collect (REF.TO.PAGE X)))
              (SETQ PAGELST (INTERSECTION PAGELST PAGELST))
              (SETQ PAGELST (LDIFFERENCE PAGELST PRIMARY.PAGELST))
              ;; PAGELST.BY.CHAPTER is list with elements of form: ((chap1 page1) (chap1 page1) ...) partioned by chapter, and sorted by
              ;; chapter
              (SETQ PAGELST.BY.CHAPTER (PARTITION.LIST PAGELST [FUNCTION (LAMBDA (A B)
```

```
                                                                    (STRING-EQUAL (MKSTRING A)
                                                                           (MKSTRING B]
                                                 (FUNCTION CAR)
                                                 (FUNCTION ALPHORDER)))
                    [COND
                       (PRIMARY.PAGELST (DUMPOUT FONT BOLD DUMP.CHARS
                                                 (CONCATLIST (for X in PRIMARY.PAGELST
                                                                bind (LAST.PRIMARY.PAGE _ (CAR (LAST PRIMARY.PAGELST))
                                                                     )
                                                                join (APPEND (COND
                                                                                ((CAR X)
                                                                                 (LIST (CAR X)
                                                                                       "-"))
                                                                                (T NIL))
                                                                             (LIST (CADR X)
                                                                                   (COND
                                                                                      ((OR (NOT (EQUAL X
                                                                                                       LAST.PRIMARY.PAGE
                                                                                                       ))
                                                                                           PAGELST.BY.CHAPTER)
                                                                                       "; ")
                                                                                      (T ""]
                       (for CHAP.PAGES in PAGELST.BY.CHAPTER bind CHAP.STRING NUMS
                          do [SETQ CHAP.STRING (CONCAT (COND
                                                          ((EQ CHAP.PAGES (CAR PAGELST.BY.CHAPTER))
                                                           "")
                                                          (T "; "))
                                                       (COND
                                                          ((CAAR CHAP.PAGES)
                                                           (CONCAT (CAAR CHAP.PAGES)
                                                                   "-"))
                                                          (T ""]
                             [SETQ NUMS (SORT (for X in CHAP.PAGES collect (CADR X]
                                                                                  (* (* ;; "terrible kludge for the sole purpose of merging runs of
                                                                                  page numbers, to produce index entries like '1,3-5' instead of
                                                                                  '1,3,4,5'") (bind (FIRSTNUM _ (CAR NUMS))
                                                                                  (SECONDNUM _ (CAR NUMS)) (NEWNUMS _ NIL) for X in
                                                                                  (APPEND (CDR NUMS) (QUOTE
                                                                                  (BADNUM))) do (COND ((AND (NUMBERP SECONDNUM)
                                                                                  (NOT (EQUAL X (ADD1 SECONDNUM))))
                                                                                  (push NEWNUMS (COND ((EQUAL FIRSTNUM SECONDNUM)
                             FIRSTNUM) (T (PACK* FIRSTNUM "-" SECONDNUM)))) (SETQ FIRSTNUM X))) (SETQ SECONDNUM X) finally
                                                                                  (SETQ NUMS (REVERSE NEWNUMS))))
                             (for SINGLE.PAGE in NUMS do (DUMPOUT FONT NIL DUMP.CHARS (COND
                                                                                         ((EQ SINGLE.PAGE (CAR NUMS))
                                                                                          CHAP.STRING)
                                                                                         (T ","))
                                                                                      DUMP.CHARS SINGLE.PAGE]
                    (DUMPOUT CR CR)
```

;;; print out any sub references

```
                    (PRINT.INDEX.SUBREFS SUB.REFS VOLUME.INFO SUB.LEVEL])
```

## (PRINT.INDEX.SUBREFS

```
  [LAMBDA (SUB.REFS VOLUME.INFO SUB.LEVEL)                    (* mjs "14-Jul-86 12:39")
```

;; Print sub-item references. This is how the Notecards manual's Analog-clock sub-items get printed, e.g.:

;; ANALOG CLOCK 168

;; creating 168

;; known problems 170

```
    (PROG ((NEW.SUB.LEVEL (COND
                             ((NULL SUB.LEVEL)
                              1)
                             (T 2)))
           SHIFTED.REFS)
          (COND
             ([NOT (AND SUB.REFS (MEMB SUB.LEVEL '(NIL 1]
```

;; There must be references to print, and we only print 2 levels of sub-references.

```
              (RETURN)))
```

;; Move everything in the references over: SUBTYPEW -> TYPE, SUBTEXT -> SAV, etc, and move SUBSUB... up into the SUB fields.

;; Many items were created with only SUBTEXT, so use that as the name if need be (using TERM as the type.)

```
          [SETQ SHIFTED.REFS (for X in SUB.REFS collect (PROG ((PLIST (fetch (IM.INDEX.DATA PROPLIST) of X)))
                                                               (RETURN (create IM.INDEX.DATA
                                                                               NAME _ (OR (LISTGET PLIST 'SUBNAME)
                                                                                          (LISTGET PLIST 'SUBTEXT))
                                                                               TYPE _ (OR (LISTGET PLIST 'SUBTYPE)
                                                                                          'TERM)
                                                                               SAV _ (LISTGET PLIST 'SUBTEXT)
                                                                               PROPLIST _ (LIST 'SUBNAME
                                                                                                (LISTGET PLIST
                                                                                                         'SUBSUBNAME)
```

```
                                                                                    'SUBTYPE
                                                                                    (LISTGET PLIST
                                                                                           'SUBSUBTYPE)
                                                                                    'SUBTEXT
                                                                                    (LISTGET PLIST
                                                                                           'SUBSUBTEXT))
                                                                        using X]
          (for LST in (PARTITION.LIST SHIFTED.REFS NIL [FUNCTION (LAMBDA (A)
                                                                   (CONCAT
                                                                    (if (MEMB '*NOPAGE* (fetch (IM.INDEX.DATA INFO)
                                                                                          of A))
                                                                        then "ZZZZZZZ"
                                                                      else "")
                                                                    (fetch (IM.INDEX.DATA NAME) of A]
                       (FUNCTION ALPHORDER))
             do ;; partition sort by ref name <putting *NOPAGE* refs at the end>, and for each name, handle refs
                (for X in (PARTITION.LIST LST NIL [FUNCTION (LAMBDA (A)
                                                              (fetch (IM.INDEX.DATA TYPE) of A]
                             (FUNCTION ALPHORDER))
                   do                                                ; parition sort refs by type
                      (PRINT.INDEX.OBJECT (fetch (IM.INDEX.DATA NAME) of (CAR X))
                             X VOLUME.INFO NEW.SUB.LEVEL])
```

## (PROCESS.IM.CHAPTERS

```
  [LAMBDA (CHAPNAMES NULL.IP.FLG)                                   (* mjs "25-Sep-85 15:47")
    (PROG (CHAPS.TO.PROCESS)
          [SETQ CHAPS.TO.PROCESS (if (NULL CHAPNAMES)
                                     then
```

(* if chapter names are not given, check which chapters have been updated since they were last processed)

```
                                         (COLLECT.MODIFIED.IM.CHAPTERS)
                                   elseif (EQ CHAPNAMES T)
                                     then (for X in IM.MANUAL.CHAPTERS collect (LIST (CAR X)
                                                                                     (CADR X)))
                                   else (for NAM in (U-CASE (MKLIST CHAPNAMES))
                                           join (for X in IM.MANUAL.CHAPTERS
                                                   when (EQ (U-CASE (CADR X))
                                                            NAM)
                                                   collect (LIST (CAR X)
                                                                 (CADR X]
          (printout T "will re-process chapters: " CHAPS.TO.PROCESS T)
          (for X in CHAPS.TO.PROCESS
             do (SETQ GLOBAL.CHAPTER.NUMBER (CAR X))
                (IM.TEDIT (PACKFILENAME 'BODY (CADR X)
                                'BODY IM.MANUAL.DIRECTORY)
                        (if NULL.IP.FLG
                            then '{NULL}FOO.IP
                          else (CADR X)))
                (for EXT in (if NULL.IP.FLG
                                then '(IMPTR IMERR)
                              else '(IP IMPTR IMERR))
                   bind (NAM _ (FILENAMEFIELD (CADR X)
                                              'NAME))
                        FROMFILE TOFILE
                   do (SETQ FROMFILE (PACKFILENAME 'NAME NAM 'EXTENSION EXT 'BODY '{DSK}FOO.IP))
                      (SETQ TOFILE (PACKFILENAME 'NAME NAM 'EXTENSION EXT 'BODY IM.MANUAL.DIRECTORY))
                      (if (INFILEP FROMFILE)
                          then (printout T "copying from " FROMFILE " to " TOFILE "...." T)
                               (RENAMEFILE FROMFILE TOFILE])
```

## (REF.TO.PAGE

```
  [LAMBDA (REF DEFAULT-CHAPTER)                                     ; Edited  8-Dec-91 15:08 by jds
    ;; Returns a list of the form (chap# page#)
    (LIST (CAR (LAST (OR (fetch (IM.INDEX.DATA SUBSEC) of REF)
                         DEFAULT-CHAPTER)))
          (fetch (IM.INDEX.DATA PAGE#) of REF])
```

## (REFS.TO.PAGES

```
  [LAMBDA (REFS)                                                    (* mjs " 6-Jun-85 14:43")
          (* * REFS is list of index refs. Returns a list of the form ((chap1 page1) |...|
          (chapN pageN)))
    (for X in REFS collect (REF.TO.PAGE X])
)

(DEFINEQ
```

## (MAKE.IM.INDEX

```
  [LAMBDA (OUTFILE.FLG VOLUME.INFO IMPTR.FILES IMPTR.TYPES)         ; Edited 30-Aug-91 10:38 by jds
```

```
    (SETQ IMPTR.FILES (OR IMPTR.FILES (DIRECTORY "*.IMPTR;")))
    (PROG* ((*READTABLE* *TEDIT-FILE-READTABLE*)
            [IMPTR.DATA (COND
                           (IMPTR.FILES (PROG ((IMPTR.HASH (HASHARRAY 500))
                                               (IMPTR.TOC.LIST NIL)
                                               (IMPTR.NAME.LIST NIL))
                                              (DECLARE (SPECVARS IMPTR.HASH IMPTR.TOC.LIST IMPTR.NAME.LIST))
                                              (for X in (MKLIST IMPTR.FILES)
                                                 do (printout T "Grabbing IMPTR file: " X T)
                                                    (GRAB.IMPTR X NIL IMPTR.TYPES))
                                              (RETURN (LIST IMPTR.HASH IMPTR.TOC.LIST IMPTR.NAME.LIST]
            (IMPTR.HASH (COND
                           (IMPTR.FILES (CAR IMPTR.DATA))
                           (T IMPTR.HASH)))
            (IMPTR.TOC.LIST (COND
                               (IMPTR.FILES (CADR IMPTR.DATA))
                               (T IMPTR.TOC.LIST)))
            (IMPTR.NAME.LIST (COND
                                (IMPTR.FILES (CADDR IMPTR.DATA))
                                (T IMPTR.NAME.LIST)))
            INDEX.DATA INDEX.DATA.BY.TYPE)
           (DECLARE (SPECVARS IMPTR.HASH IMPTR.TOC.LIST IMPTR.NAME.LIST))
```

;;; Use either global or local versions of IMPTR.HASH IMPTR.TOC.LIST IMPTR.NAME.LIST

```
           [SORT IMPTR.NAME.LIST (FUNCTION (LAMBDA (A B)
                                     (COND
                                        ((OR (NUMBERP A)
                                             (NUMBERP B))
                                         (ALPHORDER (MKSTRING A)
                                                    (MKSTRING B)))
                                        (T (ALPHORDER A B]
                                                    ; sort, then put all names before first A--- at end of list
           [PROG ((X IMPTR.NAME.LIST))
                 (until [OR (NULL X)
                            (EQ 'A (U-CASE (NTHCHAR (CADR X)
                                                    1]
                    do (SETQ X (CDR X)))
                 (COND
                    (X (RPLACD (LAST X)
                               IMPTR.NAME.LIST)
                       (SETQ IMPTR.NAME.LIST (CDR X))
                       (RPLACD X NIL]
           (SETQ SUBSEC.COUNT.LIST '(INDEX))
           (RETURN (MAKE.IM.DOCUMENT [LIST (FUNCTION (LAMBDA NIL
                                                     (DUMP.HEADERS.FOOTERS "INDEX" "INDEX")
                                                     (DUMPOUT FONT IM.CHAPTER.TITLE.FONT PARALOOKS
                                                            `(PARALEADING 0 LINELEADING 0 QUAD LEFT
                                                                    1STLEFTMARGIN 0 LEFTMARGIN 0 RIGHTMARGIN
                                                                    %, IM.TEXT.RIGHTMARGIN TABS %,
                                                                    IM.RIGHT.MARGIN.TABS TYPE PAGEHEADING
                                                                    SUBTYPE TITLEHEAD)
                                                           TAB DUMP.CHARS "INDEX" CR CR)
                                                     (DUMP.HRULE 6 NIL
                                                            `(PARALEADING 0 LINELEADING 0 QUAD LEFT
                                                                    1STLEFTMARGIN 0 LEFTMARGIN 0 RIGHTMARGIN
                                                                    %, IM.TEXT.RIGHTMARGIN TABS %,
                                                                    IM.RIGHT.MARGIN.TABS TYPE PAGEHEADING
                                                                    SUBTYPE TITLEHEADRULE))
                                                     (for INDEX.NAME in IMPTR.NAME.LIST
                                                        bind (LAST.CHAR _ NIL)
                                                             CURRENT.CHAR
                                                        do (SETQ CURRENT.CHAR (U-CASE (NTHCHAR INDEX.NAME 1)))
                                                           (COND
                                                              ((NEQ CURRENT.CHAR LAST.CHAR)
                                                               (DUMPOUT CR CR FONT BOLD PARALOOKS
                                                                      '(HEADINGKEEP ON PARALEADING 12)
                                                                       DUMP.CHARS CURRENT.CHAR CR CR)
                                                               (SETQ LAST.CHAR CURRENT.CHAR)))
                                                           (SETQ INDEX.DATA (GETHASH INDEX.NAME IMPTR.HASH))
                                                              ; INDEX.DATA is a lists of index refs
                                                           [SETQ INDEX.DATA.BY.TYPE
                                                            (PARTITION.LIST INDEX.DATA NIL
                                                                   [FUNCTION (LAMBDA (A)
                                                                               (U-CASE (COND
                                                                                          ((fetch (
                                                                                                  IM.INDEX.DATA
                                                                                                   TYPE)
                                                                                              of A))
                                                                                        (T 'TERM]
                                                                   (FUNCTION (LAMBDA (A B)
                                                                               (LIST.ORDER (U-CASE A)
                                                                                      (U-CASE B]
```

;; INDEX.DATA.BY.TYPE is a list of index refs partioned by type, case-independent, and sorted by
;; the ALPHORDER of the U-CASE of the types. Eventually, may want to make sorting more
;; dependent on the types <terms before fns before vars, etc>

```
                                                    (for X in INDEX.DATA.BY.TYPE
                                                       do (PRINT.INDEX.OBJECT INDEX.NAME X VOLUME.INFO)

                                                          ; print the info about a single object of a single type
                                                            ))
                                                 (COND
                                                    (IM.EVEN.FLG

                                    ;; this is a hack so that if you are going to print '[This page intentionally left blank]' on a blank page at
                                    ;; the end, skip to the right column or the next page

                                                                  (DUMPOUT CR CR START.PARA PARALOOKS
                                                                         '(NEWPAGEBEFORE T)
                                                                         DUMP.CHARS "    " CR CR]
                                                 OUTFILE.FLG
                                                 [TEDIT.COMPOUND.PAGEFORMAT [TEDIT.SINGLE.PAGEFORMAT NIL NIL NIL NIL NIL
                                                                                 IM.PAGE.LEFTMARGIN IM.PAGE.RIGHTMARGIN
                                                                                 IM.INDEX.PAGE.FIRST.TOPMARGIN IM.PAGE.BOTTOMMARGIN 2
                                                                                 NIL 18 `((RECTOFOOT %, IM.PAGE.LEFTMARGIN %,
                                                                                          IM.FOOTER.Y)
                                                                                    (RECTOFOOTRULE %, IM.PAGE.LEFTMARGIN %,
                                                                                          IM.FOOTER.RULE.Y)
                                                                                    (DRAFTMESSAGE %, IM.DRAFT.MESSAGE.X %,
                                                                                          IM.DRAFT.MESSAGE.BOTTOM.Y)
                                                                                    (TITLEHEAD %, IM.PAGE.LEFTMARGIN %,
                                                                                          IM.HEADER.Y)
                                                                                    (TITLEHEADRULE %, IM.PAGE.LEFTMARGIN %,
                                                                                          IM.HEADER.RULE.Y]
                                                       [TEDIT.SINGLE.PAGEFORMAT NIL NIL NIL NIL NIL IM.PAGE.LEFTMARGIN
                                                              IM.PAGE.RIGHTMARGIN IM.PAGE.TOPMARGIN IM.PAGE.BOTTOMMARGIN 2 NIL 18
                                                              `((DRAFTMESSAGE %, IM.DRAFT.MESSAGE.X %, IM.DRAFT.MESSAGE.TOP.Y)
                                                                (VERSOHEAD %, IM.PAGE.LEFTMARGIN %, IM.HEADER.Y)
                                                                (VERSOHEADRULE %, IM.PAGE.LEFTMARGIN %, IM.HEADER.RULE.Y)
                                                                (VERSOFOOT %, IM.PAGE.LEFTMARGIN %, IM.FOOTER.Y)
                                                                (VERSOFOOTRULE %, IM.PAGE.LEFTMARGIN %, IM.FOOTER.RULE.Y)
                                                                (DRAFTMESSAGE %, IM.DRAFT.MESSAGE.X %, IM.DRAFT.MESSAGE.BOTTOM.Y]
                                                       (TEDIT.SINGLE.PAGEFORMAT NIL NIL NIL NIL NIL IM.PAGE.LEFTMARGIN
                                                              IM.PAGE.RIGHTMARGIN IM.PAGE.TOPMARGIN IM.PAGE.BOTTOMMARGIN 2 NIL 18
                                                              `((DRAFTMESSAGE %, IM.DRAFT.MESSAGE.X %, IM.DRAFT.MESSAGE.TOP.Y)
                                                                (RECTOHEAD %, IM.PAGE.LEFTMARGIN %, IM.HEADER.Y)
                                                                (RECTOHEADRULE %, IM.PAGE.LEFTMARGIN %, IM.HEADER.RULE.Y)
                                                                (RECTOFOOT %, IM.PAGE.LEFTMARGIN %, IM.FOOTER.Y)
                                                                (RECTOFOOTRULE %, IM.PAGE.LEFTMARGIN %, IM.FOOTER.RULE.Y)
                                                                (DRAFTMESSAGE %, IM.DRAFT.MESSAGE.X %, IM.DRAFT.MESSAGE.BOTTOM.Y]
                                                 "Hardcopy of Index"
                                                 `(QUAD LEFT 1STLEFTMARGIN %, IM.INDEX.1STLEFTMARGIN LEFTMARGIN %, IM.INDEX.LEFTMARGIN
                                                        POSTPARALEADING 0 PARALEADING 0 LINELEADING 0])

)

(RPAQQ IM.MANUAL.CHAPTERS
       (((NIL iii iv v vi vii viii ix x xi xii xiii xiv xv)
         ChapAck)
        (1 ChapIntro)
        (2 ChapLitatoms)
        (3 ChapLists)
        (4 ChapStrings)
        (5 ChapArrays)
        (6 ChapHashArrays)
        (7 ChapNumbers)
        (8 ChapRecordPackage)
        (9 ChapControlFns Conditionals PROG IterativeStatements)
        (10 ChapFnDef FNTYPES FNDEF FNEVAL MACROS)
        (11 ChapStack STACKORG STACKFNS INTERPRETERBLIPS GENERATORSANDCOROUTINES)
        (12 ChapMisc GREET IdleMode VirtualMemory VersionInfo DATETIME DURATION RESOURCES PATTERNMATCH)
        (13 ChapExec PAINTRO PACOMS PAMISC PAGUTS PAFNS)
        (14 ChapErrors BREAKS ERRORFNS RESETVARS ERRORLIST)
        (15 ChapBreaking BREAKFNS ADVISING)
        (16 ChapEditor DEDIT EDITORATTNCOMS EDITORMODCOMS EDITORMISC EDITORFNS)
        (17 ChapFP FPINTRO FPFNS FPTYPES FPCOMS FPFILECOMS FPFORMAT)
        (18 ChapCompiler COMPILERINTRO COMPILERISSUES COMPILERFNS BLOCKCOMPILER COMPILERERRORS)
        (19 ChapMasterScope MSLANG MSORG)
        (20 ChapDWIM DWIMINTRO DWIMORG SPELLINGCORRECTION)
        (21 ChapCLISP CLISPINTRO CLISPCHARS CLISPDECLARATIONS CLISPORG CLISPGUTS)
        (22 ChapPerformance GarbageCollection VariableBindings MEASURING GAINSPACE)
        (23 ChapProcesses)
        (24 ChapFiles FileStreams FileNames FileOther Directories FileServers)
        (25 ChapIO INPUTFNS OUTPUTFNS RANDOMIO PRINTOUT READFILE READTABLES)
        (26 ChapUserIO Inspector PROMPTFORWORD ASKUSER TTYIN PRETTYPRINT)
        (27 ChapGraphics GraphicsPrimitives ImageStreams ImageStreamsImpl FONTS IMAGEOBJECTS)
        (28 ChapWindows WindowSystem Windows Menus AttachedWindows)
        (29 ChapHardcopy)
        (30 ChapTerminal InterruptChars TERMINALTABLES Dribble Mouse Keyboard Screen)
        (31 ChapEther ETHEROVERVIEW ETHERPUP ETHERNS ETHERINTROLEVELONE ETHERPUPLEVELONE ETHERNSLEVELONE
            ETHEROTHERLEVELONE)))

(RPAQQ IM.MANUAL.VOLUMES
       ((I 1 2 3 4 5 6 7 8 9 10 11 12)
```

```
        (II 13 14 15 16 17 18 19 20 21 22 23)
        (III 24 25 26 27 28 29 30 31)))

(RPAQQ IM.MANUAL.DIRECTORY "{PELE:}<DOC>IRM>")

(FILESLOAD IMTEDIT FREEMENU)

(PUTPROPS IMTOOLS COPYRIGHT ("Xerox Corporation" 1985 1986 1987 1988 1991))
```

## FUNCTION INDEX



## VARIABLE INDEX