(cmovfil)	<nine, message, 03093>	PROCEDURE	8C1L
(ghname)	<nine, message, 01583>	PROCEDURE	8C1I4R
(initstr)	<nine, message, 02760>	PROCEDURE	8C1I4A
(jform3)	<nine, message, 0406>	PROCEDURE	8C1D2
(message)	<nine, message, 03070>	EXT	8C1B1
(minmes)	<nine, message, 01853>	PROCEDURE	8C1G1
(mladdresses)	<nine, message, 01025>	PROCEDURE	8C1I4B
(mladrusr)	<nine, message, 01150>	PROCEDURE	8C1I4D
(mlalgiuh)	<nine, message, 01486>	PROCEDURE	8C1I4L
(mlaliuh)	<nine, message, 01507>	PROCEDURE	8C1I4M
(mlaluh)	<nine, message, 01522>	PROCEDURE	8C1I4N
(mlcisfusr)	<nine, message, 01115>	PROCEDURE	8C1I4C
(mldeferred)	<nine, message, 01414>	PROCEDURE	8C1I4T
(mldeliver)	<nine, message, 01304>	PROCEDURE	8C1I4G
(mlendgroup)	<nine, message, 01570>	PROCEDURE	8C1I4Q
(mlfileo)	<nine, message, 01178>	PROCEDURE	8C1I4E
(mlgarbagecollect)	<nine, message, 01550>	PROCEDURE	8C1I4P
(mlimmediate)	<nine, message, 01378>	PROCEDURE	8C1I4H
(mlitemlabel)	<nine, message, 01534>	PROCEDURE	8C1I4O
(mlsephost)	<nine, message, 01451>	PROCEDURE	8C1I4K
(mlsegg)	<nine, message, 01244>	PROCEDURE	8C1I4F
(sortmes)	<nine, message, 0885>	PROCEDURE	8C1B3
(xcheckdist)	<nine, message, 02678>	PROCEDURE	8C1I2
(xfilnam)	<nine, message, 02501>	PROCEDURE	8C1E1
(xmcopy)	<nine, message, 02651>	PROCEDURE	8C1F1
(xmterm)	<nine, message, 03003>	PROCEDURE	8C1J1
(xreform)	<nine, message, 01715>	PROCEDURE	8C1D1
(xsndmsg)	<nine, message, 02786>	PROCEDURE	8C1I3
(xsortmes)	<nine, message, 02862>	PROCEDURE	8C1H2

Message
  The Message subsystem enables you to handle Executive System SNDMSG communications through NLS. With this subsystem you can move your messages into NLS, sort messages, reformat Sendmail items to correspond with those sent through SNDMSG, and automatically send messages through SNDMSG via NLS. To load Message (unless Message has been made automatically available to you with the Useroptions Include command), use the Goto command, type the option character <CTRL-U>, and type "message". Once Message has been loaded during a work session, simply give the command Goto Message. The following commands are provided:
% Index in <nls,>
Copy Message (file) OK/CONTENT (to follow) DESTINATION LEVEL-ADJUST OK
  The Message command "Copy" copies your MESSAGE.TXT file (or another file you specify for CONTENT) to the DESTINATION you point to. It changes the format to make the messages easy to handle in NLS. 2
Move Message (file) OK/CONTENT (to follow) DESTINATION LEVEL-ADJUST OK
  The Message command "Move" copies your MESSAGE.TXT file (or another file you specify for CONTENT) to the DESTINATION you point to and then deletes the MESSAGE.TXT file. It changes the format to make the messages easy to handle in NLS. If you use this command, a TENEX work file should contain your messages in case something goes wrong. 3
Reformat (citations in) STRUCTURE (at) DESTINATION OK
  The Message command "Reformat" reformats your Sendmail citations to correspond with the way the Message subsystem command "Copy" and "Move" reformats SNDMSG citations, including date of receipt and the distribution list if possible. 4
Send Message OK
(To:) CONTENT
(Cc:) CONTENT
(Subject:) CONTENT
(Message:) MSGITEM
  ---Status of the distibution is diplayed here---
(Send the message now? (Type N to add to list)) ANSWER OK

MSGITEM = Message CONTENT or
          Statement/Branch/Group/Plex DESTINATION VIEWSPECS OK

  The Message command "Send" prompts you to provide information for a message, and then sends an Executive System SNDMSG to the people whose usernames or idents you specify. A network addres or IDENT may be specified for CONTENT following the prompts: (To:) and (Cc:). You can say "Message" for MSGITEM and follow it with CONTENT; OR you can specify a STRUCTURE command word and point to a STRUCTURE you have already created. Note that the command pays attention to viewspecs. 5
Sort Message (plex at) DESTINATION OK
  The Message command "Sort" orders your messages chronologically, with the most recent date first. 6
Universal commands (available in all subsystems)
##<nls, universal>## 7
%source-code% 8
  %Documentation%
    % Copied from [sri-kl]<programs>message.nls.53 on 22-Aug-77 %
    % inputs all of mail.txt file
      Copies/Moves sequential message file to plex at given level.
      Recognizes messages sent from NLS using SENDMAIL and attempts
      to rebuild structure of file. On all other messages, leaves

```
      in! CR-LF's and breaks to new statement on double CRs.  MOVE
      deletes sequential file when done.  Sort sorts by date on
      message, most recent first.
      Author: NDM %
   %sends messages from NLS
      requests distribution list, which may be any combination of:
         -local directory names
         -remote directory names, followed by "@hostname"
         -individual or group idents (when an ident is the same as a
         directory name, the directory will be assumed unless you
         preceed the ident by an asterisk *).
      Allows you to type in title of message, terminated by <CA>.
       Sends statement, branch, group, plex, or text (typein).
      The elements of the distributions can be separated by spaces
      and/or commas.  When terminated by a <CA>, the list will be
      checked and the program will tell you which names it didn't
      recognize.  It will then give you chance to go back and add to
      the list (if the problem was a typo, you can try again) or
      begin the distribution.
      The command Move Message will recognize messages sent by
      SENDMAIL and will attempt to rebuild the original NLS
      structure of the file.
      Authors: JEW NDM %
   % gets addresses given idents %
   % formats journal citations to match messages %
       Reformatted statements will look like the product of INMES,
      including date of receipt and distribution list if possible.
      Author:  NDM %
%grammar%                                                          8B
   FILE gmessage  % (arcsubsys, cgcml,) in NLS 9 (arcsubsys,
   cgcml9,)  (relnine,message.cml,) %                              8B1
      % COMPILE INSTRUCTIONS %                                     8B1A
        INCLUDE <nine, nls-grammar, flags !subsystems>            8B1A1
      % DECLARATIONS %                                            8B1B
        INCLUDE <nine, nls-grammar, declarations !universal>      8B1B1
        DECLARE FUNCTION                                          8B1B2
           xmterm, xsortmes, xfilnam, xmovfil, xmcopy, xsndmsg,
           xcheckdist, xreform;                                  8B1B2A
        DECLARE VARIABLE namfil, filtre, first, dlist;            8B1B3
        DECLARE COMMAND WORD
           "MESSAGEFILENAME" = 6 SELECTOR
              TYPEIN = adr1fo ADDRESS = adr1fo;                  8B1B4A1
      % COMMON RULES %                                            8B1C
        INCLUDE <nine, nls-grammar, rules !universal>
      % COMMANDS % SUBSYSTEM message KEYWORD "MESSAGE"             8B1D
        TERMINATION
           termrule = xmterm();  %cleanup FTP stuff%
        msort COMMAND =  "SORT" !L2! "MESSAGE" <"plex at">        8B1D2
           dest _ DSEL(#"STATEMENT")
           CONFIRM
           xsortmes(dest) ;
        mmove COMMAND =  "MOVE" "MESSAGE" <"file">                8B1D3
           ( CONFIRM
            / namfil _ LSEL( #"MESSAGEFILENAME"))
           xfilnam(namfil -> namfil)
           SHOW ( namfil)  %display file name%                    8B1D3D
```

```
      <"to follow">
      dest _ DSEL(#"STATEMENT")
      level _ levadj()
      CONFIRM
      xmcopy(dest, level, TRUE %delete message file%);
   mcopy COMMAND =  "COPY" "MESSAGE" <"file">                    8B1D4
      ( CONFIRM
       /namfil _ LSEL( #"MESSAGEFILENAME") )
      xfilnam(namfil -> namfil)
      SHOW ( namfil)  %display file name%                        8B1D4D
      <"to follow">
      dest _ DSEL(#"STATEMENT")
      level _ levadj()
      CONFIRM
      xmcopy( dest, level, FALSE %leave message file%);
   msend COMMAND = "SEND" "MESSAGE" <"structure"> CONFIRM 8B1D5
      CLEAR <"To (and cc):"> param _ LSEL (#"TEXT")
      CLEAR <"Subject:"> source _ LSEL (#"TEXT")
      CLEAR <"Message/STRUCTURE:">
         ( dent _ structure <"at"> dest _ DSEL(dent)
            vs _ viewspecs()
            % ps routine will accept strings as well as
            structure but that added capability was not
            requested and might confuse users %
         / "MESSAGE"
            CLEAR
            dent _ #"TEXT"
            dest _ LSEL(#"TEXT")
         % / sent _ "FILE" source _ NULL %
         % / sent _ "SEQUENTIAL" !L2! <"file"> source _ LSEL
         (#"FILE") %
         )
      CONFIRM
      first _ TRUE  %initialize ident string in BE%              8B1D5E
      CLEAR  <"verifying distribution list">                     8B1D5F
      xcheckdist(first, param -> dlist)
      CLEAR SHOWCONFIRM( dlist )                                 8B1D5H
      first _ FALSE  %ident string initialized in BE%            8B1D5I
      addlist                                                    8B1D5J
      CONFIRM
      CLEAR xsndmsg(source,dent,dest,vs);

      addlist =  CLEAR <"Send the message? (Type n to add to
      list)">                                                    8B1D5N
      param2 _ ANSWER                                            8B1D5O
      ( IF param2 / IF NOT param2 getdist addlist);

      getdist =                                                  8B1D5R
         CLEAR <"To (and cc):">
         param _ LSEL(#"TEXT")
         CLEAR  <"verifying distribution list">                  8B1D5R3
         xcheckdist(first, param -> dlist)
         CLEAR SHOWCONFIRM( dlist );                             8B1D5R5
   mrefor COMMAND = "REFORMAT" <"citations in">                    8B1D6
      filtre _ FALSE
      dent _ structure <"at">
```

```
        dest _ DSEL(dent)
        [ OPTION filtre _ TRUE <"filtered:">
           vs _ viewspecs()
           %needconfirm _ TRUE%]
        CONFIRM xreform(dent,dest,filtre,vs);
     INCLUDE <nine, nls-grammar, commands !universal>          8B1D7
     END.                                                      8B1D8
FINISH
```

```
                                                                    8B1E
%be%                                                                  8C
   FILE smessage % NLS 8.5 (arcsubsys,x110,) in NLS 9
   (arcsubsys,1109,) TO (relnine,message.subsys,)  %
      ALLOW!
      % declarations %
         (message) EXTERNAL _ (                                     8C1B1
            $"XMTERM", $xmterm,
            $"XSORTMES", $xsortmes,
            $"XFILNAM", $xfilnam,
            $"XMCOPY", $xmcopy,
            $"XSNDMSG", $xsndmsg,
            $"XCHECKDIST", $xcheckdist,
            $"XREFORM", $xreform,
            0,0);
      DECLARE STRING msgfilename[200], mstrng[2000], line[500],
      cc[1500] ;
      % reformat routine %
         (xreform)   % CL: ;  reformat journal references %
         PROCEDURE (ident REF, idest REF, filtre, vs REF );     8C1D1
            % Procedure description
               FUNCTION
                   reformat journal references in initial file
                     Buffer pages required:  3
                      Reformatted statements will look like the
                     product of INMES, including date of receipt and
                     distribution list if possible.
                     Author:  NDM
               ARGUMENTS
                  ident   structure
                  idest   text selection
                  filtre  filter flag
                  vs  viewspecs
               RESULTS
                  none
               NON-STANDARD CONTROL
                  none
               GLOBALS
                  none
               %
            % Declarations %
               LOCAL vspec, save1, save2, savca, savus, da REF,
               last, sw REF;
               LOCAL dest REF, dent;
            % Pick out relevant elts of param list %
               dent _ ELEM #ident#[cwtype];  %entity type%
               &dest _ ELEM #idest#[tppair];  %text ptr block
               address%
            % Reformat journal messages%
               &da _ lda();
               curmkr _ dest;  curmkr[1] _ 1;
               save1 _ da.davspec;
               save2 _ da.davspc2;
               savca _ da.dacacode;
               savus _ da.dausqcod;
               INVOKE (catxreform);
```

```
            IF filtre THEN
               BEGIN  %get new viewspecs%
               da.davspec _ vs.vs1;
               da.davspc2 _ vs.vs2;
               % Third and fourth word not set in argument
               conversion -- code commented out
               da.dacacode _ vs.vscacode;
               da.dausqcod _ vs.vsusqcod;
               %
               END
            ELSE
               BEGIN
               da.davspec _ 0;
               da.davspec.vslev _ da.davspec.vstrnc _
                  da.davspec.vsnamf _ da.davspec.vsdaft _
                     da.davspec.vsindf _ -1;
               da.dacacode _ da.dausqcod _ 0;
               END;
            CASE dent OF
               = cwstatement: %- statement -%
                  BEGIN
                  dpset(dsprfmt, dest, endfil, dest);
                  jform3 (dest) ;
                  END;
               = cwgroup %- group -%, = cwplex %- plex -%, =
               cwbranch %- branch -%:
                  BEGIN
                  dpset(dspallf, dest, endfil, endfil);
                  dest _ grptst(dest, [&dest+d2sel]: last);
                  vspec _ da.davspec;
                  vspec.vsbrof _ vspec.vsplxf _ FALSE;
                  %open sequence work area%
                     &sw _ openseq(dest, last, vspec, da.davspc2,
                     da.dausqcod, da.dacacode) ;
                  % work on all statements in seq work area%
                     WHILE seqgen(&sw) # endfil
                        DO jform3 (sw.swstid);
                  %close seq work area%
                     closeseq(&sw);
                  END;
               ENDCASE err(notyet);
            da.davspec _ save1;
            da.davspc2 _ save2;
            da.dacacode _ savca;
            da.dausqcod _ savus;
         % Return %
            RETURN;
         % Catchpharases %
            (catxreform) CATCHPHRASE();                    8C1D1F1
               BEGIN
               CASE SIGNALTYPE OF
                  = aborttype :
                     BEGIN
                     da.davspec _ save1;
                     da.davspc2 _ save2;
                     da.dacacode _ savca;
```

```
                da.dausqcod _ savus;
                END;
             ENDCASE;
          CONTINUE;
          END;
    END.

(jform3) PROCEDURE(stid);                                    8C1D2
    % declarations %
       LOCAL msgflag, newstid, i, j;
       LOCAL TEXT POINTER ide, df, de, tf, te, nf, ne,
       titlef, titlee, locf, loce;
       LOCAL STRING loc[100], rcvd[100], dist[500] ;
    IF NOT FIND SF(stid) > [EOL] $SP ^titlef < 1$NP ^ne >
    titlef THEN RETURN;
    %isolate location or message, or else return%
       IF FIND ["Location:"] $NP ^locf < $NP $PT EOL $NP
       ^titlee > locf [EOL/ENDCHR] < $NP ^loce >
          THEN  %not a journal message%
             BEGIN
             *loc* _ "Location:      ", locf loce;
             msgflag _ IF FIND locf > $PT SP "JRNL" THEN 1
             ELSE 0;
             END
          ELSE IF FIND ["Message:"] $NP ^locf < $NP $PT EOL
          $NP ^titlee >
             THEN  %an old format journal message%
                BEGIN
                FIND SE(locf) $NP ^loce "*****ypoC rohtuA
                :etoN*****" $NP ^loce;
                %copy message to a new sub-statement%
                   newstid _ newrng(locf.stfile);
                   insgrp(locf, levdown, newstid, newstid);
                   ST newstid _ locf loce;
                msgflag _ 2;
                END
             ELSE RETURN;  %not a journal reference%
    %isolate sender(s)%
       IF NOT FIND ne < 1$D ^nf $NP ^te [":] ^tf _tf ^de [
       "- 3L "- ] 2CH ^df $NP ^ide THEN RETURN;
    %if message, put in journal number %
       IF msgflag=2 THEN *loc* _ "Journal Message ", nf ne ;
       IF msgflag THEN *rcvd* _ "J, nf ne ; %to do name
       lookup for message for dist%
    %get distribution list%
       *dist* _ nf ne, ".NLS" ;
       nf _ orgstid ;
       INVOKE (catjform, nodist);
       IF (nf.stfile _ cloafil($dist) ) THEN
          BEGIN
          IF msgflag AND lookup ($nf, $rcvd, nametyp) =
          endfil THEN GOTO nodist;
          IF FIND SF(nf) ["Author(s):"] ["/] ^nf [";] ^ne
          _ne THEN
             *dist* _ nf ne
             ELSE dist.L _ 0;
```

```
            IF FIND SF(nf) ["Distribution:"] ["/] ^nf [";] ^ne
            _ne THEN
               *dist* _ *dist*, SP, nf ne ;
            close (nf.stfile) ;
            LOOP IF FIND SF(*dist*) ["(] ^nf _nf [")] ^ne THEN
            ST nf ne _ NULL ELSE EXIT LOOP ;
            IF dist.L THEN *dist* _ EOL, "   Distribution:  ",
            *dist* ;
            END
         ELSE dist.L _ 0;
      %get "Rcvd" date from signature; note if "edited"%
         (nodist): rcvd.L _ 0;                                  8C1D2G1
         fechsig (ide, $rcvd);
         IF FIND SF(*rcvd*) $NP "XXX" 1$SP ^nf SE(nf) $NP ^ne
            THEN *rcvd* _ EOL, "   Received at:   ", nf ne
            ELSE rcvd.L _ 0;
      %add sender, date, time, and number onto "new" %
         ST ide _ df de, tf te, "-P    ", SF(ide) ide, ":, SP,
         SP, titlef titlee, *dist*, *rcvd*, EOL, "   ", *loc*
         ;
      DROP (catjform);
      RETURN;
      (catjform) CATCHPHRASE();                                 8C1D2K
         BEGIN
         CASE SIGNALTYPE OF
            = aborttype :
               IF SIGNAL = ofilerr THEN
                  BEGIN
                  DISABLE(catjform);
                  dist.L _ 0;
                  TERMINATE;
                  END;
            ENDCASE;
         CONTINUE;
         END;
      END.
%Execute Move Command%
   (xfilnam)   % CL: ;  set up message file name %
   PROCEDURE ( inamfil REF, results REF);                       8C1E1
      % Procedure description
         FUNCTION
            Place the name of the message file in a global
            string.  If the User did not specify a file name,
            use the default message file name (message.txt or
            mail.txt).
         ARGUMENTS
            namfil:  file name specified or zero
            results:  result list passed backed to FE -- set
            with name of message file
         RESULTS
            none
         NON-STANDARD CONTROL
            none
         GLOBALS
            "msgfilename" set to name of message file
         %
```

```
        % Declarations %
          LOCAL namfil REF, fptrs[30] ;
          LOCAL TEXT POINTER fn1, fn2 ;
        % Get relevant elements from param lists %
          IF &inamfil THEN
             &namfil _ ELEM #inamfil#[tppair];  %txt ptr block%
        % Determine file name an place in string %
          msgfilename.L _ 0 ;
          IF NOT &inamfil THEN %default to login mail.txt or
          message.txt%
             BEGIN
             gdname (cuno, $msgfilename); %get login directory
             name%
             IF tops20flag THEN
                *msgfilename* _ "(, *msgfilename*,
                ",MAIL.TXT,)"
             ELSE
                *msgfilename* _ "(, *msgfilename*,
                ",MESSAGE.TXT,)" ;
             END
          ELSE
             BEGIN
             fn1_namfil; fn1[1]_namfil[1]; fn2_namfil[2];
             fn2[1]_namfil[3];
             *msgfilename* _ fn1 fn2 ;
             FIND SF(*msgfilename*) ^fn1 ;
             IF NOT FIND fn1 ["."] THEN
                BEGIN
                lnkprs ($fn1, $fptrs); %scans string, puts
                pointers in array; see (nls,const,)%
                fn1[1] _ fptrs[fe+1] ;
                ST fn1 fn1 _ ".TXT" ;
                END;
             END;
        % Set up results list %
          #results# _ *msgfilename*;
        % Return %
          RETURN;
        END.

    %Execute Copy Command%
      (xmcopy)   % CL: ;  execute copy/move command %
      PROCEDURE (idest REF, ilevel, dflag);                    8C1F1
        % Procedure description
          FUNCTION
             Execute copy or move message file command.  The
             difference between the Move and Copy command is
             that with the Move command, the message file is
             deleted while with the Copy command the message
             file remains unchanged.  This procedure is called
             after the message file name has been set up
          ARGUMENTS
             idest:   LIST --destination
             ilevel:  INTEGER --relative level adjust number
             dflag: BOOLEAN
                TRUE to delete message file (Move)
```

```
                FALSE to leave message file (Copy)
            RESULTS
               none
            NON-STANDARD CONTROL
               none
            GLOBALS
               msgfilename
            %
         % Declarations %
            (dest) LOCAL REF;                                    8C1F1B1
         % Get relevant elements from param lists %
            &dest _ ELEM #idest#[tppair];  %txt ptrs to
            destination%
         % Insert moved file into NLS file %
            minmes($msgfilename, &dest, ilevel, dflag);
         % Return %
            RETURN;
         END.

   %Utility routines for Move and Copy x-routines%
      (minmes)   % CL: ;  insert message file %
      PROCEDURE (namfil REF, dest REF, rlevel, dflag );          8C1G1
         % Procedure description
            FUNCTION
               insert message file into NLS file
            ARGUMENTS
               namfil:  addr message file string
               dest:   addr of text ptrs to NLS destination
               rlevel:    relative level adjust number
               dlag:  TRUE -- delete message file; FALSE -- leave
               message file
            RESULTS
               none
            NON-STANDARD CONTROL
               ABORT if cannot get jfn or open message file
            GLOBALS
               none
            %
         % Declarations %
            LOCAL stid, next, jfn, count, nlsorigin, level,
            lastlv, stindent, indent, spcnt ;
            (ptrs) LOCAL [4];                                    8C1G1B2
            LOCAL TEXT POINTER headstid, pt1, pt2, pt3, pt4, rc1,
            rc2;
            LOCAL STRING dir[50], rcd[50], date[50], title[1500],
            author[300] ;
         %Initialize local variables%
            dir.L _ mstrng.L _ date.L _ stid _ jfn _ 0;
            headstid _ dest;  %get text ptr%
         % recreate display %
            dpset (dspstrc, headstid, endfil, dpstp(headstid)) ;
         % Set up text pointers %
            FIND SF(*namfil*) ^pt1 SE(*namfil*) ^pt2 > ;
            ptrs_pt1;
            ptrs[1]_pt1[1];
            ptrs[2]_pt2;
```

```
        ptrs[3]_pt2[1];
      % open input file %
        lnbfls($ptrs,0, $line) ;
        IF NOT jfn _ lgetjfn (0, $line, $txtext, gtjoisf,
        $lit)
           THEN ABORT (ofilerr, $lit) ;
        IF NOT sysopen (jfn, read, chrtyp, $lit) THEN
           BEGIN
           reljfn (jfn) ;
           ABORT (ofilerr, $lit) ;
           END;
        jfntostr (jfn, $dir, 1B10) ;
        *dir* _ *dir*, ", " ; %include recipient in dist
        list%
      % for each message%
        LOOP
           BEGIN
              date.L _ title.L _ author.L _ cc.L _ next _
              nlsorigin _ level _ lastlv _ 0 ;
           %read header and get count else exit loop%
              IF NOT isread (jfn, $line, LF) THEN EXIT LOOP ;
              %header%
              IF NOT (FIND SE(*line*) $NP ((1$(D) ";)/TRUE)
              ^pt2 1$(D) ^pt1 ", ^rc2 SF(*line*) $NP ^rc1)
              THEN
                 BEGIN
                 line.L _ MAX (0, line.L-2) ;
                 IF line.L THEN stid _ headstid _ cisstr
                 (headstid, $line, sucdir) ;
                 REPEAT LOOP;
                 END;
              *date* _ pt1 pt2 ; %number of characters in
              message%
              count _ VALUE ($date) ;
              *rcd* _ rc1 rc2; %date and time received%
           %get title line of message%
              LOOP
                 BEGIN
                 IF count <= 0 THEN EXIT LOOP ;
                 isread (jfn, $line, LF); %get next line%
                 count _ count - line.L;
                 CCPOS SF(*line*) ;
                 CASE READC OF
                    ="D: %date, distribution%
                       CASE READC OF
                          ="a: %date and time%
                             BEGIN
                             IF NOT FIND "te: " ^pt1 SE(pt1)
                             $NP > ^pt2 THEN REPEAT CASE 2
                             (next) ;
                             *date* _ pt1 pt2 ;
                             next _ 0;
                             END;
                          ="i: %sendmail distribution list%
                             BEGIN
                             IF NOT FIND "stribution: " ^pt1
```

```
            SE(pt1) $NP > ^pt2 THEN REPEAT
            CASE 2 (next) ;
            *line* _ pt1 pt2 ;
            FIND SE(*line*) ^pt1 ;
            LOOP
               IF FIND pt1 [SP] ^pt1 THEN
               ST pt1 pt1 _ ", ELSE EXIT
               LOOP;
            FIND SF(*line*) ^pt1 ;
            LOOP
               IF FIND pt1 ["@] ^pt1 _pt1
               ^pt2
                  THEN
                  CASE READC OF
                  = ",, ="): ST pt1 pt2 _
                  NULL;
                  ENDCASE ST pt1 pt2 _ " at
                  "
                  ELSE EXIT LOOP;
            *cc* _ *line*, ", " ;
            next _ 0 ;
            END;
         ENDCASE REPEAT CASE 2 (next) ;
="F: %author%
   BEGIN
   IF NOT FIND "rom: " ^pt1 SE(pt1) $NP >
   ^pt2 THEN REPEAT CASE (next) ;
   *author* _  pt1 pt2 ;
   next _ 0;
   IF FIND SF(*author*) $NP $(LD/"-) "(
   $PT < ") > THEN nlsorigin _ TRUE ;
   END;
="R: %title line%
   BEGIN
   IF NOT (FIND "e:" 1$SP ^pt1 SE(*line*)
   $NP ^pt2 >) THEN REPEAT CASE (next) ;
   *title* _ pt1 pt2 ;
   next _ 2 ;
   END;
="S: %title, sender%
   CASE READC OF
      ="u: %subject%
         BEGIN
         IF NOT (FIND "bject:" 1$SP ^pt1
         SE(*line*) $NP ^pt2 >) THEN
         REPEAT CASE (next) ;
         *title* _  pt1 pt2 ;
         next _ 2 ;
         END;
      ="e: %sender%
         BEGIN
         IF NOT FIND "nder: " ^pt1
         SE(pt1) $NP > ^pt2 THEN REPEAT
         CASE (next) ;
         *author* _  pt1 pt2 ;
         next _ 0;
```

```
            END;
          ENDCASE REPEAT CASE 2 (next) ;
=2:  %additional line of title%
   BEGIN
   line.L _ line.L - 2 ; %remove CR LF%
   *title* _ *title*, EOL, *line* ;
   END;
='T: %To:%
   BEGIN
   IF NOT FIND "o: " $NP ^pt1 SE(pt1) $NP
   > ^pt2 THEN REPEAT CASE (next) ;
   *cc* _ *cc*, + pt1 pt2, ", " ;
   next _ 0 ;
   END;
='c, ='C: %cc:%
   BEGIN
   IF NOT FIND "c: " $NP ^pt1 SE(pt1) $NP
   > ^pt2 THEN REPEAT CASE (next) ;
   IF NOT cc.L THEN *cc* _ *dir* ;
   *cc* _ *cc*, - pt1 pt2, ", " ;
   next _ 3 ;
   END;
=3:  %additional cc: lines%
   BEGIN
   line.L _ line.L - 2 ;
   *cc* _ *cc*, *line*, ", " ;
   END;
='M: %mail from site%
   BEGIN
   IF NOT (FIND "ail from " 1$PT ^pt2
   SF(pt2) ^pt1) THEN REPEAT CASE (next)
   ;
   *title* _ pt1 pt2 ;
   next _ 0;
   END;
='N: %Net mail from site%
   BEGIN
   IF NOT (FIND "et mail from site " 1$PT
   ^pt2 SF(pt2) ^pt1) THEN REPEAT CASE
   (next) ;
   *title* _ pt1 pt2 ;
   next _ 0;
   END;
='-: %line of dashes?%
   BEGIN
   IF FIND (" - - -" / "------") CR LF
   THEN
      BEGIN
      next _ 0;
      REPEAT LOOP;
      END;
   REPEAT CASE (next) ;
   END;
=SP: %null line%
   BEGIN
   IF FIND $SP CR LF THEN
```

```
                        BEGIN
                        next _ 0;
                        REPEAT LOOP;
                        END;
                     REPEAT CASE (next) ;
                     END;
                  =CR: %null line%
                     BEGIN
                     IF READC = LF THEN
                        BEGIN
                        next _ 0;
                        REPEAT LOOP;
                        END;
                     REPEAT CASE (next) ;
                     END;
                  =LF: %null line%
                     BEGIN
                     next _ 0;
                     REPEAT LOOP;
                     END;
                  ="<^A>: %remove control-a, remnant of old
                  nlsorigin recognition scheme%
                     BEGIN
                     IF FIND SF(*line*) ^pt1 "<^A> ^pt2
                     THEN
                        ST pt1 pt2 _ NULL ;
                     REPEAT CASE (0) ;
                     END;
                  =0:   %exit header loop%
                     BEGIN
                     line.L _ line.L - 2 ;
                     IF nlsorigin THEN
                        BEGIN
                        FIND SF(*line*) ^pt1 $SP ^pt2 ;
                        stindent _ pt2[1] - pt1[1] ;
                        ST pt1 pt2 _ NULL ;
                        END
                     ELSE *line* _ *line*, EOL ;
                     EXIT LOOP;
                     END;
                  ENDCASE REPEAT CASE (next) ;
               END;
            IF FIND SF(*date*) ^pt1 $1SP 1$2D ^pt2 SP ^pt3
            3UL ^pt4 " 19" ^rc1 SE(pt1) $NP > ^rc2 THEN
            %new sndmsg format%
               *date* _ pt1 pt2, "-, pt3 pt4, "-, rc1 rc2 ;
            %put together string%
               IF cc.L <= 2 THEN *cc* _ *dir* ;
               cc.L _ MAX (0,cc.L-2) ; %extra ", "%
               *mstrng* _ *date*, SP, SP, *author*, ":, SP,
               SP, *title*, EOL, "   Distribution:  ", *cc*
               ;
               IF date.L
                  THEN *mstrng* _ *mstrng*, EOL, "
                  Received at:   ", *rcd*
                  ELSE *mstrng* _ *rcd*, *mstrng* ;
```

```
                    (test):                                    8C1G1G1C4
                    stid _ headstid _ cisstr (headstid, $mstrng,
                    rlevel) ;
                    rlevel _ levdown ;
                    *mstrng* _ *line* ;
                    lastlv _ 0;
                 %get body of message%
                    UNTIL count <= 0 DO
                       BEGIN
                       IF NOT isread (jfn, $line, LF) THEN EXIT
                       LOOP ;
                       count _ count - line.L ;
                       IF FIND SE(*line*) LF CR > THEN
                          BEGIN
                          line.L _ MAX (0, line.L - 2) ; %pull off
                          CR LF%
                          IF NOT nlsorigin THEN *line* _ *line*,
                          EOL ;
                          END;
                       IF nlsorigin
                          THEN
                             BEGIN
                             FIND SF(*line*) ^pt1 $SP ^pt2 ;
                             IF NOT mstrng.L THEN
                                BEGIN
                                IF stid=headstid
                                   THEN %first line held only
                                   <control-a>%
                                      BEGIN
                                      stindent _ pt2[1] - pt1[1];
                                      END
                                   ELSE %not first mstrng in
                                   messsage%
                                      BEGIN
                                       level _ MIN(((indent _
                                      pt2[1] - pt1[1]) -stindent) /
                                      3, lastlv+1);
                                      IF indent < stindent THEN
                                         BEGIN
                                         stindent _ indent;
                                         level _ 0;
                                         END;
                                      spcnt _ indent - stindent -
                                      level*3;
                                      pt1[1] _ spcnt+1; %leave in
                                      leading spaces not acounted
                                      for by statement level%
                                      CASE level OF
                                         <lastlv:
                                         BEGIN
                                         rlevel _ 0;
                                         FOR next _ level UP UNTIL
                                         = lastlv DO BUMP rlevel ;
                                         END;
                                         >lastlv: rlevel _ levdown
                                         ;
```

```
                                    ENDCASE rlevel _ 0 ;
                                 lastlv _ level ;
                                 END;
                              ST pt1 pt2 _ NULL ;
                              END
                           ELSE IF (NOT FIND SE(*mstrng*) 1NP)
                           THEN ST pt1 pt2 _ SP ELSE ST pt1 pt2 _
                           NULL ;
                           END;
                     IF (mstrng.L + line.L) > 1900 THEN %start a
                     new statement%
                        BEGIN
                        IF FIND SF(*mstrng*) ^pt1 SE(pt1) 1$NP
                        ("-------" $NP / TRUE) ^pt2 > THEN
                        *mstrng* _ pt1 pt2 ;
                        stid _ cisstr (stid, $mstrng, rlevel) ;
                        rlevel _ 0;
                        *mstrng* _ *line* ;
                        REPEAT LOOP;
                        END;
                     IF FIND SF(*line*) (EOL/SP) ENDCHR THEN
                     %start a new statement%
                        BEGIN
                        IF FIND SF(*mstrng*) ^pt1 SE(pt1) 1$NP
                        ("-------" $NP / TRUE) ^pt2 > THEN
                        *mstrng* _ pt1 pt2 ;
                        IF mstrng.L THEN
                           BEGIN
                           stid _ cisstr (stid, $mstrng, rlevel)
                           ;
                           mstrng.L _ rlevel _ 0 ;
                           END;
                        REPEAT LOOP;
                        END;
                     *mstrng* _ *mstrng*, *line* ;
                     END;
                  IF FIND SF(*mstrng*) ^pt1 SE(pt1) 1$NP
                  ("-------" $NP / TRUE) ^pt2 > THEN *mstrng* _
                  pt1 pt2 ;
                  IF mstrng.L THEN stid _ cisstr(stid, $mstrng,
                  rlevel) ;
                  rlevel _ 0;
               END;
            % Delete message file if delete flag set.  Must do this
            before closing file so that incoming messages won't be
            lost.  Therefore must do the delete by changing the fdb
            -- the delete file JSYS won't work if the file is
            opened. %
               IF dflag THEN
                  BEGIN
                  chnfdb(jfn, $fdbctl, 04B10, 04B10);  %delete bit%
                  chnfdb(jfn, $fdbsiz, -1, 0);  %size to 0 bytes%
                  END;
            % close the input file %
               IF NOT sysclose (jfn, $lit) THEN
                  BEGIN
```

```
            reljfn (jfn) ;
            dismes (2, $lit) ;
            %GOTO STATE;%
            END;
         % Return %
            RETURN;
         END.

      % sort command %
         DECLARE FIELD chif=[0,7:35] ;
         (xsortmes)   % LB: ;  sort x-routine %
         PROCEDURE (isource REF);                                    8C1H2
            % Procedure description
               FUNCTION
                  sort x-routine
               ARGUMENTS
                  isource:  source
               RESULTS
                  none
               NON-STANDARD CONTROL
                  none
               GLOBALS
                  none
               %
            % Declarations %
               LOCAL plexptr REF, stid1, stid2, stidx, rlevcnt;
            % Get relevant element from param list %
               &plexptr _ ELEM #isource#[tppair];  %addr. txt ptr.
               block%
            % Sort the plex %
               stid1 _ plxset (plexptr : stid2 ) ;
               stidx _ colgdest(stid1 : rlevcnt) ;
               sort (stidx, stid1, stid2, rlevcnt, $sortmes, FALSE)
               ;
               IF nlmode=fulldisplay THEN alldsp() ;
            % Return %
               RETURN;
            END.

         (sortmes)PROCEDURE(stid,outb,num);                          8C1H3
            LOCAL TEXT POINTER pt1, pt2, pt3;
            LOCAL STRING date[70];
            REF outb;
            %find and convert date to internal format%
               LOOP
                  BEGIN
                  IF NOT (FIND SF(stid) (D / ENP D]) ^pt1 _pt1 ^pt2
                  SE(pt1) ^pt3) THEN EXIT LOOP;
                  pt2[1] _ pt2[1] + 60;
                  pt2[1] _ MIN(pt3[1], pt2[1]);
                  *date* _ pt1 pt2;
                  IF (FIND SF(*date*) [" at"] ^pt2 < 3CH ^pt1) THEN
                  ST pt1 pt2 _ NULL;
                  IF NOT SKIP !idtim($date+chbmty, 0) AND NOT SKIP
                  !idtim($date+chbmty, 2B6) THEN EXIT LOOP;
                  outb _ 4B11-R2;
```

```
            RETURN (TRUE, 1);  %assign "backward" sortkey
            values to dates%
            END;
         outb _ 0;  %put "bad" ones on top%
         RETURN (TRUE, 1);
         END.
%message sending routines%
   %declarations%
      DECLARE mlent, mlgo, mlstop, mlvsf, mlusqc, mlhdr, mlsw;
      SET uclocal = 1, ucremote = 2, ucgrpid = 3, ucidvid = 4;
      SET alsepkeep = 1, alsepdiscard = 2;
      SET timeimmediate = 1, timedeferred = 2;
      SET fdbkterse = 1, fdbkverbose = 2;
      REF ftperm, ftpoh;
   (xcheckdist)   % CL: ;  adds idents to global dist. list %
   PROCEDURE (first, param REF, results REF);                    8C1I2
      % Procedure description
         FUNCTION
            adds distribution it to global string dlist and
            returns a string with the distribution list and
            the discarded idents
         ARGUMENTS
            first:  TRUE if first call this command, FALSE if
            not first call
            param:  idents to be added
            results:  results list to be returned to FE -- set
            to string to be displayed
         RESULTS
            results value set in 'results' parameter
         NON-STANDARD CONTROL
            none
         GLOBALS
            mstring --  intialized by 'initstr', set by
            'mladdresses'
         %
      % Declarations %
         (addr) LOCAL REF;                                      8C1I2B1
         LOCAL TEXT POINTER ptr1, ptr2;
         LOCAL STRING workarea[2000], dlist[500], dluk[500],
         dlna[500], dlbh[500] ;
      % Get relevant element from param lists %
         &addr _ ELEM #param#[tppair];  %addr. of text ptr.
         block%
      % initialize string if first time through%
         IF first THEN initstr();
      % initializations %
         dluk.L _ dlna.L _ dlbh.L _ workarea.L _ 0;
      % get distribution string %
         ptr1 _ addr; ptr1[1] _ addr[1];
         ptr2 _ addr[d2sel]; ptr2[1] _ addr[d2sel+1] ;
         *dlist* _ ptr1 ptr2;
      % check list %
         mladdresses ($dlist, $dluk, $dlna, $dlbh, $workarea)
         ;
         *mstrng* _ *mstrng*, " ", *workarea* ;
      %build message%
```

```
        workarea.L _ 0;
        IF dluk.L THEN *workarea* _ *workarea*, "   Unknown:
                ", *dluk*, CR, LF ;
        IF dlna.L THEN *workarea* _ *workarea*, "   No
        network address:  ", *dlna*, CR, LF ;
        IF dlbh.L THEN *workarea* _ *workarea*, "   Bad host
        name:       ", *dlbh*, CR, LF ;
        IF workarea.L THEN *workarea* _ CR, LF, "The
        following were discarded:", CR, LF, *workarea*, CR,
        LF;
        *workarea* _ *workarea*, CR, LF, "Distribution list:
              ", *mstrng*;
      %place message to be displayed in results list%
        #results# _ *workarea*;
      % Return %
        RETURN;
      END.

    (xsndmsg)   % LB: ;  xroutine for sending a message %
    PROCEDURE (isource REF, ident REF, idest REF, vs REF); 8C1I3
      % Procedure description
        FUNCTION
          xroutine for sending a message
        ARGUMENTS
          isource:   pointer to title or subject string
          ident:    type of source for message - a STRUCTURE
          idest:     pointer to the message
          vs:     viewspecs
        RESULTS
          none
        NON-STANDARD CONTROL
          none
        GLOBALS
          none
        %
      % Declarations %
        LOCAL dirno, dent, dest REF, source REF;
        LOCAL TEXT POINTER first, last, sf, sl, hf, hl, mlst;
        LOCAL STRING title[200], dlist[500], dluk[500],
        dlna[500], dlbh[500], workarea[2000], user[35],
        host[35], datim[50], messid[100], waste1[35],
        waste2[35] ;
      % Get relevant elements from param lists %
        dent _ ELEM #ident#[cwtype];  %entity token value%
        &source _ ELEM #isource#[tppair];  %addr. txt ptr.
        block%
        &dest _ ELEM #idest#[tppair];  %addr. txt ptr. block%
      % initializations %
        dirno _ hf _ hl _ 0;
        % for subject or title %
          sf _ source; sf[1] _ source[1];
          sl _ source[d2sel]; sl[1] _ source[d2sel+1];
          *title* _ sf sl;
        % for text of message %
          first _ dest; first[1] _ dest[1];
          last _ dest[d2sel]; last[1] _ dest[d2sel+1];
```

```
%create sequential file%
   %get filename%
      *dlbh* _ "FTPMAILWORK.", *initsr*, fvrdchar, "1" ;
   %get carbon copies%
      FIND SF(*mstrng*) ^hf SE(*mstrng*) ^hl >;
      WHILE mlalgiuh($hf, $hl, $waste1, $waste2, $user,
      $host, $mlst) DO
         BEGIN
         IF workarea.L THEN *workarea* _ *workarea*, ",
         ";
         *workarea* _ *workarea*, *user*;
         IF host.L THEN *workarea* _ *workarea*, "@,
         *host*;
         END;
   %get author%
      ! JSYS gjinf ;
      dirno _ R1 ;
      gdname (dirno, $dluk) ;
   %get host%
      ghname(lhostn, $dlna);
   %make header%
      *workarea*  _ "From: ", *dluk*, " at ", *dlna*,
      EOL, "To: ", *workarea*, EOL;
      datfrmt(-1, 2B7, $datim);
      *messid* _ "<[", *dlna*, "], *datim*, ". , *dluk*,
      ">;
      *workarea* _ *workarea*, "Message-ID: ", *messid*,
      EOL;
      FIND SF(*workarea*) ^hf SE(*workarea*) ^hl;
   %set up pointers%
      FIND SF(*title*) ^sf SE(*title*) ^sl ;
   INVOKE(catsndmsg);
   mlfileo ($dlbh, $dluk, $dlna, $sf, $sl, $hf, $hl,
   dent, $first, $last, &vs) ;
%mail it%
   mldeliver ($mstrng, $dlbh, $fdbkverbose,
   $timeimmediate, TRUE);
   ftpelim ($dlbh) ;
% clean up FTP stuff %
   ftpend();
% Return %
   DROP(catsndmsg);
   RETURN;
(catsndmsg) CATCHPHRASE();                              8C1I3I
   BEGIN
   CASE SIGNALTYPE OF
      = aborttype :
         BEGIN
         DISABLE(catsndmsg);
         ftpelim ($dlbh) ;
         END;
      ENDCASE;
   CONTINUE;
   END;
END.
```

```
%mail support routines -- assume "$emrtnfalse"%
   (initstr)   % CL: ;  init. distribution list and FTP %
   PROCEDURE;                                                    8C1I4A
      % Procedure description
         FUNCTION
            initializes the  distribution list string to
            zero initstr initializes FTP stuff
         ARGUMENTS
            none
         RESULTS
            none
         NON-STANDARD CONTROL
            none
         GLOBALS
            none
         %
      % Declarations %
      %empty distribution list string%
         mstrng.L _ 0;
      %initialize FTP stuff if FTP fork does not yet exist%
         IF NOT ftphnd THEN ftpbgn () ;
         ftpsem ($emrtnfalse) ;
      % Return %
         RETURN;
      END.

   (mladdresses) PROCEDURE (dlst, ulst, nalst, bhlst,
   olst);                                                        8C1I4B
      %derive address list from distribution list%
      %----------%
      LOCAL naflg, bhflg, oflg, usrcls, idstid, list,
      listflg;
      LOCAL STRING ftpusr [35], ftphst [35], user [35],
      memusr [35], mlst [1600], elst [400], nelst [400],
      erlst [400], xlit[700];
      LOCAL TEXT POINTER tp1, tp2, tp3, tp4, tp5;
      REF dlst, ulst, nalst, bhlst, olst;
      %----------%
      %initialize%
         ulst.L _ nalst.L _ bhlst.L _ olst.L _ 0;
         idstid _ orgstid;
         idstid.stfile _ open (0, jfname ($"identfile"));
         INVOKE(catmladd);
      %loop through distribution list%
         FIND SP(*dlst*) ^tp2;
         LOOP
            BEGIN
            IF NOT FIND tp2 > $(SP/",) ^tp1
            1$(LD/"-/"@/"*/"//"./"") ^tp2
               THEN EXIT LOOP;
            *user* _ tp1 tp2;
            usrcls _ mlclsfusr ($user, idstid.stfile,
            $ftpusr, $ftphst, $xlit);
            CASE usrcls OF
               = FALSE:
                  *ulst* _ *ulst*, *user*, SP;
```

```
= $uclocal:
   *olst* _ *olst*, *ftpusr*, "@ ";
= $ucremote:
   IF ftpvhst ($ftphst)
      THEN *olst* _ *olst*, *ftpusr*, "@,
      *ftphst*, SP
   ELSE *bhlst* _ *bhlst*, *user*, SP;
= $ucidvid:
   IF ftpusr.L THEN
      IF ftphst.L THEN
         BEGIN
         list _ IF ftpvhst ($ftphst)
            THEN &olst
            ELSE &bhlst;
         *[list]* _ *[list]*, *user*, "(,
         *ftpusr*, "@, *ftphst*, ") ";
         END
      ELSE *olst* _ *olst*, *user*, "(,
      *ftpusr*, "@) "
   ELSE
      *nalst* _ *nalst*, *user*, SP;
= $ucgrpid:
   BEGIN
   naflg _ bhflg _ oflg _ FALSE;
   lmemlist ($xlit, $mlst, $elst, $nelst,
   $erlst, idstid.stfile);
   FIND SF(*mlst*) ^tp5;
   LOOP
      BEGIN
      IF NOT FIND tp5 > [PT] ^tp3 _tp3 ["(]
      ^tp4 _tp4 ["]] ^tp5
         THEN EXIT LOOP;
      *memusr* _ tp3 tp4;
      IF NOT ckident ($memusr, $xlit,
      idstid.stfile)
         THEN err ();
      mladrusr ($xlit, $ftpusr, $ftphst);
      IF ftpusr.L  THEN
         IF NOT ftphst.L OR ftpvhst
         ($ftphst) THEN
            BEGIN
            list _ &olst;
            listflg _ $oflg;
            END
         ELSE
            BEGIN
            list _ &bhlst;
            listflg _ $bhflg;
            END
      ELSE
         BEGIN
         list _ &nalst;
         listflg _ $naflg;
         END;
      IF NOT ([listflg] := [listflg]+1) THEN
         *[list]* _ *[list]*, *user*, SP,
```

```
                                "[Members ";
                             *[list]* _ *[list]*, *memusr*, "(,
                             *ftpusr*, "@, *ftphst*, ") ";
                             END;
                          mlendgroup (&nalst, naflg);
                          mlendgroup (&bhlst, bhflg);
                          mlendgroup (&olst, oflg);
                          END;
                       ENDCASE err ();
                    END;
                 %terminate%
                    IF ulst.L THEN BUMP DOWN ulst.L;
                    IF nalst.L THEN BUMP DOWN nalst.L;
                    IF bhlst.L THEN BUMP DOWN bhlst.L;
                    IF olst.L THEN BUMP DOWN olst.L;
                    close (idstid.stfile := 0);
                    DROP (catmladd);
                    RETURN;

                 (catmladd) CATCHPHRASE();                          8C1I4B11
                    BEGIN
                    CASE SIGNALTYPE OF
                       = aborttype :
                          BEGIN
                          DISABLE (catmladd);
                          IF idstid THEN close (idstid.stfile);
                          END;
                       ENDCASE;
                    CONTINUE;
                    END;
                 END.
              (mlclsfusr) PROCEDURE (user, idfileno, ftpusr, ftphst,
              entry);                                               8C1I4C
                 %identify string as local user, ident, or remote
                 user%
                 %----------%
                 LOCAL STRING temp [35];
                 LOCAL TEXT POINTER tp1, tp2, tp3, tp4;
                 REF user, ftpusr, ftphst, entry;
                 %----------%
                 %local directory?%
                    INVOKE(catmlc, notdir);
                    *temp* _ *user*;
                    transdir ($temp);
                    *ftpusr* _ + SF(*user*) SE(*user*);
                    ftphst.L _ 0;
                    DROP (catmlc);
                    RETURN ($uclocal);
                    (notdir):                                       8C1I4C7H
                       %sysg2 _ 0;%
                 %user @ host?%
                    IF FIND SF(*user*) > ^tp1 ["@] ^tp3 ^tp2 _tp2
                    SE(*user*) ^tp4 THEN
                       BEGIN
                       *ftpusr* _ tp1 tp2;
                       *ftphst* _ + tp3 tp4;
```

```
                DROP(catmlc);
                RETURN ($ucremote);
                END;
            %ident?%
               FIND SF(*user*) ("*/) ^tp1 SE(*user*) ^tp2;
               *user* _ + tp1 tp2;
               IF ckident (&user, &entry, idfileno) THEN
                  BEGIN
                  IF orgrptst (&entry, 0) THEN
                     BEGIN
                     DROP(catmlc);
                     RETURN ($ucgrpid);
                     END;
                  mladrusr (&entry, &ftpusr, &ftphst);
                  *ftphst* _ + tp3 tp4;
                  DROP(catmlc);
                  RETURN ($ucidvid);
                  END;
            %unidentifiable string%
               DROP(catmlc);
               RETURN (FALSE);

            (catmlc) CATCHPHRASE();                              8C1I4C11
               BEGIN
               CASE SIGNALTYPE OF
                  = aborttype :
                     BEGIN
                     DISABLE (catmlc);
                     TERMINATE;
                     END;
                  ENDCASE;
               CONTINUE;
               END;
            END.
         (mladrusr) PROCEDURE (entry, ftpusr, ftphst);           8C1I4D
            %fetch mailing address for individual ident%
            %----------%
            LOCAL delmode;
            LOCAL STRING user [35], localhostname [35];
            LOCAL TEXT POINTER tp1;
            REF entry, ftpusr, ftphst;
            %----------%
            %select delivery mode%
               getinma (&entry, &ftpusr, 0, 0);
               getihost (&entry, &ftphst, 0, 0);
               getiuser (&entry, $user, 0, 0);
               FIND SF(*entry*) ^tp1;
               delmode _ ldelivery (tp1);
               IF ((delmode.delol AND user.L AND NOT
               delmode.delnet) OR NOT ftpusr.L OR NOT ftphst.L)
               THEN
                  BEGIN
                  *ftpusr* _ + SF(*user*) SE(*user*);
                  ftphst.L _ 0;
                  END
               ELSE
```

```
            *ftphst* _ + SF(*ftphst*) SE(*ftphst*);
         %if local host name, substitute null%
            ghname(lhostn, $localhostname);
            IF *ftphst* = *localhostname* THEN ftphst.L _ 0;
         %return%
            RETURN;
            END.

      (mlfileo) PROCEDURE (filename REF, author REF,
      authorhost REF, titlefront REF, titleend REF, hdrfront
      REF, hdrend REF, sent, sourcefront REF, sourceend REF,
      vs REF);                                                  8C1I4E
         %create formatted mail file%
         %----------%
         LOCAL delim, da REF, indirect, unmatch, mlvs1, mlvs2;
         delim _ 'D-100B;
         indirect _ 'B - 100B;
         unmatch _ 'U - 100B;
         %----------%
         &da _ lda ();
         *lit* _ delim, *filename*, delim, EOL,
            delim, *author*, delim, EOL,
            delim, *authorhost*, delim, EOL,
            delim, titlefront titleend, delim, EOL,
            delim, hdrfront hdrend, delim, EOL, unmatch;
         CASE sent OF
            = cwstatement: %statement %
               BEGIN
               mlgo _ sourcefront;
               mlstop _ sourcefront;
               mlent _ sent;
               END;
            =cwbranch, =cwgroup, =cwplex:
               %branch, group, plex %
               BEGIN
               mlgo _ sourcefront;
               mlstop _ sourceend;
               mlent _ sent;
               END;
            =cwtext, =cwcharacter, =cwinvisible, =cwlink,
            =cwnumber, =cwvisible, =cwword:  %string%
               BEGIN
               *lit* _ *lit*, sourcefront sourceend ;
               mlent _ FALSE ;
               END;
            % = cwfile: file %
            % = cwsequential: sequential %
            ENDCASE err($"Bad entity type -- 'mlfileo'");
         mlhdr _ $lit;
         mlusqc _ da.dausqcod;
         mlvsf _ da.davspec.vsusqf;
         mlvs1 _ da.davspec;
         mlvs2 _ da.davspc2;
         IF vs THEN
            BEGIN
            cspvs _ vs.vs1;
```

```
            cspvs[1] _ vs.vs2;
            da.davspec _ vs.vs1;
            da.davspc2 _ vs.vs2;
            END;
         INVOKE(catmlf);
         da.dausqcod _ $mlseqg;
         da.davspec.vsusqf _ TRUE;
         opseqf (&filename, &da, FALSE, FALSE, TRUE);  %output
         sequential%
         ftpex ($"ARGF", 1, &filename);
         ftpex ($"FML", 0);
         da.dausqcod _ mlusqc;
         da.davspec _ mlvs1;
         da.davspc2 _ mlvs2;
         DROP(catmlf);
         RETURN;
         (catmlf) CATCHPHRASE();                              8C1I4E28
            BEGIN
            CASE SIGNALTYPE OF
               = aborttype :
                  BEGIN
                  DISABLE(catmlf);
                  da.dausqcod _ mlusqc;
                  da.davspec _ mlvs1;
                  da.davspc2 _ mlvs2;
                  ftpelim (&filename);
                  END;
               ENDCASE;
            CONTINUE;
            END;
         END.

      (mlseqg) PROCEDURE (sw, which);                         8C1I4F
         %sequence generator for creating formatted mail file%
         %----------%
         REF sw, mlsw;
         LOCAL mlopnusqc, delta, lines, stnumon, stnamon,
         indflg, rindflg;
         %----------%
         CASE which OF
            = sqopn:
               BEGIN
               IF mlent THEN
                  BEGIN
                  mlopnusqc _ IF mlvsf THEN mlusqc ELSE 0;
                  &mlsw _ openseq (mlgo, mlstop, sw.swvspec,
                  sw.swvsp2, mlopnusqc, sw.swcacode);
                  END;
               RETURN;
               END;
            = sqgnxt: NULL;
            = sqcls:
               BEGIN
               IF mlent THEN closeseq (&mlsw);
               RETURN;
               END;
```

```
      ENDCASE err ();
   %must ensure the formatter does not perform line
   truncation on header!%
      lines _ sw.swvspec.vstrnc;
      sw.swvspec.vstrnc _ -1;
   % statement numbers off for header%
      stnumon _ sw.swvspec.vsstnf;
      sw.swvspec.vsstnf _ FALSE;
   %statement names on for header%
      stnamon _ sw.swvspec.vsnamf;
      sw.swvspec.vsnamf _ TRUE;
   %Indenting (and relative indenting) off for header%
      indflg _ (sw.swvspec.vsindf := FALSE);
      rindflg _ (sw.swvspec.vsrind := FALSE);
   send (&sw, mlhdr);
   sw.swvspec.vstrnc _ lines;
   sw.swvspec.vsstnf _ stnumon;
   sw.swvspec.vsnamf _ stnamon;
   sw.swvspec.vsindf _ indflg;
   sw.swvspec.vsrind _ rindflg;
   IF mlent THEN
      BEGIN
      seqgen (&mlsw);
      cpysw (&mlsw, &sw);
      delta _ (sw.swclvl := 1) - 1;
      sport (&sw);
      IF mlent # cwstatement % statement % THEN
         LOOP
            BEGIN
            seqgen (&mlsw);
            cpysw (&mlsw, &sw);
            sw.swclvl _ sw.swclvl - delta;
            sport (&sw);
            END;
      END;
   sw.swstid _ endfil;
   sport (&sw);
   END.

(mldeliver) PROCEDURE (addrlist, filename, fdbkmode,
timemode, optimizeorder);                                  8C1I4G
   %deliver formatted mail to address list%
   %----------%
   LOCAL proc, mainlist, auxlist, outcome, openfail;
   LOCAL STRING msgstring [300], templist [1000], grp
   [35], individual [35], user [35], host [35],
   unihostlist [2000], label [200], dpystring[2000];
   LOCAL TEXT POINTER tp1, tp2, tpbrk;
   REF addrlist, filename;
   %----------%
   INVOKE (catmld);
   mainlist _ &addrlist; auxlist _ $templist;
   proc _ IF timemode = $timeimmediate
      THEN $mlimmediate
      ELSE $mldeferred;
   grp.L _ 0;
```

```
FIND SF(*[mainlist]*) ^tp1 SE(tp1) ^tp2;
LOOP
   BEGIN
   IF NOT mlalgiuh ($tp1, $tp2, $grp, $individual,
   $user, $host, $tpbrk)
      THEN EXIT LOOP;
   mlitemlabel ($grp, $individual, $user, $host,
   $label);
   IF fdbkmode = $fdbkverbose THEN
      *msgstring* _ *label*, SP  %display ident%
   ELSE msgstring.L _ 0;  %do not display ident%
   outcome _ [proc] (&filename, $user, $host,
   fdbkmode, $msgstring : openfail);
   ST tpbrk tp1 _ NULL;
   tp2[1] _ tp2[1] - (tp1[1] - tpbrk[1]);
   tp1[1] _ tpbrk[1];
   IF mlgarbagecollect (mainlist) THEN
      BEGIN
      grp.L _ 0;
      FIND SF(*[mainlist]*) ^tp1 SE(tp1) ^tp2;
      END;
   IF msgstring.L THEN
      BEGIN  %display delivered msg in cf window%
      IF fdbkmode = $fdbkterse THEN  %display ident%
         *msgstring* _ *label*, SP, *msgstring*;
      typseq(ccfbwindow, nocawait, clrwno, crno,
      belno, $msgstring);
         %CR and LF are in message%
      END;
   IF NOT optimizeorder OR timemode = $timedeferred
   OR NOT host.L THEN REPEAT LOOP;
   mlsephost ($alsepkeep, mainlist, $host,
   $unihostlist);
   IF NOT unihostlist.L THEN REPEAT LOOP;
   IF outcome OR NOT openfail
      THEN
         mldeliver ($unihostlist, &filename,
         fdbkmode, timemode, FALSE)
      ELSE
         BEGIN
         mldeliver ($unihostlist, &filename,
         $fdbkterse, $timedeferred, FALSE);
         %display message in cmd feedback wndw%
            *dpystring* _ "And therefore likewise for
            ", *unihostlist*, ".";
            typseq(ccfbwindow, cawait, clrwa, crno,
            belno, $dpystring);
         END;
   mlsephost ($alsepdiscard, mainlist, $host,
   auxlist);
   grp.L _ 0;
   mainlist _ (auxlist := mainlist);
   FIND SF(*[mainlist]*) ^tp1 SE(tp1) ^tp2;
   END;
IF timemode = $timeimmediate AND ftproh (0) THEN
ftpcls ();
```

```
        DROP(catmld);
        RETURN;
        (catmld) CATCHPHRASE();                                  8C1I4G17
           BEGIN
           CASE SIGNALTYPE OF
              = aborttype :
                 BEGIN
                 DISABLE(catmld);
                 IF timemode = $timeimmediate AND ftproh (0)
                 THEN ftpcls ();
                 END;
              ENDCASE;
           CONTINUE;
           END;
        END.

     (mlimmediate) PROCEDURE (filename, user, host, fdbkmode,
     msgstring);                                                 8C1I4H
        %deliver formatted mail immediately to specified
        address%
        %----------%
        LOCAL proc, openfail;
        LOCAL STRING openhost [35], localhostname [35];
        REF filename, user, host, msgstring;
        %----------%
        openfail _ FALSE;
        ftproh ($openhost);
        IF host.L AND *openhost* # *host* AND NOT (ftpcls ()
        AND ftpopn (&host)) THEN
           BEGIN
           openfail _ TRUE;
           *msgstring* _ *msgstring*,
              "Can't be delivered at this time;", CR, LF
                 %,"  ", *[sysg2]*, CR, LF%;
           mldeferred (&filename, &user, &host, $fdbkverbose,
           &msgstring);
           RETURN (FALSE, openfail);
           END;
        proc _ IF host.L THEN $ftpdml ELSE $ftplml;
        IF [proc] (&filename, &user) THEN
           IF fdbkmode = $fdbkverbose THEN
              *msgstring* _ *msgstring*, "Delivered.", CR, LF
           ELSE NULL
        ELSE
           BEGIN
           LOOP
              BEGIN
              IF host.L THEN EXIT;
              ghname(lhostn, $localhostname);
              IF mlimmediate(&filename, &user,
              $localhostname, fdbkmode, &msgstring :
              openfail) THEN RETURN (TRUE, openfail);
              RETURN (FALSE, openfail);
              END;
           *msgstring* _ *msgstring*,
              "Can't be delivered; "
```

```
            %,*[sysg2]*, CR, LF%;
         mldeferred (&filename, &user, &host, $fdbkverbose,
         &msgstring);
         END;
      RETURN (TRUE, openfail);
      END.

(mldeferred) PROCEDURE (filename, user, host, fdbkmode,
msgstring);                                              8C1I4I
      %deliver formatted mail deferred to specified
      address%
      %----------%
      LOCAL dirno ;
      LOCAL STRING directory[40] ;
      REF filename, user, host, msgstring;
      %----------%
      ! JSYS gjinf ;
      dirno _ R2 ;
      gdname (dirno, $directory) ;
      IF ftpgml (&filename, &user, &host, $directory) THEN
         IF fdbkmode = $fdbkverbose THEN
            *msgstring* _ *msgstring*, "Held for later
            delivery.", CR, LF
         ELSE NULL
      ELSE
         *msgstring* _ *msgstring*,
            "Can't be held for later delivery;", CR, LF
            %,"  ", *[sysg2]*, CR, LF%;
      RETURN (TRUE, FALSE%dummy%);
      END.


(mlsephost) PROCEDURE (action, list, host, newlist);
                                                         8C1I4K
      %create new address list with or w/o specified host%
      %----------%
      LOCAL STRING ngrp [35], nindividual [35], nhost [35],
      nuser [35];
      LOCAL TEXT POINTER tp1, tp2, tpbrk;
      REF list, host, newlist;
      %----------%
      *newlist* _ *list*;
      FIND SF(*newlist*) ^tp1;
      ngrp.L _ 0;
      LOOP
         BEGIN
         FIND SE(tp1) ^tp2;
         IF NOT mlalgiuh ($tp1, $tp2, $ngrp, $nindividual,
         $nuser, $nhost, $tpbrk) THEN EXIT LOOP;
         CASE action OF
            = $alsepkeep:
               IF *nhost* # *host* THEN
                  BEGIN
                  ST tpbrk tp1 _ NULL;
                  tp2[1] _ tp2[1] - (tp1[1] - tpbrk[1]) ;
                  tp1[1] _ tpbrk[1] ;
```

```
                        END;
                    = $alsepdiscard:
                        IF *nhost* = *host* THEN
                        BEGIN
                        ST tpbrk tp1 _ NULL;
                        tp2[1] _ tp2[1] - (tp1[1] - tpbrk[1]) ;
                        tp1[1] _ tpbrk[1] ;
                        END;
                    ENDCASE err ();
                END;
            IF list.L # newlist.L THEN mlgarbagecollect
            (&newlist);
            RETURN;
            END.

        (mlalgiuh) PROCEDURE (tps, tpe, grp, individual, user,
        host, tpbrk);                                          8C1I4L
            %FIND and skip past
              group[Members ... ^tpbrk individual(user@host)
            ...] or
              group[Members ... ^tpbrk user@host ...]%
            %----------%
            LOCAL TEXT POINTER tp1, tp2;
            REF tps, tpe, grp, individual, user, host, tpbrk;
            %----------%
            IF NOT grp.L AND
              FIND > BETWEEN tps tpe ($SP ^tp1 1$(LD/'-) ^tp2
              $SP "[Members" ^tps)
              THEN *grp* _ tp1 tp2;
            IF grp.L AND
              FIND > BETWEEN tps tpe ($SP '] ^tps)
              THEN grp.L _ 0;
            IF &tpbrk THEN FIND tps ^tpbrk;
            individual.L _ 0;
            IF mlaliuh (&tps, &tpe, &individual, &user, &host)
              OR mlaluh (&tps, &tpe, &user, &host)
              THEN RETURN (TRUE);
            IF FIND > BETWEEN tps tpe ([PT]) THEN err ();
            RETURN (FALSE);
            END.

        (mlaliuh) PROCEDURE (tps, tpe, individual, user, host);
                                                               8C1I4M
            %FIND and skip past individual(user@host)%
            %----------%
            LOCAL TEXT POINTER tp1, tp2, tp3;
            REF tps, tpe, individual, user, host;
            %----------%
            IF NOT FIND > BETWEEN tps tpe ($SP ^tp1 1$(LD/'-)
            ^tp2 '( ^tp3)
              THEN RETURN (FALSE);
            IF NOT (mlaluh ($tp3, &tpe, &user, &host)
              AND FIND > BETWEEN tp3 tpe ($SP ') ^tps))
              THEN err ();
            *individual* _ tp1 tp2;
            RETURN (TRUE);
```

```
    END.

(mlaluh) PROCEDURE (tps, tpe, user, host);          8C1I4N
    %FIND and skip past user@host%
    %----------%
    LOCAL TEXT POINTER tp1, tp2, tp3;
    REF tps, tpe, user, host;
    %----------%
    IF NOT FIND > BETWEEN tps tpe ($SP ^tp1
    1$(LD/"-/"//"./"") ^tp2 "@ ^tp3 $(LD/"-) ^tps) THEN
    RETURN (FALSE);
    *user* _ tp1 tp2;
    *host* _ tp3 tps;
    RETURN (TRUE);
    END.

(mlitemlabel) PROCEDURE (grp, individual, user, host,
label);                                             8C1I4O
    %construct feedback label for specified addressee%
    %----------%
    REF grp, individual, user, host, label;
    %----------%
    *label* _ *user*, "@;
    IF host.L THEN
       *label* _ *label*, *host*;
    IF individual.L THEN
       *label* _ *individual*, "(, *label*, ");
    IF grp.L THEN
       *label* _ *grp*, "[Member ", *label*, "];
    IF rlmode = typewriter THEN *label* _ CR, LF,
    *label*;
    RETURN;
    END.

(mlgarbagecollect) PROCEDURE (list);                8C1I4P
    %delete null group lists from address list%
    %----------%
    LOCAL outcome;
    LOCAL TEXT POINTER tp1, tp2;
    REF list;
    %----------%
    outcome _ FALSE;
    FIND SF(*list*);
    LOOP
       IF NOT FIND > ["[Members"] $SP "] $SP ^tp2 < ["[]
       $SP 1$(LD/"-) ^tp1
          THEN EXIT LOOP
       ELSE
          BEGIN
          ST tp1 tp2 _ NULL;
          outcome _ TRUE;
          END;
    RETURN (outcome);
    END.

(mlendgroup) PROCEDURE (list, listflg);             8C1I4Q
```

```
            %terminate group's portion of address list%
            %----------%
            REF list;
            %----------%
            IF listflg THEN
               BEGIN
               BUMP DOWN list.L;
               *list* _ *list*, "] ";
               END;
            RETURN;
            END.

         (ghname)   PROCEDURE % Convert host number and into a
         hostname string.  Calls err if error occurs. %     8C1I4R
            (hstno, % TENEX host number %
            hname   % Address of string into which host name is
            to
               be placed %
            );
            LOCAL length, bp;
            REF hname;
            R1 _ chbmty + &hname;
            bp _ R1;
            R2 _ hstno;
            IF NOT SKIP !cvhst() THEN err($"System error");
            length _ slngth(bp, R1);
            IF length > hname.M THEN err($"NLS System error");
            hname.L _ length;
            RETURN;
            END.

      % termination procedure %
         (xmterm)   % CL: ;  termination procedure %
         PROCEDURE;                                              8C1J1
            % Procedure description
               FUNCTION
                  If FTP fork still exists, call "ftpend"
               ARGUMENTS
                  none
               RESULTS
                  none
               NON-STANDARD CONTROL
                  none
               GLOBALS
                  ftphnd
               %
            % Declarations %
            % If FTP fork handle is non-zero,  kill fork %
               IF ftphnd THEN ftpend();
            % Return %
               RETURN;
            END.

      % procedures from (nine,identsupport,) %
         INCLUDE <nine, identsupport, sendmail>                  8C1K1
      (cmovfil)   % GB: core NLS Move File procedure %
```

```
PROCEDURE (rhost1, file1, rhost2, file2, astr % => no value
%);                                                          8C1L
   % Procedure description
      FUNCTION
         Moves the NLS file(s) specified by 'file1' on host
         'rhost1' to file(s) 'file2' on host 'rhost2'.  Any
         partial copies are also moved.  Asterisks (*) in the
         source file specification (file1) must be matched by
         asterisks in the destination file specification
         (file2).  Messages describing the results are put in
         'astr'.  Note: this moves the source file's origin
         statement along with its other statements.
      ARGUMENTS
         rhost1 - INTEGER -
            number of the host on which the files reside.
            Only the local host ('lhostn') is currently
            implemented.
         file1 - STRING -
            file group name string in TENEX format.  Connected
            directory is assumed if a directory is not
            specified.  Examples:
               "*.*"
               "<SMITH>*.PC"
               "<SMITH>FOO.NLS;3"
         rhost2 - INTEGER -
            number of the host to which the files are to be
            moved.  Only the local host ('lhostn') is
            currently implemented.
         file2 - STRING -
            file group name string in TENEX format.  Connected
            directory is assumed if a directory is not
            specified.  Examples:
               "*.*"
               "<SMITH>*.PC"
               "<SMITH>FOO.NLS;3"
            Note: the *'s here must match one-for-one with the
            *'s in file1.
         astr - STRING - string to receive the file names
      RESULTS
         none
      NON-STANDARD CONTROL
         Error if
            <CTRL-F> or <ALTMODE> is used in 'file2'
            *'s don't match
            input files don't exist
            output files are deleted but not expunged
            other file-moving problems, e.g. protection
            violations
      GLOBALS
         <lots>
      EXAMPLE
         cmovfil(lhostn, $"*.NLS;*", lhostn,
         $"<SMITH>*.NLS;*", $string)
      %
   % Declarations %
      LOCAL
```

```
            jfn1, % main jfn %
            jfn2, % jfn for destination file %
            jfnfl1,       % left half flags for a group jfn %
            jfnfl2,       % left half flags for destination group
            jfn %
            pcjfn1,       % jfn of partial copy %
            pcjfn2,       % jfn of partial copy %
            flag1,        % bits indicating * typed for file name
            fields %
            flag2;        % * typed for destination file name
            fields %
         LOCAL STRING
            uf1[200],     % complete name as entered by user %
            jfnnm1[200],% complete name from jfns %
            uf2[200],     % complete destination name as entered
            by user %
            udir2[40],    % user input directory destination name
            %
            ufil2[40],    % user input file destination name %
            uext2[40],    % user input extension destination name
            %
            uver2[40],    % user input destination version number
            %
            ulft2[40],    % user destination input beyond version
            number %
            jfnnm2[200],  % complete destination name from jfns %
            errstring[200],       % eror message string %
            tstring[200]; % temporary string %
         REF file1, file2, astr;
      % initial variables %
         jfn1 _ jfnfl1 _ pcjfn1 _ flag1 _ 0;
         jfn2 _ jfnfl2 _ pcjfn2 _ flag2 _ 0;
      % find out if remote or local (disk) file %
         CASE rhost1 OF
            = lhostn:  NULL;
            ENDCASE err( $"remote file manipulations not
            implemented yet" );
         CASE rhost2 OF
            = lhostn:  NULL;
            ENDCASE err( $"remote file manipulations not
            implemented yet" );
      % now parse user source input string %
         parseinput( &file1, $udir2, $ufil2, $uext2, $uver2,
         $ulft2, $uf1, $flag1);
      % parse user destination input string %
         parseinput( &file2, $udir2, $ufil2, $uext2, $uver2,
         $ulft2, $uf2, $flag2);
         % illegal to terminate destination fields with ^F or alt
         %
            CASE *udir2*[udir2.L] OF
               = "<^F>, = "<ESC>: err($"Illegal use of ^F or
               ALTMODE");
               ENDCASE;
            CASE *ufil2*[ufil2.L] OF
               = "<^F>, = "<ESC>: err($"Illegal use of ^F or
               ALTMODE");
```

```
            ENDCASE;
         CASE *uext2*[uext2.L] OF
            = "<^F>, = "<ESC>: err($"Illegal use of ^F or
            ALTMODE");
            ENDCASE;
         CASE *uver2*[uver2.L] OF
            = "<^F>, = "<ESC>: err($"Illegal use of ^F or
            ALTMODE");
            ENDCASE;
         CASE *ulft2*[ulft2.L] OF
            = "<^F>, = "<ESC>: err($"Illegal use of ^F or
            ALTMODE");
            ENDCASE;
      % check for matching astericks %
         IF flag1.dstar AND NOT flag2.dstar THEN err($"Illegal
         use of *");
         IF flag1.fstar AND NOT flag2.fstar THEN err($"Illegal
         use of *");
         IF flag1.estar AND NOT flag2.estar THEN err($"Illegal
         use of *");
         IF flag1.vstar AND NOT flag2.vstar THEN err($"Illegal
         use of *");
      % get main source jfn (old file only, group jfn) %
         IF NOT (jfn1 _ sgtjfn( gtjoif .V gtjstr + 0, $uf1,
         $errstring : jfnfl1 ) )THEN err( $errstring );
      % now find out if we support this type of file %
         IF NOT chkdev( jfn1 ) THEN err( $"only disk files
         supported" );
      % now loop to get all files in the group %
         LOOP
            BEGIN
            % check to make sure this not a PC with * for
            extension name %
               IF NOT chkpcs( jfn1, flag1) THEN GOTO gtmnxtjfn;
            % get actual file name into string for use later
            (maybe) %
               R3 _ 01111086      %  directory file ext ver %
                  + (IF jfnfl1.lhjb9  THEN 1B6 ELSE 0) %
                  ;Protection %
                  + (IF jfnfl1.lhjb10 THEN 1B5 ELSE 0) % ;Account
                  %
                  + (IF jfnfl1.lhjb11 THEN 4B4 ELSE 0) % ;T %
                  + 1;              % with proper file punctuation
                  %
               jfnflink( jfn1, $jfnnm1, R3);
            % get source PC jfn & find if this user can access
            this PC %
               IF NOT getpcjfn( jfn1, $pcjfn1, $errstring, FALSE)
               THEN
                  BEGIN   % this user can't access this file %
                  *tstring* _ *jfnnm1*, " not moved because [",
                  *errstring*, "];
                  IF flag1 THEN
                     BEGIN
                     *astr* _ *astr*, "   ***   ", *tstring*, CR,
                     LF;
```

```
                    GOTO gtmrxtjfn;
                    END
                ELSE
                    BEGIN
                    IF NOT SKIP !rljfn( jfn1) THEN NULL;
                    err( $tstring );
                    END;
                END;
            % get destination jfns %
              IF NOT jfn2 _
                gtdesjfn( jfn1, pcjfn1, $udir2, $ufil2, $uext2,
                $uver2, $ulft2, flag2, $errstring : pcjfn2)
                THEN
                    BEGIN        % can't move file (or partial
                    copy) %
                    *tstring* _ *jfnnm1*, " not moved because ",
                    *errstring*;
                    IF flag1 THEN
                      BEGIN
                      *astr* _ *astr*, "   ***   ", *tstring*,
                      CR, LF;
                      GOTO gtmnxtjfn;
                      END
                    ELSE
                      BEGIN
                      IF NOT SKIP !rljfn( jfn1) THEN NULL;
                      IF NOT SKIP !rljfn( pcjfn1) THEN NULL;
                      err( $tstring );
                      END;
                    END;
            % get destination file name into string for use later
            %
              R3 _ 011110B6      %  directory file ext ver %
                + (IF jfnfl1.lhjb9  THEN 1B6 ELSE 0) %
                ;Protection %
                + (IF jfnfl1.lhjb10 THEN 1B5 ELSE 0) % ;Account
                %
                + (IF jfnfl1.lhjb11 THEN 4B4 ELSE 0) % ;T %
                + 1;             % with proper file punctuation
                %
              jfnflink( jfn2, $jfnnm2, R3);
            % now move the file (and its partial copy) %
              IF NOT movflandpc(jfn1, pcjfn1, jfn2, pcjfn2,
              $errstring) THEN
                BEGIN   % can't move file (or partial copy) %
                *tstring* _ *jfnnm1*, " not moved because ",
                *errstring*;
                IF flag1 THEN
                    BEGIN
                    *astr* _ *astr*, "   ***   ", *tstring*, CR,
                    LF;
                    GOTO gtmnxtjfn;
                    END
                ELSE
                    BEGIN
                    IF NOT SKIP !rljfn( jfn1) THEN NULL;
```

```
                    IF NOT SKIP !rljfn( pcjfn1) THEN NULL;
                    err( $tstring );
                    END;
                END;
            % update the file return ring %
              uplsfnm( $jfnnm1, $jfnnm2);
            % now tell the user we have moved this file %
              *astr* _ *astr*, "   ", *jfnnm1*;
              IF pcjfn1 THEN *astr* _ *astr*, " [and PCJ";
              *astr* _ *astr*, " to ", *jfnnm2*;
              IF pcjfn2 THEN *astr* _ *astr*, " [and PCJ";
              *astr* _ *astr*, CR, LF;
            % now get rid of the jfn for the partial %
              IF NOT SKIP !rljfn( pcjfn1 ) THEN NULL;
            % now get the next file in the group %
              (gtmnxtjfn):                                    8C1L10A11A
              R1.LH _ jfnfl1;
              R1.RH _ jfn1;
              IF NOT SKIP !gnjfn( R1 ) THEN EXIT LOOP;
            END;
        % now get rid of any lingering jfns %
            IF NOT SKIP !rljfn( jfn1 ) THEN NULL;
            IF NOT SKIP !rljfn( pcjfn1 ) THEN NULL;
        % all done now, so return %
           RETURN;
        END.   %%
```

```
FINISH
   (mcnvname)   % LB: ;  convert to TENEX/TOPS20 name format %
   PROCEDURE (nstring REF, tstring REF);                    8C1M1
      % Procedure description
         FUNCTION
            Get the TENEX/TOPS20 file name format for an NLS
            link
         ARGUMENTS
            nstring:  addr of string with NLS link
            tstring:  addr of string with TENEX/TOPS20 format
         RESULTS
            hostn:  host number
         NON-STANDARD CONTROL
            none
         GLOBALS
            none
         %
      % Declarations %
         (hostn) LOCAL;                                     8C1M1B1
         (ptrs) LOCAL [4];                                  8C1M1B2
         LOCAL TEXT POINTER fn1, fn2;
      % Set up text pointers %
         FIND SF(*nstring*) ^fn1 SE(*nstring*) ^fn2 > ;
         ptrs_fn1;
         ptrs[1]_fn1[1];
         ptrs[2]_fn2;
         ptrs[3]_fn2[1];
      % convert to TENEX/TOPS20 format %
         hostn _ lnbfls($ptrs, 0, &tstring);
      % Return %
         RETURN(hostn);
      END.

   (xmovfil)   % LB: ;  interface grammar to move procedures %
   PROCEDURE (idest REF, level);                            8C1M2
      % Procedure description
         FUNCTION
            Interface to the various BE procedure that execute
            the move file command.  This procedure was written
            for NLS9 to make the call on the BE more
            efficient, i.e. the grammar calls this one BE
            procedure rather than making the 4 calls to the BE
            it had made in NLS 8.5 ("xworkfil", "xmovfil",
            "xinmes", "xdelfil").  When this procedure is
            caled, "msgfilename" has already been set to the
            name of the message file.
         ARGUMENTS
            idest:  destination
            level:  relative level adjust number
         RESULTS
            none
         NON-STANDARD CONTROL
            none
         GLOBALS
            msgfilename, tempfil
```

```
    %
% Declarations %
    (dest) LOCAL REF;                                          8C1M2B1
    (mhostn) LOCAL;                                            8C1M2B2
    (whostn) LOCAL;                                            8C1M2B3
    (mfstr) STRING[200];                                       8C1M2B4
       %msg file name in complete TENEX/TOPS20 format%
    (wfstr) STRING[200];                                       8C1M2B5
       %work file name in complete TENEX/TOPS20 format%
    (dummy) STRING[200];                                       8C1M2B6
    LOCAL TEXT POINTER  fn1, fn2;
% Get relevant elements from param lists %
    &dest _ ELEM #idest#[tppair];  %txt ptrs to
    destination%
% Get file names in operating system format %
    mhostn _ mcnvname($msgfilename, $mfstr);
    whostn _ mcnvname($tempfil, $wfstr);
% Move message file to work file %
    cmovfil(mhostn, $mfstr, whostn, $wfstr, $dummy);
% Insert moved file into NLS file %
    minmes($wfstr, &dest);
% Delete work file %
    cdelfil( whostn, $wfstr, $lit);
% Return %
    RETURN;
END.
```

(cpspst)	<nine, modify, 0344>	PROC	8C8B
(csubsp)	<nine, modify, 0375>	LOCAL	8C8E
(delname)	<nine, modify, 0208>	PROCEDURE	8C5B
(delsp)	<nine, modify, 0228>	PROCEDURE	8C5C
(dl)	<nine, modify, 096>	STRING	8C2D
(doadd)	<nine, modify, 0245>	PROCEDURE	8C6B
(dr)	<nine, modify, 0528>	STRING	8C2E
(getnsp)	<nine, modify, 0358>	PROCEDURE	8C8D
(gtnsp)	<nine, modify, 0351>	PROCEDURE	8C8C
(lowercase)	<nine, modify, 0266>	PROCEDURE	8C7C
(modify)	<nine, modify, 097>	EXT	8C2F
(pt)	<nine, modify, 0527>	TEXT POINTER	8C2C
(sdb)	<nine, modify, 094>	REF	8C2A
(sf)	<nine, modify, 095>	TEXT POINTER	8C2B
(xappgro)	<nine, modify, 0107>	PROCEDURE	8C3A
(xdelcol)	<nine, modify, 0160>	LOCAL	8C4A
(xdelstr)	<nine, modify, 0195>	LOCAL	8C5A
(xforsen)	<nine, modify, 0257>	LOCAL	8C7B
(xinstb)	<nine, modify, 0234>	PROCEDURE	8C6A
(xsubsp)	<nine, modify, 0325>	LOCAL	8C8A

Modify
The subsystem "Modify" contains five groups of commands that are extensions of the Base subsystem's editing commands. See also: writing. .Text[Null]="y >>> < NINE, MODIFY.NLS;13, >, 20-Apr-78 18:25 BLP ;;;; EXTERNAL LINKS: <nine, sysgd,>";
% First Searches branch. Index in <nls,>.
commands
##<modify>## 1A
Append Group/PLex (at) DESTINATION (join with) CONTENT OK
The Modify command "Append" will append all of the highest-level statements in the Group or Plex you specify to one another successively. If any of these statements have substatements, the sub-statements will be placed together in order under the new single statement. When the accumulated statements reach the maximum size (2000 characters), a new statement will begin. See Base Append. 2
Delete Column / Leading / Names ... 3
Column: Delete Column (of width beginning at) DESTINATION (through) DESTINATION OK
The Modify command "Delete Column" will delete the text between the two points you specify in that line. It will repeat this in the corresponding lines of all statements in the plex of the same level as the statement pointed to. 3A
Visible: Delete Visible (beginning in column of width) DESTINATION (through) DESTINATION OK
The Modify command "Delete Visible" will delete the text between the two points you specify in that line, only if the first character of the column begins a visible. It will repeat this in thee corresponding lines (where the condition is met) of all statements in the plex of the sae level as the statement pointed to; other lines in those statements will not be edited. 3A1
Leading: Delete Leading (spaces in) STRUCTURE (at) DESTINATION OK / [FILTER OK]
The Modify command "Delete Leading" will delete spaces that begin statements in the STRUCTURE you specify. See also: FILTER. 3B
Names: Delete Names (in) STRUCTURE (at) DESTINATION OK / [FILTER OK]
The Modify command "Delete Names" will delete the names that begin statements. See statement-names. See also: FILTER. 3C
Force (sentence case in) STRUCTURE (at) DESTINATION OK
The Modify command "Force" will change statements that are all uppercase to be in lowercase, leaving uppercase characters at the beginning of the statement and following spaces after a period, questionmark, or exclamation point (usually the first character of each sentence). 4
Insert Front/Back (of) STRUCTURE (at) DESTINATION (the text) CONTENT OK
The Modify command "Insert" augments the NLS Base subsystem's Insert command. 5
Front: Insert Front/Back (of) STRUCTURE (at) DESTINATION (the text) CONTENT OK
The Modify command "Insert" adds the characters you specify for CONTENT to the beginning (Insert Front) or end (Insert Back) of every statement in the STRUCTURE at the DESTINATION you specify. 5A
Filter option:
Before specifying the STRUCTURE to be modified, you may type <CTRL-U> to have the insertion operate through the FILTER viewspecs you specify.

Insert Front/Back (of) [Filtered: VIEWSPECS] STRUCTURE (at)
DESTINATION (the text) CONTENT OK 5A1
STRUCTURE: Statement, Branch, Group, or Plex
##<structure>## 5A2
Substitute (spaces in) [(filtered:) VIEWSPECS] STRUCTURE (at)
DESTINATION
(number of spaces between sentences) CONTENT
(number of spaces after colon) CONTENT
(number of spaces after semicolon) CONTENT
(number of spaces after comma) CONTENT OK
The Modify command "Substitute" will ensure that the number of spaces you specify for CONTENT occur after the punctuations indicated. "between sentences" means where a period, exclamation point, or questionmark is optionally followed by double quotes and is followed by some or no spaces and an uppercase letter. Periods after initials are not considered to end sentences. No spaces will be placed after commas that end links, nor after colons and commas that are surrounded by numbers. Specify zero for CONTENT where you do not want to change the spaces after a particular punctuation. Type your OPTION key to have the operation take place through a filter. 6
% syntax for modify commands % 7

Jump (to) DESTINATION CONFIRM 7B
Jump (to) Address (relative to) DESTINATION (modified by address) CONTENT CONFIRM 7C
Jump (to) Jump (to) Address DESTINATION CONFIRM 7D
Jump (to) Item DESTINATION CONFIRM 7E
Jump (to) Successor DESTINATION CONFIRM 7F
Jump (to) Predecessor DESTINATION CONFIRM 7G
Jump (to) Up DESTINATION CONFIRM 7H
Jump (to) Down DESTINATION CONFIRM 7I
Jump (to) Head DESTINATION CONFIRM 7J
Jump (to) Tail DESTINATION CONFIRM 7K
Jump (to) End (of Branch) DESTINATION CONFIRM 7L
Jump (to) Back DESTINATION CONFIRM 7M
Jump (to) Origin DESTINATION CONFIRM 7N
Jump (to) Next DESTINATION CONFIRM 7O
Jump (to) Link CONTENT CONFIRM 7P
Jump (to) Name Any CONTENT CONFIRM 7Q
Jump (to) Name First CONTENT CONFIRM 7R
Jump (to) Name Next CONTENT CONFIRM 7S
Jump (to) Name External CONTENT CONFIRM 7T
Jump (to) Return CONFIRM ANSWER 7U
Jump (to) Return CONFIRM ANSWER ANSWER 7V
Jump (to) File DESTINATION CONFIRM 7W
Jump (to) File Named CONTENT CONFIRM 7X
Jump (to) File Return CONFIRM ANSWER 7Y
Jump (to) File Return CONFIRM ANSWER ANSWER 7Z
Jump (to) Content First CONFIRM 7A@

Jump (to) Content First CONFIRM 7AB
Jump (to) Content First CONTENT CONFIRM 7AC
Jump (to) Content Next CONFIRM 7AD
Jump (to) Content Next CONTENT CONFIRM 7AE
Jump (to) Content Next CONFIRM 7AF

Jump (to) Content Next CONTENT CONFIRM 7AG
Jump (to) Word First CONFIRM 7AH
Jump (to) Word First CONTENT CONFIRM 7AI
Jump (to) Word First CONFIRM 7AJ
Jump (to) Word First CONTENT CONFIRM 7AK
Jump (to) Word Next CONFIRM 7AL
Jump (to) Word Next CONTENT CONFIRM 7AM
Jump (to) Word Next CONFIRM 7AN
Jump (to) Word Next CONTENT CONFIRM 7AO
Handle CONFIRM 7AP
Goto (subsystem) CONTENT CONFIRM 7AQ
Goto (subsystem) OPTION (subsystem name) CONTENT CONFIRM 7AR

Substitute (spaces in) OPTION (Filtered:) Group (at) DESTINATION (number of spaces between sentences) CONTENT (number of spaces after colon) CONTENT (number of spaces after semicolon) CONTENT (number of spaces after comma) CONTENT CONFIRM (substitute in progress) 7AT
Substitute (spaces in) OPTION (Filtered:) Branch (at) DESTINATION (number of spaces between sentences) CONTENT (number of spaces after colon) CONTENT (number of spaces after semicolon) CONTENT (number of spaces after comma) CONTENT CONFIRM (substitute in progress) 7AU
Substitute (spaces in) OPTION (Filtered:) Plex (at) DESTINATION (number of spaces between sentences) CONTENT (number of spaces after colon) CONTENT (number of spaces after semicolon) CONTENT (number of spaces after comma) CONTENT CONFIRM (substitute in progress) 7AV
Substitute (spaces in) Statement (at) DESTINATION (number of spaces between sentences) CONTENT (number of spaces after colon) CONTENT (number of spaces after semicolon) CONTENT (number of spaces after comma) CONTENT CONFIRM (substitute in progress) 7AW
Substitute (spaces in) Group (at) DESTINATION (number of spaces between sentences) CONTENT (number of spaces after colon) CONTENT (number of spaces after semicolon) CONTENT (number of spaces after comma) CONTENT CONFIRM (substitute in progress) 7AX
Substitute (spaces in) Branch (at) DESTINATION (number of spaces between sentences) CONTENT (number of spaces after colon) CONTENT (number of spaces after semicolon) CONTENT (number of spaces after comma) CONTENT CONFIRM (substitute in progress) 7AY
Substitute (spaces in) Plex (at) DESTINATION (number of spaces between sentences) CONTENT (number of spaces after colon) CONTENT (number of spaces after semicolon) CONTENT (number of spaces after comma) CONTENT CONFIRM (substitute in progress) 7AZ
Insert Front (of) OPTION (Filtered:) Statement (at) DESTINATION (the text) CONTENT CONFIRM 7B@
Insert Front (of) OPTION (Filtered:) Group (at) DESTINATION (the text) CONTENT CONFIRM 7BA
Insert Front (of) OPTION (Filtered:) Branch (at) DESTINATION (the text) CONTENT CONFIRM 7BB
Insert Front (of) OPTION (Filtered:) Plex (at) DESTINATION (the text) CONTENT CONFIRM 7BC
Insert Front (of) Statement (at) DESTINATION (the text) CONTENT CONFIRM 7BD
Insert Front (of) Group (at) DESTINATION (the text) CONTENT CONFIRM 7BE
Insert Front (of) Branch (at) DESTINATION (the text) CONTENT CONFIRM 7BF
Insert Front (of) Plex (at) DESTINATION (the text) CONTENT CONFIRM

```
                                                                    7BG
Insert Back (of) OPTION (Filtered:) Statement (at) DESTINATION (the
text) CONTENT CONFIRM                                               7BH
Insert Back (of) OPTION (Filtered:) Group (at) DESTINATION (the
text) CONTENT CONFIRM                                               7BI
Insert Back (of) OPTION (Filtered:) Branch (at) DESTINATION (the
text) CONTENT CONFIRM                                               7BJ
Insert Back (of) OPTION (Filtered:) Plex (at) DESTINATION (the text)
CONTENT CONFIRM                                                     7BK
Insert Back (of) Statement (at) DESTINATION (the text) CONTENT
CONFIRM                                                             7BL
Insert Back (of) Group (at) DESTINATION (the text) CONTENT CONFIRM
                                                                    7BM
Insert Back (of) Branch (at) DESTINATION (the text) CONTENT CONFIRM
                                                                    7BN
Insert Back (of) Plex (at) DESTINATION (the text) CONTENT CONFIRM
                                                                    7BO
Force (Sentence Case in) OPTION (Filtered:) Statement (at)
DESTINATION CONFIRM                                                 7BP
Force (Sentence Case in) OPTION (Filtered:) Group (at) DESTINATION
CONFIRM                                                             7BQ
Force (Sentence Case in) OPTION (Filtered:) Branch (at) DESTINATION
CONFIRM                                                             7BR
Force (Sentence Case in) OPTION (Filtered:) Plex (at) DESTINATION
CONFIRM                                                             7BS
Force (Sentence Case in) Statement (at) DESTINATION CONFIRM         7BT
Force (Sentence Case in) Group (at) DESTINATION CONFIRM             7BU
Force (Sentence Case in) Branch (at) DESTINATION CONFIRM            7BV
Force (Sentence Case in) Plex (at) DESTINATION CONFIRM              7BW
Delete Column (of width beginning at) DESTINATION CONFIRM           7BX
Delete Visible (beginning in column of width) DESTINATION CONFIRM
                                                                    7BY
Delete Names (in) Statement (at) DESTINATION OPTION (Filtered:)
CONFIRM                                                             7BZ
Delete Names (in) Statement (at) DESTINATION CONFIRM                7C@
Delete Names (in) Group (at) DESTINATION OPTION (Filtered:) CONFIRM
                                                                    7CA
Delete Names (in) Group (at) DESTINATION CONFIRM                    7CB
Delete Names (in) Branch (at) DESTINATION OPTION (Filtered:) CONFIRM
                                                                    7CC
Delete Names (in) Branch (at) DESTINATION CONFIRM                   7CD
Delete Names (in) Plex (at) DESTINATION OPTION (Filtered:) CONFIRM
                                                                    7CE
Delete Names (in) Plex (at) DESTINATION CONFIRM                     7CF
Delete Leading (spaces in) Statement (at) DESTINATION OPTION
(Filtered:) CONFIRM                                                 7CG
Delete Leading (spaces in) Statement (at) DESTINATION CONFIRM       7CH
Delete Leading (spaces in) Group (at) DESTINATION OPTION (Filtered:)
CONFIRM                                                             7CI
Delete Leading (spaces in) Group (at) DESTINATION CONFIRM           7CJ
Delete Leading (spaces in) Branch (at) DESTINATION OPTION
(Filtered:) CONFIRM                                                 7CK
Delete Leading (spaces in) Branch (at) DESTINATION CONFIRM          7CL
Delete Leading (spaces in) Plex (at) DESTINATION OPTION (Filtered:)
CONFIRM                                                             7CM
Delete Leading (spaces in) Plex (at) DESTINATION CONFIRM            7CN
```

```
Append Group (at) DESTINATION (join with) CONTENT CONFIRM                7CO
Append Plex (at) DESTINATION (join with) CONTENT CONFIRM                 7CP
% Source-code for Modify subsystem .IgB;                                    8
% Maintenance Index Branch.<<never sysguide>> <<never index>>  LF: 8A
   DESCRIPTION <modify>
   CHANGES
      Date and Time of change IDENT <link to change>
         Text descriptions of changes made to the tool or subsystem,
         each change note should include the name of the person
         making the change, the date, and a reason for the change,
         as well as the description of the change.
      11-JUL-77 KJM (nine,modify,xappgro)
         Changed to append only top-level (relative to the plex or
         group) statements, leaving substructure unappended and in
         its original order.
      21-Jun-77 16:05 LLG (nine, modify, xappgro)
         Fixed call to cis to be a call to cisstr. (CIS is a JSYS,
         used to be an 8.5 procedure)  This fix handles the problem
         with appending a group/plex and overrunning the 2000 char
         limit for a single statement.
      20-APR-77 14:04 ROM
         Changed calls to fltprc to accept destptr as a list
         (without having to make the translation at each xroutine.
            *** Note: The change in fltprc takes effect only in the
            <relnine> version of nls9, consequently modify was not
            compiled so the current code still works.
      10-JUN-76 12:09 KIRK <nine, modify, xinsfb>
         Changed to call fltprc with approporiate arguments.
      10-JUL-76 12:09 KIRK <nine, psedt2, fltprc>
         Moved from modify.
   GRAMMAR SOURCE <<default update new>> SF: <grammar>
      Processing History
         <<default update new>> SF: (white,dpsprm,)
      Update History
      Print History
      3 Includes;
      3 Conditionals;
      Compile <grammar> Using <sysnine, cgcml,> To <sysnine,
      modify.gram,>
   GRAMMAR COMPILED <nsw-sources, modify.cml,>
   GRAMMAR COMPACTED <arcsubsys, modify.cgr,>
   PARSE FUNCTIONS SOURCE <nine, nls-parse.nls,>
      Processing History
         <<default update new>> SF: (white,dpsprm,)
      Update History
      Print History
      Conditionals;
      Compile <nine, nls-parse.nls, 1> Using <sysnine, xl10,> To
      <sysnine, modify.pfc,>
   PARSE FUNCTIONS COMPILED <nsw-sources, modify.pfc,>
   BACKEND SOURCE <<default update new>> SF: <be>
      Processing History
         <<default update new>> SF: (white,dpsprm,)
      Update History
      Print History
      Conditionals;
```

```
      Compile <be> Using <sysnine, x110,> To <sysnine,
      modify.subsys,>
   BACKEND COMPILED <arcsubsys, modify.subsys,>
   USER DOCUMENTATION <modify> <modify.print,> <modify.txt,>
   MAINTENANCE DOCUMENT
      links to documents describing the internal procedures and data
      structures of this tool or subsystem, this documentation is
      intended to be useful to persons debugging or upgrading the
      tool or subsystem.
         File Naming Conventions
            toolname-MAINTDOC.NLS
   INTERFACE DOCUMENT
      links to documents describing the interfaces to this tool or
      subsystem, especially any data structures shared with other
      tools or subsystems.
         File Naming Conventions
            toolname-INTERFACEDOC.NLS
   RUNFILES
      links to RUNFILES that are useful in building new versions of
      this tool or sybsystem
         File Naming Conventions
            toolname-function.RUN
% GRAMMAR SOURCE <arcsubsys, cgcml,> <arcsubsys, cgcml9,>  <relnine,
modify.cml,>  %                                                    8B
   FILE cmodify
   % COMPILE INSTRUCTIONS %                                       8B2
      INCLUDE <nine, nls-grammar, flags !subsystems>             8B2A
   % DECLARATIONS %                                              8B3
      INCLUDE <nine, nls-grammar, declarations !universal>       8B3A
      DECLARE FUNCTION % subsystem xroutines %                   8B3B
         xappgro, xdelcol, xdelstr, xforsen, xinsadr, xinsfb,
         xsubsp;                                                8B3B1
      DECLARE COMMAND WORD %Modify subsystem command words%      8B3C
         "FRONT" = 60,
         "COLUMN" = 97,
         "LEADING" = 98,
         "NAMES" = 99;
      DECLARE VARIABLE sntsp, clnsp, smisp, cmasp;               8B3D
   % COMMON RULES %                                              8B4
      INCLUDE <nine, nls-grammar, rules !universal>
   % COMMANDS % SUBSYSTEM modify KEYWORD "MODIFY"
      append COMMAND =                                           8B5A
         "APPEND" sent _ ( "GROUP" / "PLEX" ) <"at">
         dest _ DSEL(sent) <"join with">                        8B5A2
         source _ LSEL(#"TEXT")                                 8B5A3
         CONFIRM                                                8B5A4
          xappgro( sent, dest, source);
      delete COMMAND =                                           8B5B
         "DELETE"
            dent _ NULL                                       8B5B1A
            sent _ NULL                                       8B5B1B
            ( "COLUMN"  <"of width beginning at">
               sent _ #"COLUMN" coltxt                        8B5B1C1
            / "VISIBLE"  <"beginning in column of width">
               sent _ #"VISIBLE" coltxt                       8B5B1D1
            / "NAMES" <"in"> sent _ #"NAMES" filtstruc
```

```
        / "LEADING" <"spaces in"> sent _ #"LEADING" filtstruc
        ) ;
      coltxt =                                                    8B5B2
        dest _ DSEL(#"TEXT")                                      8B5B2A
        CONFIRM  xdelcol(sent,dest);                              8B5B2B
      filtstruc =                                                 8B5B3
        vs _ NULL                                                 8B5B3A
        dent _ structure <"at">                                   8B5B3B
        dest _ DSEL(dent)                                         8B5B3C
        [OPTION <"filtered:"> vs _ viewspecs()]
        CONFIRM  xdelstr(dent,dest,sent,vs);                      8B5B3E
    force COMMAND =                                               8B5C
      "FORCE" <"sentence case in">
      vs _ NULL                                                   8B5C2
      [OPTION <"filtered:"> vs _ viewspecs()]
      dent _ structure                                            8B5C4
      <"at">  dest _ DSEL(dent)
      CONFIRM                                                     8B5C6
       xforsen(dent, dest, vs);
    insert COMMAND = "INSERT"                                     8B5D
      sent _ ( "FRONT" / "BACK" )   <"of">                        8B5D1
        vs _ NULL                                                 8B5D1A
        [ OPTION  <"filtered:"> vs _ viewspecs() ]
        dent _ structure                                          8B5D1C
        <"at">  dest _ DSEL(dent)
        <"the text">  source _ LSEL(#"TEXT")
        CONFIRM                                                   8B5D1F
         xinsfb(sent, source, dent, dest, vs)
      ;
    substitute COMMAND =                                          8B5F
      "SUBSTITUTE" <"spaces in">
      vs _ NULL                                                   8B5E2
      sntsp _ NULL                                                8B5E3
      clnsp _ NULL                                                8B5E4
      smisp _ NULL                                                8B5E5
      cmasp _ NULL                                                8B5E6
      ( OPTION  <"filtered:"> vs _ viewspecs() dent _
      notstatement
      / dent _ structure )
      <"at">  dest _ DSEL(dent)
      CLEAR <"number of spaces between sentences"> sntsp _
      LSEL(#"NUMBER")                                             8B5E10
      CLEAR <"number of spaces after colon"> clnsp _
      LSEL(#"NUMBER")                                             8B5E11
      CLEAR <"number of spaces after semicolon"> smisp _
      LSEL(#"NUMBER")                                             8B5E12
      CLEAR <"number of spaces after comma"> cmasp _
      LSEL(#"NUMBER")                                             8B5E13
      CONFIRM                                                     8B5E14
      CLEAR <"substitute in progress">                            8B5E15
       xsubsp(sntsp, clnsp, smisp, cmasp, dent, dest, vs) ;
    INCLUDE <nine, nls-grammar, commands !universal>              8B5F
    END.                                                          8B5G
FINISH
```

```
                                                                    8B6
% BE SOURCE %                                                        8C
   FILE modify % <arcsubsys, x110,> to  <relnine, modify.subsys,>
            or <arcsubsys, 1109,> to <relnine, modify.subsys,> %
   % Declarations %                                                 8C2
      (sdb) REF ;                                                  8C2A
      (sf) TEXT POINTER;                                           8C2B
      (pt) TEXT POINTER;                                           8C2C
      (dl) STRING [1];                                             8C2D
      (dr) STRING [1];                                             8C2E
      (modify) EXTERNAL _ (                                        8C2F
         $"XAPPGRO", $xappgro,
         $"XDELCOL", $xdelcol,
         $"XDELSTR", $xdelstr,
         $"XINSFB", $xinsfb,
         $"XFORSEN", $xforsen,
         $"XSUBSP", $xsubsp,
         0,0);
   % appends a group or plex of statements to-gether.%
      (xappgro)PROCEDURE                                           8C3A
         (sent,                % front/back %
         destptr REF,                  % location of structure %
         srceptr REF);
         % DECLARATIONS %
            LOCAL source REF, dest REF;
            LOCAL from, first, last;
            LOCAL TEXT POINTER ptr1, ptr2, fptr1, fptr2;
            LOCAL STRING text[1000], str1[4001], str2[4001],
            nullstr[6] ;
         IF &srceptr THEN &source _ ELEM #srceptr#[tppair];
         IF &destptr THEN &dest _ ELEM #destptr#[tppair];
         dpset(dspallf, dest, endfil, endfil);
         % get begin and end of structure %
            first _ dest; last _ [&dest+d2sel];
            IF first = last THEN
               BEGIN
               dismes(1, $"That's already one statement!");
               RETURN;
               END;
         % append all statements %
            DO
               BEGIN
               from _ getsuc(first);
               *str1* _ SF(from) SE(from);
               *str2* _ SF(first) SE(first);
               IF ( ((str1.L) + (str2.L) + (text.L)) > 1999 )
                  THEN
                     BEGIN
                     %insert new statement%
                        from _ cisstr(first, $"", down) ;
                     %move substructure to urder new statement%
                        IF getsub(first) # first THEN %substructure%
                           cmovgro(from, levdown,
                           gethed(getsub(first)),
                           getail(getsub(first)), 0);
                     %append a statement on%
```

```
                        first _ from ;
                        from _ getsuc(first);
                        nullstr.L _ 0; %to fake out cappsta%
                        FIND SF(*nullstr*) ^fptr1 SE(*nullstr*)
                        ^fptr2;
                        cappsta(first, from, $fptr1, $fptr2);
                     %delete initial space that insert statement put
                     on%
                        ptr1 _ first;
                        ptr1[1] _ 1;
                        cdeltex($ptr1, $ptr1, TRUE);
                     END
                  ELSE
                     BEGIN
                     cappsta(first, from, &source, &source+d2sel,
                     1);
                     END;
               END
            UNTIL from=last;
         RETURN;
         END.

   % delete column %
      (xdelcol)                                                    8C4A
         % deletes bugged column, assuming columnar format
            Buffer pages required: 1
            Execute. Bug beginning and end of column. It will ask
            if you want it to do the delete only if it finds a
            visible beginning in the column. If you answer yes, it
            will only do the delete if it finds a visible beginning
            at the first bug mark. If you answer no, it will delete
            the character positions between and including the bug
            marks no matter what. It works on a plex.
            Author: NDM %
         PROCEDURE
            (dent REF,                  % entity selection %
            destptr REF);               % text selection %
         LOCAL dest REF;
         LOCAL TEXT POINTER col1, col2;
         IF &destptr THEN &dest _ ELEM #destptr#[tppair];
         dpset(dspallf, dest, endfil, endfil);
         % get begin and end of text %
            col1 _ dest; col1[1] _ dest[1];
            col2 _ dest[d2sel]; col2[1] _ dest[d2sel+1];
         % go to head of plex %
            col1 _ col2 _ gethed(col1);
         IF ELEM#dent#[cwtype] = 4 THEN
            LOOP % for each visibile in statement in plex, delete
            column %
               BEGIN
               IF FIND col1 < (NP/-CH) > col1 PT  THEN
                  ST col1 col2 _ NULL ;
               IF getftl (col1) THEN EXIT LOOP;
               col1 _ col2 _ getsuc(col1) ;
               END
         ELSE % for each statement in plex, delete column %
```

```
            LOOP
               BEGIN
               ST col1 col2 _ NULL ;
               IF getftl (col1) THEN EXIT LOOP;
               col1 _ col2 _ getsuc(col1) ;
               END;
         RETURN;
         END.
   % delete names and leading spaces %
      (xdelstr)                                                    8C5A
         PROCEDURE
            (dent,            % structure type %
            destptr REF,            % structure selection %
            sent REF,               % names / leading (spaces %
            vs
            );
         CASE ELEM #sent#[cwtype] OF
            = 99: % names % fltprc(dent, &destptr, vs, $delnam);
            = 98: % Leading spaces % fltprc(dent, &destptr, vs,
            $delsp)
            ENDCASE ABORT (badarg, $"
            Internal error: the command word number doesn't
            match.");
         RETURN;
         END.
      (delname) PROCEDURE (stid) ; % deletes statement names from
      structure %                                                  8C5B
         LOCAL rdl, ldl;
         FIND SF(stid);
         IF getnmf(stid) THEN %has name to be deleted%
            BEGIN
            ldl _ getnmdl( stid : rdl);
            *dl* _ ldl;
            *dr* _ rdl;
            IF (ldl=0) OR (ldl=SP AND FIND 1$SP) OR (FIND $SP *dl*)
            THEN %found left name delimiter%
               BEGIN
               FIND $SP $(LD / '- / '' ); %scan over name%
               IF (rdl=0) OR (rdl=SP AND FIND SP) OR (FIND $SP *dr*)
               THEN %found name and right delimiter%
                  BEGIN
                  FIND $NP ^pt;
                  ST pt _ pt SE(pt) ;
                  END;
               END;
            END;
         RETURN;
         END.
      (delsp) PROCEDURE(stid) ;  % deletes leading spaces from
      structure %                                                  8C5C
         IF FIND SF(stid) 1$SP ^pt THEN
            ST pt _ pt SE(pt) ;
         RETURN;
         END.
   % inserts given text to front/back of statements.%
      (xinsfb) PROCEDURE (                                         8C6A
```

```
        sent REF,               % front/back %
        source REF,                     % typed-in text %
        dent REF,               % type of structure entity %
        destptr REF,                    % location of structure %
        vs REF );               % filter viewspecs %
        (st) STRING [2048];                                          8C6A6
        getpstring(&source, $st);
        fltprc(&dent, &destptr, &vs, $doadd, $st, &sent);
        RETURN;
        END.

    (doadd)PROCEDURE(stid,text REF,bckfrt REF); %adds text to
    statement%                                                       8C6B
        (tp) TEXT POINTER;                                           8C6B1
        tp _ stid;
        tp[1] _ 1;
        IF ELEM #bckfrt#[cwtype] = 59 THEN FIND SE(tp) ^tp ;
        ST tp tp _ *text*;
        RETURN(TRUE);
        END.
  % Force Sentence Case %
    % recovers from an erroneous Set Case
        Buffer pages required:  2
        Content analizer program.  Reformats leaving first printing
        char of statement and after periods uppercase, forces other
        alphabetics to lower case.                                   8C7A2
        Author:  CHI %
    (xforsen)                                                        8C7B
        PROCEDURE
            (dent,              % structure %
            destptr REF,                % text selection %
            vs
            );
        fltprc(dent, &destptr, vs, $lowercase) ;
        RETURN;
        END.
    (lowercase) PROCEDURE (sw);                                      8C7C
        LOCAL char;
        LOCAL TEXT POINTER stid;
        LOCAL STRING str[2000];
        *str* _ NULL;
        FIND SF(sw) ^stid;
        LOOP
            BEGIN
            %look for the first letter, make it upper case.%
                CASE char _ READC OF
                    = ENDCHR: EXIT LOOP;
                    = L: IF char IN ["a, "z] THEN *str* _ *str*, char
                    - 40B
                        ELSE *str* _ *str*, char;
                    ENDCASE
                        BEGIN
                        *str* _ *str*, char;
                        REPEAT CASE;
                        END;
            %make the rest lower case, until find a punctuation. %
```

```
            CASE char _ READC OF
               = ENDCHR: EXIT LOOP;
               IN ["A, "Z]:
                  BEGIN
                  *str* _ *str*, char + 40B;
                  REPEAT CASE;
                  END;
               = ".:
                  BEGIN
                  *str* _ *str*, char;
                  CASE char _ READC OF
                     = D:
                        BEGIN
                        *str* _ *str*, char;
                        REPEAT CASE;
                        END;
                     ENDCASE;
                  *str* _ *str*, char;
                  REPEAT LOOP;
                  END;
               = "?, = "!:
                  BEGIN
                  *str* _ *str*, char;
                  REPEAT LOOP;
                  END;
               ENDCASE
                  BEGIN
                  *str* _ *str*, char;
                  REPEAT CASE;
                  END;
         END;
      ST stid _ *str*;
      RETURN;
      END.
% substitute the given number of spaces after punctuation %
   (xsubsp)  % substitutes the number of spaces passed.  Empty
   string means "leave alone". %                                    8C8A
      PROCEDURE
         (sensp,          % number of spaces between sentences %
         colsp,           % after colon %
         semsp,           % after semicolon %
         comsp,           % after coma %
         dent,            % type of structure entity %
         destptr REF,              % location of structure %
         vs);                      % filter viewspecs %
      LOCAL dest REF;
      LOCAL STRING sentsp[11], colnsp[11], semisp[11], comasp[11]
      ;
      % copy strings to minimize page faulting  ? %
         <cpspst>(sensp,$sentsp,colsp,$colnsp,semsp,$semisp,
         comsp, $comasp) ;
      % get number of spaces to insert %
         <gtnsp>($sentsp, $colnsp, $semisp, $comasp);
      % do it %
         <fltprc>(dent, &destptr, vs, $csubsp, $sentsp, $colnsp,
         $semisp, $comasp, 0);
```

```
   RETURN;
   END.

(cpspst) PROC (sensp, sentsp,colsp, colnsp,semsp,
semisp,comsp, comasp) ;                                          8C8B
   <getpstring>(sensp, sentsp);
   <getpstring>(colsp, colnsp);
   <getpstring>(semsp, semisp);
   <getpstring>(comsp, comasp);
   RETURN;
   END.

(gtnsp) PROCEDURE (sentsp, colnsp, semisp, comasp) ;             8C8C
   <getnsp>(sentsp);
   <getnsp>(colnsp);
   <getnsp>(semisp);
   <getnsp>(comasp);
   RETURN;
   END.

(getnsp) PROCEDURE (astrng) ; % given an address to a string
containing a number, returns with the string containing that
number of spaces %                                               8C8D
   LOCAL cnt, num;
   REF astrng;
   IF astrng.L THEN
      BEGIN
      num _ VALUE(&astrng);
      astrng.L _ 0;
      cnt _ 0;
      WHILE cnt < num DO
         BEGIN
         *astrng* _ *astrng*, SP;
         BUMP cnt;
         END;
      END
   ELSE *astrng* _ "no change" ;
   RETURN;
   END.

(csubsp)  % substitutes the number of spaces in the statement
at stid for numbers passed.  Empty string means "leave alone".
%                                                                8C8F
   PROCEDURE
      (stid,            % location of statement %
      sentsp,           % number of spaces between sentences %
      colnsp,           % after colon %
      semisp,           % after semicolon %
      comasp,           % after coma %
      param5);                % extra param because called
      from fltprc %
   LOCAL TEXT POINTER ptr1, ptr2, ptr3;
   REF sentsp, colnsp, semisp, comasp;
   ptr1 _ stid;
   ptr1[1] _ 1;
   IF *sentsp* # "no change" AND FIND ptr1 > [UL] ^ptr1 THEN
```

```
        LOOP
           BEGIN
           IF NOT FIND ptr1 > [".!"?!"!] $1"" ^ptr2 ^ptr1 THEN EXIT
           LOOP;
           IF FIND ptr1 *sentsp* -SP THEN REPEAT LOOP; % already ok
           (may be just as fast to leave this out) %
           IF NOT FIND $SP ^ptr3 UL THEN REPEAT LOOP; %not followed
           by sentence%
           IF FIND ptr1 < CH L NP THEN REPEAT LOOP; %initial%
           IF FIND ptr1 < $NP "; THEN REPEAT LOOP; % directive%
           IF FIND ptr1 < CH > ". UL $11L (";/""/"=) THEN REPEAT
           LOOP; % directive we assume %
           ST ptr2 ptr3 _ *sentsp*;
           END;
        ptr1[1] _ 1;
        IF *colnsp* # "no change" THEN LOOP
           BEGIN
           IF NOT FIND ptr1 > [":] ^ptr2 ^ptr1 $SP ^ptr3 THEN EXIT
           LOOP;
           IF FIND ptr1 > (*colnsp* -SP)/(NP AND -SP)/ENDCHR THEN
           REPEAT LOOP;
           IF FIND ptr1 > D < 2CH D THEN REPEAT LOOP; % a time %
           ST ptr2 ptr3 _ *colnsp*;
           END;
        ptr1[1] _ 1;
        IF *semisp* # "no change" THEN LOOP
           BEGIN
           IF NOT FIND ptr1 > [";] ^ptr2 ^ptr1 $SP ^ptr3 THEN EXIT
           LOOP;
           IF FIND ptr1 > ($NP ".) / (*semisp* -SP)/(NP AND
           -SP)/ENDCHR THEN REPEAT LOOP;
           ST ptr2 ptr3 _ *semisp*;
           END;
        ptr1[1] _ 1;
        IF *comasp* # "no change" THEN LOOP
           BEGIN
           IF NOT FIND ptr1 > [",] ^ptr2 ^ptr1 $SP ^ptr3 THEN EXIT
           LOOP;
           IF FIND ptr1 > (*comasp* -SP)/(NP AND -SP)/ENDCHR/">/")
           THEN REPEAT LOOP;
           IF FIND ptr1 > "" THEN REPEAT LOOP;   % dialogue %
           IF FIND ptr1 > D < 2CH D THEN REPEAT LOOP;  % a number %
           ST ptr2 ptr3 _ *comasp*;
           END;
        RETURN(TRUE);
        END.

FINISH
```

Sendmail subsystem

The Sendmail subsystem allows you to send messages and documents to a list of people known to NLS and have these messages cataloged and stored in the NLS Journal. The recipients may receive hardcopy or notice of the item in their initial file (with a link to allow immediate online access), or the item itself if it is short.

% FIRST SEARCHES Index in <nls,>

subcollection 1A

the Sendmail Subcollections command

##<commands !subcollection>##

a Journal subcollection

##<subcollection>##

subcollections 1B

the Sendmail Subcollections command

##<commands !subcollection>##

a Journal subcollection

##<subcollection>##

submission

##<sendmail !item>## 1C

hoW to use the Sendmail subsystem 2

Journal (stored Sendmail items)

The Sendmail Journal is a data base of recorded items sent in the Sendmail subsystem. Each Journal item is given a unique number that names and identifies it for future reference. When you send or receive a Journal item, you will find a citation in your initial file. See also: how, Base Output Journal. 2A

citation

The Journal citation is the information the Sendmail subsystem places in your initial file when someone sends you mail. This citation usually consists of the following: In the first line is the author(s) of the item sent, followed by the time and date it was sent, and its Journal number. In the second line may appear a title. The third contains the location of the item, with a link to the Journal file where it is stored. Following this may be a notification about its distribution (e.g. a row of stars with the word ACTION embedded in it). If the item sent to you is over 1000 characters, this will be all you will see, unless there is a Comment (on the item being sent). If the item is shorter, the whole item will appear in your file preceded by the word "Message". 2A1

% backlinks: <base, print !journal>

author

The author of a Sendmail item is usually specified by the IDENT of the person logged in to distribute the mail. If you use "Interrogate," the author will be the person who logged in unless you specify otherwise. Other authors can be substituted or added with the Sendmail Authors command. After you have mailed an item that you authored, a citation to it will appear in your initial file under a branch with the statement name author. This is your automatic bibliography of all the items you "published" in the Journal via the Sendmail subsystem. 2A2

Sendmail Authors command

##<commands !authors>##

distribution

The list of people (their IDENTS) to which Sendmail items were sent via Sendmail's Distribute command. Secondary distribution (to people not on the original IDENTLIST) is done with the Sendmail Forward command. See Distribute. 2A3

IDENT

a string of 2-4 characters that identifies you to NLS and is recorded with your address in a master file. It is often your initials and can be the same as your EXEC password, but they are keys to different systems. IDENTs designate Journal authorship and distribution. To discover the IDENT of an NLS user, see last-name search and content-search, below. If you have no ident or if your ident information needs updating, send your request via Sendmail or SNDMSG to FEEDBACK (at office-1). See also: initial, Show Record. 2A3A

GROUP-IDENT

A GROUP-IDENT is a shorthand way of specifying many users' IDENTs. Several users who are working together may form a group that has one single IDENT; items sent to the GROUP-IDENT will then go to all the members. A GROUP-IDENT may be any length. See also: subcollection. 2A3A1

IDENTLIST

a list of IDENTs separated by commas. In Sendmail distributions, an IDENT may be immediately followed (no spaces) by a parenthetical comment which will only go to that IDENT, i.e., IDENT(comment). The parenthetical comment may not include a period. See also: Sendmail Distribute. 2A3A2

last-name search: .lastname, .last...

A last-name search may be made whenever you are prompted for an IDENT in a command. (It is frequently used in the Sendmail Show Record command.) If you type in a period and then a last name, the Sendmail subsystem will find the idents for all people with that last name. The beginning characters of a last name preceded by a period and followed by three periods will find the idents for all last names beginning with those characters. If only one ident satisfies your request, you will be shown this ident and asked if it is the correct one. Typing a <CA> will put this ident into the command. When an IDENTLIST is called for, any last-name search must have a comma separating it from the next item in the IDENTLIST. Typing a <CTRL-O> will stop an ident search at any time. For another kind of ident search, see content. 2A3A3

content search: "TYPEIN"

You can do a content search in the IDENT files whenever you are prompted for ident in a command. A string of text enclosed in double quotes ("TYPEIN") causes a search for all of the Users, Groups, or Organizatons (in that order) that have that text in their name. (Capitalization does not matter in your TYPEIN.) <CTRL-O> stops the search of the Users to begin the search of the Groups and stops the search of the Groups to begin the search of organizations. For another kind of ident search, see last-name. 2A3A4

restricted distribution

In using the Sendmail subsystem, when the IDENT of a group is used for distribution or for the author of a journal item, you may restrict distribution to the co-ordinator of the group by typing an ampersand (&) before the ident. See expanded distribution. 2A3B

expanded distribution

In using the Sendmail subsystem, when an IDENT belongs to a group, the group may by default have restricted distribution. That is, items sent to or from it are sent only to the co-ordinator. In either case you may distribute an item to the whole group by putting an uparrow (^) before the ident. See restricted distribution. 2A3C

files

Any recorded piece of mail is kept in the Journal files. The contents of Journal files are never changed. 2A4

ITEM

A Journal or Sendmail ITEM may originate as a typed in message, a file, a STRUCTURE from a file, or an offline (hardcopy) item. However, it always ends up as either a message (under 1000 characters) or a document (over 1000 characters). 2A5

Message

Message is used in two senses in the Sendmail subsystem. A Message in a Journal citation in your initial file is any

Sendmail item (other than a Comment) under 1000 characters. For example, if a file or branch from a file was sent, but it does not exceed 1000 characters, it will appear in your initial file under the heading "Message:". (Items over 1000 are called documents.) Message is also the name of a Sendmail command that allows you to type in text as your Sendmail item. See also: document. 2A5A

Sendmail Message command
##<commands !message>##

Document
any item over 1000 characters long in the Journal or Sendmail subsystem. It is stored as a file in the Journal directory that has as its FILENAME its Journal number. See also: message. 2A5B

STRUCTURE
##<nls, structure>## 2A5C

file
##<sendmail !file>##

offline
##<sendmail !offline>##

% backlinks: <base, print !journal>

header
The text that is automatically created at the beginning of each recorded item sent via Sendmail. This header contains the following: author(s) IDENT, date and time item was sent; Journal number; title (if any); full name of author(s); distibution list; subcollections; the AccessList (if any); and many output processor directives followed by the origin statement of the source if the item was an entire file. See also: accesslist, origin statement. 2A6

Journal inserted directives and those in your Journal item
There is a possible user hazard regarding directives and the Journal worth mentioning here. Although the Journal system does not tamper with any directives in your file, caution should be exercised that there are no print directives in a file that would contradict those added by the Journal in the origin statement. It is recommended that the user delete or edit conflicting directives so as not to detract from the format of the hardcopy of the Journal document being entered. Directives added by the Journal are: HJournal, H1, IGD=0, SNF=HJRM, RM=HJRM-7, PN=-1 and YBS. Thus the greatest potential conflicts are in headers, statement number format, right margin, and lines between statements. See publication directives.

numbering system
The Sendmail number system consists of a single sequential list of numbers starting from 1 and counting to about 34000 as of January, 1976. There are many overlapping subsets of these numbers (some subcollections have secondary numbering sequences) that have various ambiguous and synonymous names in common usage. All of the numbers in the sequential list are called Catalog Numbers. Journal Numbers include only items that have been entered via the Sendmail subsystem. In addition, there are XDOC numbers referring to external documents that have been given a number but have not been entered via the Sendmail subsystem. You may reserve a Journal number(s) to be used at a later time with

the Sendmail Reserve command. Reserved Journal numbers are also called "Preassigned" numbers. See also: catalogs, subcollections. 2A7

getting a number

The Sendmail Reserve command gives you a preassigned number, which can then be used at a later time. Using the Number command shows you the number for the item you are currently sending in the Sendmail subsystem. If you have not reserved a number, the Sendmail subsystem automatically numbers your item when it is sent. For numbering XDOCs, see XDOC. See also: Number, Reserve. 2A7A

using a preassigned number

##<number !previously>##

catalogs and indices

online and offline sources for finding Sendmail items recorded in the NLS Journal and XDOC collections. There are three kinds of Journal indices: Author, Number, and Title-word. Jump (to) Link <userguides,locator,journal.2> to access the indexes. See also: numbering. 2A8

subcollection

A Journal subcollection is a subset of Sendmail items identified by a special IDENT. It is usually a GROUP-IDENT for an organizational or special interest group (e.g. the DPCS subcollection is for the Documentation Production and Control Systems Interest Group. Sendmail items sent to this IDENT will be placed in the DPCS subcollection.) See also: group-ident. 2A9

XDOC (acronym for external document)

The term XDOC is used to distinguish offline documents from online documents journalized via Sendmail. You may reserve XDOC numbers for items that will be indexed in the ARC Catalog system, but will not appear as an online file in the Journal. See also: Reserve, Offline, numbering. 2A10

% backlinks: <base, output !journal>

reading your mail

Mail you receive through Sendmail appears in your initial file in the form of citations in a branch named Journal. New citations are added at the top for quicker viewing of the most recent ones. The message "You have new Journal mail" appears when you login if mail has been delivered to you since you last logged in. In DNLS use the Jump commands to read your mail. To read it in TNLS, see below. See also: initial, SNDMSG. 2B

to read your mail in TNLS

There are two basic methods of reading your mail in TNLS. In the first, use the Print Journal command. This will print at your terminal each citation in the branch named Journal, followed by the complete item. It is a recommended practice to create a branch in your initial file to which Journal items can be moved as they have been read. Then the Print Journal command will only get new items. The second method of reading your mail provides more control over what you see. Use the Print Branch command, with the statement name "journal" for your ADDRESS. This prints each citation in your Journal branch and enables you to choose which of the longer items not appearing in the Journal branch you wish to read. Use the Jump (to) Address command, with the statement number followed by ".1" for your ADDRESS. You can then

read the full document with the Print commands. See position for an explanation of ".1" ("link").

sending mail

You may use individual Sendmail commands to send an item via the Sendmail subsystem, or you may use the Sendmail Interrogate command, which automatically begins six different commands for you. See commands. 2C

Interrogate command

##<sendmail !interrogate>##

sources for sendmail items

##<sendmail !item>## 2C2

Use Initialize when you get an error message in Sendmail.

##<initialize>##

forms for sending mail

You may insert a Sendmail form listing Sendmail commands into a file where they can be modified and used repeatedly as a template with the Base Insert Sendmail (form) command.

##<base, insert !sendmail !forms>## 2C4

workfile

a Sendmail file that is ordinarily invisible and unreadable in NLS, containing the current status of your Sendmail item. If you have completed sending an item, this file is empty. To clear it out, use the Sendmail Initialize command. See also: Show Status. 2C5

commands in the Sendmail subsystem

Commands in the Sendmail subsystem allow you to send ITEMS to anyone on the system having an IDENT, assign unique Journal numbers, and perform related services to make online communication flexible and efficient. See also: reading, sending. 3

Authors CONTENT OK

The Sendmail subsystem command "Authors" lets you name the author(s) of a Sendmail item. Specify the IDENTs of the author(s) of the item for CONTENT. If you don't use this command, you will be the author. 3A

Branch (at) SOURCE OK

The Sendmail subsystem command "Branch" allows you to send an NLS branch. For SOURCE, specify the Branch you are sending with an ADDRESS, BUG (or optional TYPEIN). VIEWSPECS do not affect the branch being sent. 3B

Comment CONTENT OK

The Sendmail command "Comment" allows you to enter a comment as part of your Journal item. It will be attached to the citation you send. The comment appears as page one of the hardcopy item and is the last thing in its online header. See also: citation, header. 3C

to enter comments for specific idents using the Distribute command

##<distribute !special>## 3C1

Distribute (for) CATEGORY (to) CONTENT OK

CATEGORY = Action or Information (only)

The Sendmail command "Distribute" allows you to specify to whom your mail will be sent and whether it will be sent for action or for information only. If you specify the command word Action for CATEGORY, your item will be marked with the word ACTION; if you specify Information, no special words will be attached to it. CFor CONTENT, provide a list of IDENTS separated by commas to show who will receive copies of your message or copies of a citation with a

link (if a document). See distribution. 3D
special distribution features
In the Sendmail Distribute command, an expression in parentheses typed immediately after an IDENT (that is with no spaces) will serve as a brief message to that recipient only. You may not use a period (.) within the parentheses. The default distribution list for a GROUP-IDENT may be overridden to limit it to the coordinator or expand it to the whole group by preceding the IDENT with & or ^, respectively. See expanded, restricted. 3D1

Expedite OK
The Sendmail subsystem command "Expedite" marks expedite on an item, but at present nothing in NLS reads that mark. The plan is to deliver such marked items immediately. 3E

File DESTINATION OK
The Sendmail command "File" allows you to specify a file as the material to send. In both TNLS and DNLS, an <OK> will send the file in which you are currently located; or you may type in or bug an ADDRESS for DESTINATION. Any recorded piece of mail is kept in the Journal files. The contents of Journal files are never changed. 3F

Forward (item number) CONTENT (to) CONTENT OK
The Sendmail command "Forward" allows you to re-send a Journal item with the number you specify for the first CONTENT to the IDENTs you specify for the second CONTENT. See also: numbering, Number, Distribute. 3G

Goto SUBSYSTEM OK
##<nls, goto>## 3H

Group (from) SOURCE OK
The Sendmail command "Group" allows you to send a Group of branches or statements. Specify the Group to be mailed by pointing to the first and last statement for SOURCE. VIEWSPECS do not matter. 3I

Initialize (specifications) OK
The Sendmail command "Initialize" resets all of the Sendmail commands. It places your IDENT as author and deletes everything from the other commands. This is done automaically every time you enter the Sendmail subsystem. Use this command if you receive error messages while using Sendmail. 3J

Insert Link/Mailing/Record/Status... 3K
Link: Insert Link (to follow) DESTINATION OK
The Sendmail subsystem command "Insert Link" causes the system to create and place a link to the Sendmail item you are working on at the DESTINATION you specify if you have the right to write on the file. No link will be placed if the Number command has not been used to assign a number. Since Journal files cannot be altered, this command will not place a link in a Journal file. See also: numbering. 3K1
Mailing: Insert Mailing (address for ident(s)) CONTENT (to follow statement at) DESTINATION LEVEL-ADJUST OK
The Sendmail command "Insert Mailing" places the mailing address for each ident you specify for CONTENT at the location you point to for DESTINATION. If you are using more than one ident, they must be separated by a space. Each address will be placed in your file as a separate statement. 3K2
Record: Insert Record (for ident) CONTENT (as a statement to follow) DESTINATION LEVEL-ADJUST OK

The Sendmail command "Insert Record" will insert a statement containing all of the information in the Identfile about the person(s) whose ident(s) you specify for CONTENT. The information provided is the same you can see with the Show Record command, and includes items such as on- and off-line mailing addresses and ident groups to which the person belongs. 3K3

Status: Insert Status (form to follow) DESTINATION LEVEL-ADJUST OK

The Sendmail command "Insert Status" will insert a statement containing the current status of your Sendmail item at the DESTINATION you specify. You can modify this statement to contain the Sendmail commands and/or text appropriate for your Sendmail item. When this form is edited to your satisfaction, use the Sendmail Process (command form) command to automatically execute every command in the form. To insert a complete empty command form, use the Base Insert Sendmail (form) command. Note: If you Quit out of the Sendmail subsystem sometime before you Process your status form, you must re-specify the source of your item. See Warning, below, before sending a STRUCTURE. See also: editing. 3K4

to insert a form listing all commands, use Insert Sendmail (form)

##<base, insert !sendmail>##

Warning

If the source of the item to be sent is a Group, Branch, or Plex, the inserted status format for those commands will not be correct. It will appear as "BRANCH BEGINNING: text" ("GROUP..." etc.) and must be edited to "BRANCH AT: ADDRESS" before the Process (command form) command can be used.

% backlinks: <base, insert !sendmail>##

Interrogate OK

The Sendmail subsystem command "Interrogate" automatically begins six different Sendmail commands for you, enabling you to send a simple piece of mail. The responses you enter have the same effect as executing those commands. To skip one of the first three questions and go on to the next, in TNLS type an <OK>; in DNLS a NULL character (<CTRL-N>) must precede the <OK>. 3L

(distribute action to) CONTENT

##<sendmail !distribute>##

(distribute information-only to) CONTENT

##<sendmail !distribute>##

(title) CONTENT

##<sendmail !title>##

(type of source) ITEM

##<sendmail !item>##

(show status) ANSWER

##<sendmail !show>##

(send the mail ?) ANSWER

##<sendmail !send>##

Keywords CONTENT OK

The Sendmail command "Keywords" allows you to specify which words will typify the nature of the document. For CONTENT, type in the key words you want to have identified with your document. ARC does not at present produce key word indices of Journal items except on request. However, words in the title are used in the title word index. See also: catalogs. 3M

Message CONTENT OK
The Sendmail command "Message" allows you to type in for CONTENT one statement (up to 2000 characters). You may use backspace character (BC) and backspace word (BW) to edit the message and <CTRL-R> to see what you've written. See also: item. 3N

Number Assign/Previously ...
The Sendmail command "Number Assign" shows you the number of your current item. The command "Number Previously" allows you to give a number that you have already reserved. In addition, there is a Reserve command in the Sendmail subsystem for getting other types of numbers. See also: numbering. 3O

Assign: Number Assign OK
The Sendmail command "Number Assign" allows you to see the number that has been assigned to your current Journal item. If you do not use this command, Sendmail will not tell you the number of your item until you see it in the author branch of your initial file. Use the Sendmail Reserve command to get a number you can use at a later time. See also: numbering, Number Previously. 301

Previously: Number Previously (Reserved) CONTENT OK
The Sendmail command "Number Previously" allows you to specify a previously reserved number for the number of your current Journal item. Use the Sendmail Reserve command to reserve a number. The Send command will be carried out only if one of the authors or the logged-in user has previously reserved but not used that number. You must use the Sendmail Rfc command to use a previously reserved RFC number. See also: numbering, Number Assign. 302

Obsoletes (item number(s)) CONTENT OK
The Sendmail command "Obsoletes" places a note in the header of the Journal item being sent stating that the item number you specify for CONTENT is obsoleted (replaced) by this item and the fact is noted in the indices. 3P

Offline (item -- located at) CONTENT OK
The Sendmail command "Offline" allows you to mail a citation for and give a number to an offline item. Use the CONTENT selection to describe the offline location. See Xdoc. See also: Number, numbering. 3Q

Plex (at) SOURCE OK
The Sendmail command "Plex" allows you to send a Plex. Specify the Plex to be mailed with a SOURCE specification. VIEWSPECS do not matter. 3R

Private OK
The Sendmail command "Private" marks the Sendmail item to allow only those idents mentioned in the Distribution command to read the item. It automatically places the "AccessList IDENTLIST;" into the header of the journalized document. Only those in the AccessList: IDENTLIST can Forward the document to others not in the original distribution list. When you want to specify Private on a Sendmail form, write out the word in full, capitalized, following ACCESS STATUS:. See also: Public, Base Set Nls Private, Base privacy. 3S

% backlinks: <base, privacy;tr>

Process (sendmail form at) DESTINATION OK
The Sendmail command "Process" causes a previously entered and properly filled Sendmail form or Sendmail Status form to be processed by one command, saving the user from specifying each

command separately. Be sure you specify the type of source before using this command. See Status. See also: Base Process. 3T

Base Insert Sendmail (form) command
##<base, insert !sendmail>##
Sendmail forms for all commands
##<base, insert !sendmail !forms>##
% backlinks: <base, insert !sendmail>--see also under sendmail forms;

Public OK

The Sendmail subsystem command "Public" makes a Sendmail item marked Private in that session return to being Public. To make an item public when using a Sendmail form, leave the ACCESS STATUS: blank. 3U

Rfc (number) CONTENT OK

The Sendmail command "Rfc" allows you to use a reserved Rfc number if you know the ident for which it is reserved. Rfc stands for Request For Comments, which is a series of memoranda between Network Liaison personnel numbered and distributed at the Network Information Center. They are no longer restricted to requests for comments. You must specify a title and author using the Sendmail Title and Author commands. See also: Reserve Rfc. 3V

% This command must precede Reserve so that correct Rfc will be found.

Reserve

The Sendmail command "Reserve" allows you to reserve one of the following types of numbers. See also: numbering. 3W

Journal: Reserve Journal (numbers -- how many?) CONTENT
(insert the number list?) ANSWER
ANSWER = Yes (to follow) DESTINATION OK or
No OK

The Sendmail subsystem command "Reserve Journal" allows you to reserve up to 20 Journal numbers by specifying a count for the CONTENT. They will print out at your terminal. You can have the list inserted at the DESTINATION you specify if you type "y" (Yes) for ANSWER. See also: numbering. 3W1

Xdoc: Reserve Xdoc (numbers -- how many?) CONTENT
(insert the number list?) ANSWER
ANSWER = Yes (to follow) DESTINATION OK or
No OK

The Sendmail command "Reserve Xdoc" allows you to reserve up to 20 XDOC numbers for items that you do not intend to access online by specifying a count for the CONTENT. They will print out at your terminal. You can have the list inserted at the DESTINATION you specify if you type "y" (Yes) for ANSWER. See also: XDOC, numbering. 3W2

Rfc: Reserve Rfc (number) OK
(title) CONTENT
(author) CONTENT
(send to) CONTENT
(online document?) ANSWER
(show status?) ANSWER
(insert the number list?) ANSWER
(to follow) DESTINATION) OK

The Sendmail command "Reserve Rfc" allows you to reserve one RFC number by answering the questions. You can have the number inserted at the DESTINATION you specify if you say Yes to ANSWER.

See also: RFC, numbering. 3W3

Send (the mail) OK
The Sendmail command "Send" sets in motion the Sendmail process after you have specified correctly a source and any other information you wish. (If you put impossible things in some fields, it won't work.) See how. 3X

Show Record/Status ... 3Y

Record: Show Record (for ident) CONTENT OK
The Sendmail command "Show Record" takes an ident, or TYPEIN for a last-name search or content search, for CONTENT and then displays current information in the Identfile about that person or organization. See last-name, content. 3Y1

Status: Show Status OK
The Sendmail command "Show Status" displays to you what it knows about the current journal item to be sent. It shows the fields along with their current values. Empty fields are omitted. You may replace or add to the items in this list with the appropriate command. You may also use the Insert Status (form) command to place the status as a statement in a file. You may then modify or add to fields with NLS editing commands. When you are finished, the whole form (all fields) can be processed at once with the Sendmail Process (sendmail form) command. 3Y2

Statement (at) CONTENT OK
The Sendmail command "Statement" allows you to send an NLS statement. Specify the statement to be mailed as a SOURCE. VIEWSPECS do not matter. 3Z

Subcollections CONTENT OK
The Sendmail command "Subcollections" enables you to change the default subcollection assignment for cataloging of your item. CONTENT wants GROUP-IDENTS. If the item has an RFC number, it will automatically appear in the NIC (Network Information Center) and NWG (Network Working Group) subcollections. See also: subcollection, numbering. 3A@

Title CONTENT OK
The Sendmail command "Title" allows you to enter a title which will be used in Journal citations, hardcopy printouts of the document, catalog listings, and indices including the title-word index. See also: catalogs. 3AA

Universal commands available in all subsystems
##<nls, universal>## 3AB

Unrecorded ANSWER
The Sendmail command "Unrecorded" is not completely implemented. Items marked "Unrecorded" may be left out of Journal catalogs when they are generated. Eventually such an item will not be given a Journal number and not be recorded in the Journal. 3AC

Update (to item number(s)) CONTENT OK
The Sendmail command "Update" allows you to inform the Journal catalog system that this item updates another already in the Journal catalog. Type in the number of the old document for CONTENT. See also: numbering. 3AD

%

```
%backlinks:  <base, output !journal>, <base, output !journal !append>,
<base, output !journal !copies>, <base, output !journal !file>, <base,
print !journal>, <base, insert !sendmail>.  Backlinks to the following
from <base, insert !sendmail>:  title, comment, authors, number,
distribute, subcollections, keywords, expedite, unrecorded, offline,
rfc, obsoletes, private, update, insert link, forward, message,
statement, branch, plex, group, file, send, process.
% source-code for Sendmail .YBS=0; .PBS; .IgRest; (:b> %                  5
   % grammar source code (arcsubsys, cml10,) (relnine, sendmail.cgr,) %
                                                                          5A
      FILE csendmail
      % COMPILING INSTRUCTIONS %
         INCLUDE <nine, nls-grammar, flags !subsystems>                  5A2A
      % DECLARATIONS %
         INCLUDE <nine, nls-grammar, declarations !universal>            5A3A
         DECLARE FUNCTION % subsystem xroutines %                        5A3B
             jsetfield, xjassign, xjforward, xjzapworfil, xjinsstatus,
            xjstatus, xjdoit, xjprocess, xjreserve, xjinsrecord,
            xjrfcshow, xjrfcreserve, xjistatus, xjinsadr, xwarning,
            xjloaworfil, initjwork, xjcopycontent;
         DECLARE COMMAND WORD
            "ACTION" = 102,
            "JOURNAL" = 103,
            "AUTHORS" = 104,
            "COMMENT" = 105,
            "EXPEDITE" = 106,
            "HARDCOPY" = 107,
            "INFORMATION" = 108,
            "INSERT" = 109,
            "KEYWORDS" = 110,
            "OBSOLETES" = 111,
            "PRIVATE" = 112,
            "PUBLIC" = 113,
            "RFC" = 114,
            "SUBCOLLECTIONS" = 115,
            "TITLE" = 116,
            "UNRECORDED" = 117,
            "UPDATE" = 118,
            "TO" = 119,
            "COPIES" = 120,
            "UNDELETE" = 122;
      % COMMON RULES %                                                   5A4
         INCLUDE <nine, nls-grammar, rules !universal>
      % COMMANDS % SUBSYSTEM sendmail KEYWORD "SENDMAIL"                 5A5
         INITIALIZATION %get work file%                                  5A5A
            init =                                                      5A5A1
               % should be "xwarning" call here if front end can know
               whether or not it is an experimental system %
               xjloaworfil()  ;
         strctr COMMAND =                                                5A5B
            param _ ("STATEMENT"!L2!/"GROUP"!L2!/"PLEX"!L2!/"BRANCH")
               <"at"> param2 _ SSEL(param) CONFIRM jsetfield(param,
               param2) ;
         author COMMAND =  "AUTHORS"                                     5A5C
            param _ LSEL(#"IDENTLIST")
            CONFIRM jsetfield [shwhelprule] (#"AUTHORS", param) ;
```

```
commnt COMMAND =  "COMMENT"                                          5A5D
   param _ LSEL(#"TEXT")
   CONFIRM jsetfield(#"COMMENT", param) ;                            5A5D2
copycontent COMMAND =  "COPY"!L2!
   <"content of mail to follow ">
   dest _ DSEL(#"STATEMENT")                                         5A5E2
   level _ levadj()                                                  5A5E3
   xjcopycontent(dest, level);
distribute COMMAND =  "DISTRIBUTE"  <"for">                          5A5F
   param _ ("ACTION" <"to"> / "INFORMATION" <"only to">)             5A5F1
   param2 _ LSEL(#"IDENTLIST") CONFIRM                               5A5F2
   jsetfield [shwhelprule] (param, param2) ;                         5A5F3
expedite COMMAND =  "EXPEDITE"!L2!                                   5A5G
   CONFIRM                                                           5A5G1
   jsetfield(#"EXPEDITE", ANSWER) ;
file COMMAND =  param _ "FILE" param2 _ DSEL(#"CHARACTER")
                                                                     5A5H
   CONFIRM jsetfield(param, param2) ;
forward COMMAND =  "FORWARD"!L2! <"item number">                     5A5I
   param _ LSEL(#"NUMBER")                                           5A5I1
   param2 _ #"COPIES"                                                5A5I2
   % this could be an option:
      <"for"> param2 _ ("ACTION" / "INFORMATION" <"only">) %
   <"to"> param3 _ LSEL(#"IDENTLIST")
   CONFIRM xjforward [shwhelprule] (param, param2, param3);
                                                                     5A5I5
initialize COMMAND =  "INITIALIZE"!L2!                               5A5J
   <"specifications">
   CONFIRM xjzapworfil() ;
insert COMMAND =  "INSERT"!L2!                                       5A5K
   ( "MAILING" <"address for ident(s)">
      level _ NULL                                                   5A5K1A
      param _ LSEL(#"IDENTLIST")                                     5A5K1B
      CLEAR <"to follow statement at">                               5A5K1C
      dest _ DSEL(#"STATEMENT")                                      5A5K1D
      level _ levadj()                                               5A5K1E
      CONFIRM                                                        5A5K1F
       xjinsadr [shwhelprule] (param, dest, level)
   % (not implemented) / "LINK" <"to follow">
      param _ DSEL(#"VISIBLE")
      CONFIRM jsetfield(#"INSERT", param)
      %
   / "STATUS" <"form to follow">
      param _ DSEL(#"STATEMENT")
      param2 _ levadj()
      CONFIRM xjinsstatus(param, param2)
   / "RECORD" <"for ident">
      param _ LSEL(#"IDENT")
      <"as a statement to follow">
      param2 _ DSEL(#"STATEMENT")
      param3 _ levadj()
      CONFIRM xjinsrecord(param, param2, param3)
   );
interrogate COMMAND =  "INTERROGATE"                                 5A5L
   CONFIRM
   CLEAR <"distribute for action to:">
```

```
        param _ LSEL(#"IDENTLIST")
        jsetfield [shwhelprule] (#"ACTION", param)
    CLEAR <"distribute for information-only to:">
        param _ LSEL(#"IDENTLIST")
        jsetfield [shwhelprule] (#"INFORMATION", param)
    CLEAR <"title:"> param _ LSEL(#"TEXT")
        jsetfield(#"TITLE", param)
    CLEAR <"type of source:">  (
        "MESSAGE" param _ #"STATEMENT"
            param2 _ LSEL(#"TEXT") /                          5A5L5A1
        param _ "FILE"
            param2 _ DSEL(#"CHARACTER") /
        param _ structure
            <"at"> param2 _ SSEL(param)
        %
        / param _ "OFFLINE" <"item">
            <"located at"> param2 _ LSEL(#"TEXT")
        %
        )
        jsetfield(param, param2)
    CLEAR <"show status?">
        ( param _ ANSWER
            ( IF NOT param
            / IF param xjstatus() showresult
            )
        )
    CLEAR <"send the mail now?"> param _ ANSWER
        ( IF param xjdoit() )
    ;
keywords COMMAND =  "KEYWORDS"                                5A5M
    param _ LSEL(#"TEXT")
    CONFIRM jsetfield(#"KEYWORDS", param) ;
message COMMAND =  "MESSAGE"                                  5A5N
    param _ LSEL(#"TEXT") CONFIRM
    jsetfield(#"STATEMENT", param) ;
number COMMAND =  "NUMBER"                                    5A5O
    ("ASSIGN" CONFIRM
        param _ xjassign(#"JOURNAL")                          5A501A
        showresult                                            5A501B
    / "PREVIOUSLY" <"reserved"> param _ LSEL(#"NUMBER")
        CONFIRM
        jsetfield(#"NUMBER", param)
    ) ;
obsoletes COMMAND =  "OBSOLETES"                              5A5P
    <"item number(s)">  param _ LSEL(#"TEXT")
    CONFIRM jsetfield(#"OBSOLETES", param) ;
offline COMMAND =  "OFFLINE"!L2!                              5A5Q
    <"item -- located at"> param _ LSEL(#"TEXT") CONFIRM
    jsetfield(#"HARDCOPY", param) ;
process COMMAND =  "PROCESS"                                  5A5R
    <"sendmail form at">
    param _ DSEL(#"STATEMENT")
    CONFIRM
    xjprocess(param) ;
private COMMAND =  "PRIVATE"!L2!                              5A5S
    CONFIRM jsetfield(#"PRIVATE", TRUE) ;
```

```
public COMMAND =  "PUBLIC"!L2!                                          5A5T
   CONFIRM jsetfield(#"PUBLIC", TRUE) ;
reserve COMMAND =   "RESERVE" (                                         5A5U
   param _
      ("JOURNAL" / "XDOC")
      <"numbers -- how many?"> param2 _ LSEL(#"NUMBER")
      CLEAR <"insert the number list?"> param4 _ ANSWER
         (IF param4 <"to follow"> param3 _ DSEL(#"VISIBLE") /
         IF NOT param4 param3 _ NULL)
      CONFIRM xjreserve(param, param2, param3)
         showresult
   / param _ "RFC" <"number"> CONFIRM
      CLEAR <"title"> param _ LSEL(#"TEXT")
      CLEAR <"author"> param2 _ LSEL(#"TEXT")
      CLEAR <"distribute to"> param3 _ LSEL(#"IDENTLIST")
      CLEAR <"online document?"> source _ ANSWER
      CLEAR <"show status?"> param4 _ ANSWER
         (IF param4
            xjrfcshow(param, param2, param3, source)
            showresult
         / IF NOT param4 )
      CLEAR <"insert the number list?"> param4 _ ANSWER
         (IF param4 CLEAR <"to follow"> param4 _
         DSEL(#"VISIBLE") /
         IF NOT param4 param4 _ NULL )
      CONFIRM
      xjrfcreserve [shwhelprule] (param, param2, param3,
      source, param4)
   ) ;
rfc COMMAND =  "RFC"!L2!                                                5A5V
   <"number">  param _ LSEL(#"NUMBER")
   CONFIRM jsetfield(#"RFC", param) ;
send COMMAND =  "SEND" <"the mail"> CONFIRM <"really?">
CONFIRM xjdoit() ;                                                      5A5W
show COMMAND =  "SHOW"!L2!                                              5A5X
   ( "STATUS" CONFIRM
      xjstatus ()
   / "RECORD"  <"for ident">
      param _ LSEL(#"IDENTLIST")                                        5A5X2A
      param2 _ #"UNDELETE"
      CONFIRM
      xjistatus [shwhelprule] (param, param2) )
   showresult ;                                                         5A5X3
subcollections COMMAND =  "SUBCOLLECTIONS"!L2!                          5A5Y
   param _ LSEL(#"IDENTLIST")
   CONFIRM jsetfield [shwhelprule] (#"SUBCOLLECTIONS", param)
   ;
title COMMAND =  "TITLE"                                                5A5Z
   param _ LSEL(#"TEXT")
   CONFIRM jsetfield(#"TITLE", param) ;
update COMMAND =  "UPDATE"!L2!                                          5A5A@
   <"to item number(s)"> param _ LSEL(#"TEXT")
   CONFIRM jsetfield(#"UPDATE", param) ;
unrecorded COMMAND =  "UNRECORDED"                                      5A5AA
   CONFIRM
   jsetfield(#"UNRECORDED", ANSWER) ;
```

```
    INCLUDE <nine, nls-grammar, commands !universal>          5A5A8
    END.
FINISH
```

```
% BE source code (arcsubsys, 1109,) (relnine, sendmail.subsys,)
(netsys, include,) %                                                    5B
   FILE pssendmail
   NOMESS
   ALLOW!
   % DECLARATIONS %                                                     5B4
      (nullsource) STRING = "[No content was specified!]";             5B4A
      (sendmail) EXTERNAL _ (                                          5B4B
         $"JSETFIELD", $jsetfield,
         $"XJASSIGN", $xjassign,
         $"XJCOPYCONTENT", $xjcopycontent,
         $"XJFORWARD", $xjforward,
         $"XJZAPWORFIL", $xjzapworfil,
         $"XJINSADR", $xjinsadr,
         $"XJINSRECORD", $xjinsrecord,
         $"XJINSSTATUS", $xjinsstatus,
         $"XJSTATUS", $xjstatus,
         $"XJDOIT", $xjdoit,
         $"XJRESERVE", $xjreserve,
         $"XJPROCESS", $xjprocess,
         $"XJRESERVE", $xjreserve,
         $"XJRFCSHOW", $xjrfcshow,
         $"XJRFCRESERVE", $xjrfcreserve,
         $"XJISTATUS", $xjistatus,
         $"XJLOAWORFIL", $xjloaworfil,
         $"INITJWORK", $initjwork,
         0,0);
      % for sendmail status and forms%
         (sjnumber) EXTERNAL STRING = "NUMBER:";
         (sjtitle) EXTERNAL STRING = "TITLE:";
         (sjcomment) EXTERNAL STRING = "COMMENT:";
         (sjauthor) EXTERNAL STRING = "AUTHOR(S):";
         (sjaction) EXTERNAL STRING = "DISTRIBUTE FOR ACTION TO:";
         (sjinfo) EXTERNAL STRING = "DISTRIBUTE FOR INFO-ONLY TO:";
         (sjsubcol) EXTERNAL STRING = "SUBCOLLECTION(S):";
         (sjkeywords) EXTERNAL STRING = "KEYWORD(S):";
         (sjhandling) EXTERNAL STRING = "HANDLING INSTRUCTION:";
         (sjrecording) EXTERNAL STRING = "RECORDING INSTRUCTION:";
         (sjhardcopy) EXTERNAL STRING = "OFFLINE ITEM -- LOCATED
         AT:";
         (sjrfc) EXTERNAL STRING = "RFC NUMBER:";
         (sjobsoletes) EXTERNAL STRING = "OBSOLETES ITEM
         NUMBER(S):";
         (sjaccess) EXTERNAL STRING = "ACCESS STATUS:";
         (sjupdates) EXTERNAL STRING = "UPDATE TO ITEM NUMBER(S):";
         (sjlink) EXTERNAL STRING = "INSERT LINK TO FOLLOW:";
         (sjforward) EXTERNAL STRING = "FORWARD ITEM NUMBER:";
         (sjmessage) EXTERNAL STRING = "MESSAGE:";
         (sjbranch) EXTERNAL STRING = "BRANCH AT:";
         (sjplex) EXTERNAL STRING = "PLEX AT:";
         (sjgroup) EXTERNAL STRING = "GROUP AT:";
         (sjfile) EXTERNAL STRING = "FILE:";
         (sjsendit) EXTERNAL STRING = "SEND THE MAIL.";
         (sjbacksendit) EXTERNAL STRING = ".LIAM EHT DNES";
      %for Journal invocation from another (e.g. FTP) process%
```

```
        (ok)=1;          % successful return %                    5B4D1
        (cat)=0;         % catastrophic error return %            5B4D2
        (submission)=1; % journal submission %                    5B4D3
        (leaving)=-1;    % file leaving system %                  5B4D4
        (entering)=0;    % file entering system %                 5B4D5
    % XROUTINES %                                                  5B5
      (jsetfield)    %set the specified field to the specified
      value%
      PROCEDURE (sndmlcmd REF, valueptr REF);                      5B5A
        LOCAL value REF, proced REF;
        LOCAL TEXT POINTER t1, t2;
        LOCAL STRING
          locstr[1000];  %for passing strings to core routines%
        IF &valueptr THEN &value _ ELEM#valueptr#[tppair];
        t1 _ value[d2sel]; t1[1] _ value[d2sel+1];
        CASE ELEM#sndmlcmd#[cwtype] OF
          =102, =119 %- action or to -%:  &proced _ $setjaction;
          =104 %- authors -%:  &proced _ $setjauthor;
          =26 %- branch -%, =28 %- plex -%, =27 %- group -%:
            BEGIN
            setjsource(groupv, &value, $t1);
            RETURN;
            END;
          =105 %- comment -%:  &proced _ $setjcomment;
          =106 %- expedite -%:
            BEGIN
            setjexpedite(value);
            RETURN;
            END;
          =51 %- file -%: %number%
            BEGIN
            filnam(value.stfile, $locstr);
            FIND SF(*locstr*) ^value SE(*locstr*) ^t1;
            setjsource(filew, &value, $t1);
            RETURN;
            END;
          =107 %- hardcopy -%:
            BEGIN
            setjsource(hcopyv, &value, $t1);
            RETURN;
            END;
          =108, =120 %- information or copies -%:  &proced _
          $setjinfo;
          =109 %- insert -%: %link%
            BEGIN
            setjlink($t1);
            RETURN;
            END;
          =110 %- keywords -%:  &proced _ $setjkeyword;
          = 8 %- number -%:
            BEGIN
            *locstr* _ value t1;
            FIND SF(*locstr*) ^value $D ^t1;
            &proced _ $setjnumber;
            END;
          =111 %- obsoletes -%:  &proced _ $setjobsoletes;
```

```
        =112 %- private -%:
           BEGIN
           chprvsts (jwp1.stfile, $psprivate);
           RETURN;
           END;
        =113 %- public -%:
           BEGIN
           chprvsts (jwp1.stfile, $pspublic);
           RETURN;
           END;
        =114 %- rfc -%:  &proced _ $setjrfc;
        = 29 %- statement -%:
           BEGIN
           setjsource(stmtv, &value, $t1);
           RETURN;
           END;
        =115 %- subcollections -%:  &proced _ $setjsubcol;
        =116 %- title -%:  &proced _ $setjtitle;
        =12 %- text -%:
           BEGIN
           setjsource(textv, &value, $t1);
           RETURN;
           END;
        =117 %- unrecorded -%:
           BEGIN
           setjunrecorded (TRUE);
           RETURN;
           END;
        =118 %- update -%:  &proced _ $setjupdates;
        ENDCASE err(notyet);
     %call appropriate setj procedure%
        *locstr* _ value t1;
        proced( $locstr );
     RETURN;
     END.

(xjzapworfil)  % initialize work file %
PROCEDURE;                                                    5B5B
     LOCAL STRING locstr[200];
     IF jworkstid THEN resetf(jworkstid.stfile)
     ELSE
        BEGIN
        *locstr* _ "$JWORK-", *initsr*, "$.SYSTEM;";
        jworkstid _ openwk(0, $locstr);
        END;
     FIND SF(jworkstid) ^jwp1;
     initjwork();
     RETURN;
     END.

(xjloaworfil)  % load Journal work file %
PROCEDURE;                                                    5B5C
     LOCAL TEXT POINTER tp1, tp2;
     LOCAL STRING locstr[200];
     IF NOT jworkstid.stfile THEN
        BEGIN % load it %
```

```
      *locstr* _ "<,  *userstr*, ">, "SEND-MAIL-", *initsr*,
      ".NLS", fvrdchar, "1;P707000";
      INVOKE(jwsigtrap, finis); % if open fails, create it %
      jworkstid _ orgstid;
      jworkstid.stfile _ open(0, $locstr);
      DROP (jwsigtrap); % old file loaded %
      FIND SF(jworkstid) ^jwp1;
      [flntadr(jworkstid.stfile)].flnoclos _ TRUE;
      (finis):  %finishup and leave%                          5B5C3I
      END;
   %  commented out for now
   IF FIND SF(jworkstid) "INCOMPLETE" THEN
      BEGIN
      dismes(1, $"Last session not completed.");
      RETURN;
      END;
   %
   initjwork();
   RETURN;
   (jwsigtrap) CATCHPHRASE;                                   5B5C9
      BEGIN
      INVOKE (contcatch);
      CASE SIGNALTYPE OF = aborttype:
         BEGIN %must create it%
         jworkstid _ openwk(0, $locstr);
         TERMINATE;
         END;
      ENDCASE;
      CONTINUE;
      END;
   END.

%xjcloworfil) not called       %% close work file %%        5B5D
   PROCEDURE (resultptr, parsemode);
   REF resultptr;
   CASE parsemode OF
      = parsing:
         BEGIN
         IF jworkstid.stfile THEN
            [flntadr(jworkstid.stfile)].flnoclos _ FALSE;
         jworkstid _ 0;
         END;
      ENDCASE;
   RETURN( &resultptr );
   END.

   %
(xjassign)              % assign a number %
PROCEDURE (type, rtnlist REF);                                5B5E
   LOCAL STRING numstr[200];
   gcatnums ($numstr, $initsr, $"JOURNAL",  1, 1 );
   setjnumber($numstr);  % set the number field in item %
   % return the number to be shown %
      #rtnlist#[1] _ USE rtnstring($numstr);
   RETURN;
   END.
```

```
(xjreserve)              % reserve numbers %
PROCEDURE (type REF, numb REF, insrtlst REF, rtnlist REF); 5B5F
   LOCAL typsad, insertloc REF;
   LOCAL TEXT POINTER z1, z2, z3;
   LOCAL STRING str[200];
   typsad _ CASE ELEM#type#[cwtype] OF
      =103: $"JOURNAL";
      ENDCASE $"XDOC";
   gcatnums
      ($str,
      $initsr,
      typsad,
      IF &numb THEN
         getpint(ELEM#numb#[tppair], ELEM#numb#[tppair]+d2sel)
      ELSE 1,
      1 );
   IF &insrtlst THEN % insert it as a visible%
      BEGIN
      &insertloc _ ELEM#insrtlst#[tppair];
      dpset (dsprfmt, insertloc, endfil, insertloc);
      FIND SF(*str*) ^z1 SE(*str*) < CH > ^z2;
      z3 _ insertloc[d2sel];  z3[1] _ insertloc[d2sel+1];
      cinstex ($z3, $z1, $z2, TRUE);
      END
   ELSE %just show it%
      #rtnlist#[1] _ USE rtnstring($str);
   RETURN;
   END.

(xjrfcreserve) % reserve an RFC and a Catalog numstr %
PROCEDURE (titlestr, authstr, sendto, onlineflag, insrtlptr
REF, rtnlist REF);                                          5B5C
   LOCAL insertloc REF;
   LOCAL TEXT POINTER z1, z2;
   LOCAL STRING
      auth[50], titl[200], send[150], rfcnum[50], numstr[100];
   %build strings for rfcex%
      getpstring(titlestr, $titl);
      getpstring(authstr, $auth);
         IF NOT auth.L THEN *auth* _ *initsr*;
      getpstring(sendto, $send);
   &insertloc _ ELEM#insrtlptr#[tppair];
   rfcex($auth, $titl, $send, onlineflag, $rfcnum, $numstr);
   *numstr* _ "RFC numstr: ", *rfcnum*, ", Catalog numstr: ",
   *numstr*;
   IF insertloc THEN %insertloc it as a visible%
      BEGIN
      FIND SF(*numstr*) ^z1 SE(*numstr*) ^z2;
      cinstex (&insertloc, $z1, $z2, TRUE);
      END
   ELSE %just show him%
      #rtnlist#[1] _ USE rtnstring($numstr);
   RETURN;
   END.
```

```
(xjrfcshow)             % show status of RFC reserve requestr %
PROCEDURE (titlestr, authstr, sendto, onlineflag REF, rtnlist
REF);                                                          5B5H
   LOCAL STRING auth[50], send[150], titl[200], str[200];
   % get strings for status message%
      getpstring(titlestr, $titl);
      getpstring(authstr, $auth);
         IF NOT auth.L THEN *auth* _ *initsr*;
      getpstring(sendto, $send);
   *str* _
      "Title: ", *titl*, "
      Author(s): ", *auth*, "
      Send to: ", *send*;
   IF onlineflag THEN
      *str* _ *str*, "
      Online document"
   ELSE
      *str* _ *str*, "
      Offline document";
   %show him%
      #rtnlist#[1] _ USE rtnstring($str);
   RETURN;
   END.

(xjdoit)                % finish submission or forward request
%
PROCEDURE;                                                     5B5I
   jsubmit();
   RETURN;
   END.

(xjlock)% not used by sendmail %        % lock or unlock the
journal %
PROCEDURE (password, lockflag);                                5B5J
   LOCAL TEXT POINTER z1, z2;
   LOCAL STRING passstr[100];
   % get information out of the parameter records %
      getpstring(password, $passstr);
   % Check the password value. %
      IF *passstr* # *jnlpsw* THEN
         BEGIN
         err(1, $"Illegal password");
         END;
   % Lock or unlock the journal %
      IF lockflag THEN lockjo(0 %jlock flag%)
      ELSE unlkjo(0);
   RETURN;
   END.

(xjprocess)             % process command form %
PROCEDURE (destptr REF);                                       5B5K
   LOCAL destination REF;
   LOCAL TEXT POINTER left, right, z1, z2;
   LOCAL STRING locstr[1500];
   &destination _ ELEM#destptr#[tppair];
   %Number%
```

```
      IF FIND SF(destination) [*sjnumber*] $SP ^left [EOL] <
      CH $SP ^right > THEN
         BEGIN
         *locstr* _ left right;
         IF locstr.L THEN setjnumber ($locstr)
         ELSE %assign number and update form%
            BEGIN
            gcatnums ($locstr, $initsr, $"JOURNAL", 1, 1);
            IF locstr.L THEN
               BEGIN
               setjnumber ($locstr);
               ST left right _ *locstr*;
               END;
            END;
         END;
   %title%
      IF FIND SF(destination) [*sjtitle*] $SP ^left [EOL] < CH
      $SP ^right > THEN
         BEGIN
         *locstr* _ left right;
         IF locstr.L THEN setjtitle ($locstr);
         END;
   %comment%
      IF FIND SF(destination) > [*sjcomment*] $SP ^left [EOL]
      < CH $SP ^right > THEN
         BEGIN
         *locstr* _ left right;
         IF locstr.L THEN setjcomment ($locstr);
         END;
   %author%
      IF FIND SF(destination) > [*sjauthor*] $SP ^left [EOL] <
      CH $SP ^right > THEN
         BEGIN
         *locstr* _ left right; %supposedly an ident string%
         IF locstr.L THEN setjauthor ($locstr);
         END;
   %Action distribution%
      IF FIND SF(destination) > [*sjaction*] $SP ^left [EOL] <
      CH $SP ^right > THEN
         BEGIN
         *locstr* _ left right; %supposedly an ident string%
         IF locstr.L THEN setjaction ($locstr);
         END;
   %Information-only distribution%
      IF FIND SF(destination) > [*sjinfo*] $SP ^left [EOL] <
      CH $SP ^right > THEN
         BEGIN
         *locstr* _ left right; %supposedly an ident string%
         IF locstr.L THEN setjinfo ($locstr);
         END;
   %subcollections%
      IF FIND SF(destination) > [*sjsubcol*] $SP ^left [EOL] <
      CH $SP ^right > THEN
         BEGIN
         *locstr* _ left right; %supposedly an ident string%
         IF locstr.L THEN setjsubcol ($locstr);
```

```
        END;
    %keywords%
       IF FIND SF(destination) > [*sjkeywords*] $SP ^left [EOL]
       < CH $SP ^right > THEN
          BEGIN
          *locstr* _ left right;
          IF locstr.L THEN setjkeyw ($locstr);
          END;
    %handling instructions%
       IF FIND SF(destination) > [*sjhandling*] $SP ^left [EOL]
       < CH $SP ^right > THEN
          BEGIN
          *locstr* _ left right;
          IF locstr.L THEN setjexpedite (*locstr* =
          "Expedite");
          END;
    %recording instructions%
       IF FIND SF(destination) > [*sjrecording*] $SP ^left
       [EOL] < CH $SP ^right > THEN
          BEGIN
          *locstr* _ left right;
          IF locstr.L THEN
             setjexpedite (*locstr* = "Unrecorded");
          END;
    %hardcopy location%
       IF FIND SF(destination) > [*sjhardcopy*] $SP ^left [EOL]
       < CH $SP ^right > THEN
          BEGIN
          *locstr* _ left right;
          IF locstr.L THEN
             BEGIN
             FIND SF(*locstr*) ^left SE(*locstr*) ^right;
             setjsource (hcopyv, $left, $right);
             END;
          END;
    %rfc number%
       IF FIND SF(destination) > [*sjrfc*] $SP ^left [EOL] < CH
       $SP ^right > THEN
          BEGIN
          *locstr* _ left right;
          IF locstr.L THEN setjrfc ($locstr);
          END;
    %obsoletes%
       IF FIND SF(destination) > [*sjobsoletes*] $SP ^left
       [EOL] < CH $SP ^right > THEN
          BEGIN
          *locstr* _ left right;
          IF locstr.L THEN setjobsoletes ($locstr);
          END;
    %access status%
       IF FIND SF(destination) > [*sjaccess*] $SP ^left [EOL] <
       CH $SP ^right > THEN
          BEGIN
          *locstr* _ left right;
          chprvsts (jwp1.stfile, (IF *locstr* = "PRIVATE"
             THEN $psprivate
```

```
            ELSE $pspublic));
         END;
   %updates%
      IF FIND SF(destination) > [*sjupdates*] $SP ^left [EOL]
      < CH $SP ^right > THEN
         BEGIN
         *locstr* _ left right;
         IF locstr.L THEN setjupdates ($locstr);
         END;
   %insert link%
      IF FIND SF(destination) > [*sjlink*] $SP ^left [EOL] <
      CH $SP ^right > THEN
         BEGIN
         *locstr* _ left right;
         IF locstr.L THEN setjlink ($locstr);
         END;
   %message%
      IF FIND SF(destination) > [*sjmessage*] $SP ^left [EOL]
      < CH $SP ^right > THEN
         BEGIN
         IF POS right > left THEN setjsource (textv, $left,
         $right);
         END;
   %branch%
      IF FIND SF(destination) > [*sjbranch*] $SP ^left [EOL] <
      CH $SP ^right > THEN
         BEGIN
         z1 _ right;  z1[1] _ right[1];
         caddexp($left, $right, lda(), $z1);
         IF z1 NOT= endfil THEN setjsource (groupv, $z1, $z1);
         END;
   %plex%
      IF FIND SF(destination) > [*sjplex*] $SP ^left [EOL] <
      CH $SP ^right > THEN
         BEGIN
         z1 _ right;  z1[1] _ right[1];
         caddexp($left, $right, lda(), $z1);
         IF z1 NOT= endfil THEN
            BEGIN
            left _ gethed(z1);
            right _ getail(z1);
            left[1] _ right[1] _ 1;
            setjsource (groupv, $left, $right);
            END;
         END;
   %group%
      IF FIND SF(destination) > [*sjgroup*] $SP ^left [EOL] <
      CH $SP ^right > THEN
         BEGIN
         z1 _ right;  z1[1] _ right[1];
         caddexp($left, $right, lda(), $z1);
         IF z1 NOT= endfil THEN
            IF FIND ^right > [EOL] $SP ^left [EOL] < CH $SP
            ^right > THEN
               BEGIN
               z2 _ right;  z2[1] _ right[1];
```

```
                caddexp($left, $right, lda(), $z2);
                z1 _ grptst(z1, z2 : z2);
                z1[1] _ z2[1] _ 1;
                setjsource (groupv, $z1, $z2);
                END;
          END;
       %file%
          IF FIND SF(destination) > [*sjfile*] $SP ^left [EOL] <
          CH $SP ^right > THEN
             BEGIN
             IF POS right > left THEN setjsource (filev, $left,
             $right);
             END;
       %send it%
          IF FIND SE(destination) < *sjbacksendit* EOL THEN
             jsubmit();
       RETURN;
       END.

    (xjinsstatus)  % Insert Status of submission form %
    PROCEDURE (destptr REF, level);                                  5B5L
       LOCAL destination REF;
       LOCAL TEXT POINTER z1, z2, z3;
       LOCAL STRING string[2000];
       &destination _ ELEM#destptr#[tppair];
       jstatus ($string);
       cnvcrlfteol($string); %convert string CRLF's to EOL's%
       IF FIND SF(*string*) ^z1 ["FILE: "] ^z2 [EOL] < CH > ^z3
       THEN  %make process sendmail form work for files%
          *string* _ z1 z2, "<, z2 z3, ">, z3 SE(*string*);
       IF FIND SF(*string*) ^z1 ["[THROUGH]"] ^z3 < [ "[ ] > ^z2
       THEN  %make process sendmail form work for groups%
          *string* _ z1 z2, z3 SE(*string*);
       FIND SF(*string*) ^z1 SE(*string*) ^z2;
       curmkr _ cinssta(destination, level, $z1, $z2);
       dpset(dspstrc, curmkr, endfil, curmkr);
       curmkr[1] _ 1;
       RETURN;
       END.

    (xjinsadr)                  % insert address routine %
    PROCEDURE (identptr, stdptr REF, lev);                           5B5M
       LOCAL stid REF;
       LOCAL TEXT POINTER tp1, tp2, tp3, tp4;
       LOCAL STRING idlst[2000], identstr[10], ekname[1000],
       addr[1000];
       &stid _ ELEM#stdptr#[tppair];
       dpset(dspallf, stid, endfil, endfil);
       getpstring(identptr, $idlst);
       FIND SF(*idlst*) ^tp2;
       WHILE ( FIND tp2 $NP ^tp1 PT $PT ^tp2 ) DO
          BEGIN
          *identstr* _ tp1 tp2;
          IF NOT ckident($identstr,$lit,0) THEN
             BEGIN
             *ekname* _ "Can't find ", *identstr*;
```

```
            dismes(1, $ekname);
            REPEAT LOOP;
            END;
         getifnf ($lit, $ekname);
         laddress($lit, $addr,0,0,0);
         *lit* _ *ekname*, EOL, *addr*;
         FIND SF(*lit*) ^tp3 SE(*lit*) ^tp4;
         cinssta(stid, lev, $tp3, $tp4);
         lev _ levsuc; %successor from now on%
         END;
      RETURN;
      END.
   (xjinsrecord)  % Insert Status of ident record %
   PROCEDURE (identptr, destination REF, level);                5B5N
      LOCAL rectype;
      LOCAL TEXT POINTER z1, z2;
      LOCAL STRING idstring[500], recstring[2000], string[2000];
      getpstring(identptr, $idstring) ;
      IF NOT ckident($idstring, $recstring, idfno) THEN
         err($"Illegal Ident");
      rectype _ IF jgrptst($recstring, 0) THEN grptyp
         ELSE IF orgtst($recstring, 0) THEN orgtyp ELSE indtyp;
      jidstatus ($idstring, $recstring, rectype, $string);
      cnvcrlfteol($string); %convert string CRLF's to EOL's%
      FIND SF(*string*) ^z1 SE(*string*) ^z2;
      curmkr _ cinssta(ELEM#destination#[tppair]], level, $z1,
      $z2);
      dpset(dspstrc, curmkr, endfil, curmkr);
      curmkr[1] _ 1;
      RETURN;
      END.

   (xjcopycontent)          %Copy content of message%
   PROCEDURE (destptr REF, level);
      LOCAL stid1, stid2, destination REF;                       5B501
      &destination _ ELEM#destptr#[tppair];
      IF (stid1 _ getnxt(jwp1)) # endfil THEN %there's content%
                                                                 5B503
         BEGIN
         stid2 _ getail(stid1);  %a group%                      5B503B
         stid1 _ ccopgro(destination, level, stid1, stid2, 0, 0);
                                                                 5B503C
         dpset(dspstrc, stid1, endfil, dpstp(destination));
         END
      ELSE
         err($"No content specified yet.");
      RETURN;
      END.
   (xjstatus)               % Show Status of submission request %
                                                                 5B5P
      PROCEDURE (rtnlist REF);
      LOCAL STRING string[2000];
      jstatus ($string);
      % make it pretty %
         *string* _ CR, LF, LF, *string*;
      #rtnlist#[1] _ USE rtnstring($string);
```

```
        RETURN;
        END.

    (xjistatus)    % Show Status of ident record %
    PROCEDURE (identptr, xjnum %not used% , rtnlist REF);      5B5Q
        LOCAL rectype;
        LOCAL STRING idstring[500], recstring[2000], string[2000];
        getpstring(identptr, $idstring);
        IF NOT ckident($idstring, $recstring, idfno) THEN
           err($"Illegal Ident");
        rectype _ IF jgrptst($recstring, 0) THEN grptyp
           ELSE IF orgtst($recstring, 0) THEN orgtyp ELSE indtyp;
        jidstatus ( $idstring, $recstring, rectype, $string);
        #rtnlist#[1] _ USE rtnstring($string);
        RETURN;
        END.

    (xjforward)              % Forward J doc to new ids %
    PROCEDURE (jnumber, actionflag REF, ids);                  5B5R
        LOCAL TEXT POINTER z2;
        LOCAL STRING numberstr[20], idlst[1000];
        getpstring(jnumber,$numberstr,);
        getpstring(ids,$idlst,);
        secdist($numberstr, $idlst, (IF ELEM#actionflag#[cwtype] =
        ( 102 %+ action +% OR 119 ) % to: % THEN TRUE ELSE FALSE));
        RETURN;
        END.

% CORE ROUTINES %                                                5B6
    %....Initialize [SENDMAIL-IDENT].NLS (JWORK) file....%
        (initjwork) %Initialize  JWORK file%
        PROCEDURE;                                             5B6A1
            % DOCUMENTATION for initjwork:
              This procedure initializes the JWORK file, which is
              the work file for sending or forwarding Journal
              items.
                 The JWORK file is named a concatenation of
                 "[send-mail].IDENT , where IDENT is the ident of
                 the person using NLS.
                 The JWORK file is created in the logged in
                 directory by xjloaworfil.
              This routine initializes the origin statement as the
              Journal header.
                 Replace existing origin statement with:
                    Journal Number
                       (a dummy meaning deferred number assingment)
                    Author
                       (ident of user)
                    Message flag
                       (this will be changed when user specifies
                       source of item being sent or forwarded)
                    Clerk
                       (ident of user)
                    Journal Directives
                    2 EOL's (for delimiting comments later)
              Finally, it inserts a dummy message as statement 1.
```

```
      %
    LOCAL stid;
    %-------------------%
    %check jworkstid and setup jwp1%
      jworkstid.stpsid _ origin;  %just for safety's sake%
      IF NOT goodrng(jworkstid) THEN
        err($"Journal Work File Cannot be Initialized");
      FIND SF(jworkstid) ^jwp1;
    %make sure name delimiters are right%
      csetnsta(jworkstid, "(, ") );
    %set up Journal Header%
      ST jwp1_ ";;;; F (J) ; Author(s): /", *initsr*, ";,
        " Clerk: ", *initsr*, ";,
        %Journal Formatting Directives followed by two
        EOL's%
          "  .IGD=0; .SNF=HJRM; .RM=HJRM-7;  .PN=-1;
          .YBS=1; .PES;", EOL, EOL;
    %insert dummy message%
      IF (stid _ getsub(jworkstid)) NOT= jworkstid THEN
        BEGIN %there is substructure, delete it%
        cdelgro(stid, getail(stid), FALSE, 0);
        END;
      cisstr(jworkstid, $nullsource, levdown);
    RETURN END.

%....Set contents of Journal header fields....%
  % DOCUMENTATION FOR setj routines:
    These routines are used to set values of fields in the
    Journal header statement in JWORK file.
    They assume JWP1 is set up to point to the first
    character of the origin statement of the JWORK file and
    are, in general, called with a single parameter, the
    address of a STRING which contains the value to which
    the field is to be set.
    They subsequently search the header statement for the
    field [or the proper location of the field if the field
    is optional and non-present], and replace the existing
    field with the string [or create the field if
    necessary].
    Where there are a number of optional fields immediately
    adjacent to each other, the search algorithm uses a list
    of alternatives, searching for the first preceding field
    first, and if that fails, the second preceding field,
    and if that fails.....
    Setj routines which do unusual things will be separately
    documented.
    %
  (setjauthor) %set the author field of JWORK file%
  PROCEDURE (authorstr);                                  5B6B2
    LOCAL TEXT POINTER z1, z2;
    REF authorstr;  %address of string containing ident(s)
    of new author(s)%
    %--------------------%
    astruc(&authorstr); %Set it upper case%
    FIND jwp1 > ["Author(s):"] ["/"] ^z1 [";"] ^z2 _z2;
    ST z1 z2 _ *authorstr*;
```

```
   RETURN;
   END.

(setjaction) %set Action Distribution field in JWORK
header%
PROCEDURE (recipientstr);                                    5B6B3
   REF recipientstr; %address o string containling list of
   "action" recipients%
   LOCAL TEXT POINTER z1,z2;
   %--------------------%
   FIND jwp1 >
      (["Action Distribution: "] < ["A] SP ^z1 > [";] ^z2
      / ["Author(s): "] [";] ^z1 ^z2);
   ST z1 z2 _  " Action Distribution: /", *recipientstr*,
   ";;
   RETURN END.

(setjaxcess) %set the access list field in JWORK header%
PROCEDURE (accessstr);                                       5B6B4
   %----------%
   LOCAL TEXT POINTER z1, z2;
   REF accessstr;
   %----------%
   FIND jwp1 > (["AccessList: "] < ["A] CH ^z1 > [";] ^z2 /
      (["Updates Document(s): "] /
      ["Obsoletes Document(s): "] /
      ["RFC# "] /
      ["Sub-Collections: "] /
      ["Keywords: "] /
      ["Distribution: "] /
      ["Author(s): "]) [";] ^z1 ^z2);
   ST z1 z2 _ " AccessList: ", *accessstr*, ";;
   RETURN;
   END.

(setjinfo) %set Information Distribution field in JWORK
header%
PROCEDURE (recipientstr);                                    5B6B5
   REF recipientstr; %address o string containling list of
   "information only" recipients%
   LOCAL TEXT POINTER z1,z2;
   %--------------------%
   FIND jwp1 >
      (["Information-only Distribution: "] < ["I] SP ^z1 >
      [";] ^z2
      / ["Author(s): "] [";] ^z1 ^z2);
   ST z1 z2 _  " Information-only Distribution: /",
   *recipientstr*, ";;
   RETURN END.

(setjcomment) %set Comment field in JWORK header%
PROCEDURE (commentstr);                                      5B6B6
   REF commentstr; %address of string containing desire
   comment%
   LOCAL TEXT POINTER z1,z2;
   %---------------%
```

```
      %remove " from the string%
         FIND SF(*commentstr*);
         LOOP CASE READC OF
            = '"': %replace it with a single quote%
               BEGIN
               FIND ^z2 < CH > ^z1;
               ST z1 z2 _ '';
               FIND z2 >;
               END;
            = ENDCHR: EXIT LOOP;
            ENDCASE ;
      FIND jwp1 > [1"Clerk: "] ["

      "] ^z1;
      ST jwp1 _ SF(jwp1) z1, ".PEL;  .PN=PN-1; .GCR;",
      *commentstr*;
      RETURN END.

   (setjexpedite) %set or clear expedite attribute in JWORK
   header%
   PROCEDURE (onoff);                                          5B6B7
      %IF onoff = TRUE, set expedite attribute, otherwise
      clear it%
      LOCAL TEXT POINTER z1,z2;
      %---------------%
      FIND jwp1 >
         ([1"(Expedite) "] ^z2 < ['(] SP > ^z1 /
         ([1"Title: "] < ['T] / ["Author(s): "] < ['A]) SP >
         ^z1 ^z2);
      IF onoff THEN ST z1 z2_ " (Expedite) "
      ELSE ST z1 z2 _ NULL;
      RETURN END.

   (setjnumber) %set the contents of the Number field in the
   JWORK header%
   PROCEDURE (numberstr);                                      5B6B8
      REF numberstr;
      LOCAL TEXT POINTER z1,z2;
      %---------------%
      FIND jwp1 > [1";;;;"] ['(] (['J/'j) ^z1 ['')] ^z2 _z2;
      ST z1 z2 _ *numberstr*;
      RETURN;
      END.

   (setjkeyword) %set keywords field in JWORK header%
   PROCEDURE (keywordstr);                                     5B6B9
      REF keywordstr; %address of string containing list of
      keywords%
      LOCAL TEXT POINTER z1,z2;
      %---------------%
      FIND jwp1 >
         ([1"Keywords: "] < ['K] SP ^z1 > [';] ^z2 /
         ([1"Distribution: "]/["Author(s): "]) [';] ^z1 ^z2);
      ST z1 z2 _ " Keywords: ", *keywordstr*, ';;
      RETURN END.
```

```
(setjlink) %set INSERT LINK field in JWORK header%
PROCEDURE (linkstr);                                        5B6B10
   % this command has not been implemented, should allow
   for multiple insertions. %
   REF keywordstr; % address of string containing link %
   LOCAL TEXT POINTER z1,z2;
   %---------------%
   FIND jwp1 >
      (["Keywords: "] < ["K] SP ^z1 > [";] ^z2 /
      (["Distribution: "]/["Author(s): "]) [";] ^z1 ^z2);
   ST z1 z2 _ " Insert Links at: ", *linkstr*, ";;
   RETURN END.

(setjrfc) %set RFC field in JWORK header%
PROCEDURE (rfcstr);                                         5B6B11
   REF rfcstr; %address of string containing RFC number%
   LOCAL TEXT POINTER z1,z2;
   %---------------%
   FIND jwp1 >
      (["RFC# "] < ["R] SP ^z1 > [";] ^z2 /
      (["Sub-Collections: "] / ["Keywords: "] /
      ["Distribution: "] / ["Author(s): "]) [";] ^z1 ^z2);
   ST z1 z2 _ " RFC# ", *rfcstr*, ";;
   RETURN END.

(setjsource) %set source type and content in JWORK header%
PROCEDURE (sourcetype, s1, s2);                             5B6B12
   % souce may be of type statement, group, file, or
   hardcopy %
   LOCAL TEXT POINTER w1, w2, t1, t2;
   LOCAL STRING locstr[200];
   REF s1, s2;
   %-----------------%
   w1[1] _ w2[1] _ 1;
   FIND jwp1 > [";;;; "] ^t1 CH ^t2;
   FIND t1;
   CASE READC OF
      = "H:  % remove hardcopy location field %
         BEGIN
         IF FIND jwp1 > [")] ["Hard Copy--Location: "] <
         ["H] SP ^w1 > [";] ^w2 THEN
            ST w1 w2 _ NULL;
         END;
      = "F:  % remove origin field %
         BEGIN
         IF FIND jwp1 > ["Clerk: "] ["Origin: "] < ["O] SP
         ^w1 > ["####;"] ^w2 THEN
            ST w1 w2 _ NULL;
         END;
      = "S: NULL;  % just replace substructure is  ok? %
      ENDCASE err($"Illegal Journal Workfile format --
      setjsource");
   IF ( w1 _ getsub(jwp1) ) NOT= jwp1 THEN
      BEGIN  % delete old source %
      w2 _ getail(w1);
      cdelgro(w1, w2, FALSE, 0);
```

```
        END;
      CASE sourcetype OF
        =textv:  %text%
          BEGIN
           % insert new source %
             cinssta(jwp1, levdown, &s1, &s2);
          ST t1 t2 _ ^F;
          END;
        =stmtv:  %statement or text%
          BEGIN
           % insert new source %
             cinssta(jwp1, levdown, &s1, &s2);
          ST t1 t2 _ ^F;
          END;
        =groupv:  % group (branch, plex) %
          BEGIN % insert new group as substructure of header
          %
          IF s1.stastr THEN %user typed "group" text%
             cinssta(jwp1, levdown, &s1, &s2)
          ELSE % normal group %
             ccopgro(jwp1, levdown, s1, s2, FALSE, 0);
          ST t1 t2 _ ^F;
          END;
        =filev: % whole file %
          BEGIN
          %load the file%
             caddexp(&s1, &s2, lda(), $w1);
             IF w1 = endfil THEN
                BEGIN
                *locstr* _ "Illegal link: ", s1 s2;
                err($locstr);
                END;
             s1 _ orgstid;  s1.stfile _ w1.stfile;
          %copy origin statement into jwork header%
             IF NOT FIND jwp1 ["Clerk:"] [EOL EOL] ^w1 _w1
             _w1 THEN
                err($"Improper Journal Workfile Format --
                setjsource");
             ST w1 w1 _ " Origin: ", SF(s1) SE(s1), "####;";
          %copy substructure to jwork file%
             IF (s1 := getsub(s1)) # s1 THEN % there is
             substructure in user's file %
                BEGIN
                s2 _ getail(s1);
                REPEAT CASE (groupv);
                END;
             % no substructure in user's file -- boy is he
             stupid! %
                *locstr* _ "[no content was specified!]";
                FIND SF(*locstr*) ^s1 SE(*locstr*) ^s2;
                REPEAT CASE (stmtv);
          END;
        =forwdv:  % forward an item to specified recipients %
          BEGIN
           % insert forward request %
             cinssta(jwp1, levdown, $s1, $s2);
```

```
            ST t1 t2 _ "S;
            END;
         =hcopyv:  % hardcopy item %
            BEGIN
            FIND t2 [")] ^w1;
            ST t1 _ SF(t1) t1, "H, t2 w1, " Hard
            Copy--Location: ", s1 s2, ";, w1 SE(w1);
            END;
         ENDCASE err($"Illegal source type in setjsource");
      RETURN;
      END.
   (setjsubcol) %set Sub-Collections field in JWORK header%
   PROCEDURE (subcollstr);                                    5B6B13
      LOCAL STRING tempsr[150];
      REF subcollstr; %address of string containing list of
      Sub-Collections%
      LOCAL TEXT POINTER z1,z2;
      %---------------%
      *tempsr* _ *subcollstr*;
      astruc($tempsr); %Make sure that it is upper case%
      FIND SF(jwp1)
         ([Sub-Collections: "] < ["S] SP ^z1 > [";] ^z2 /
         ( ["Keywords: "] / ["Distribution: "] / ["Author(s):
         "]) [";] ^z1 ^z2);
      ST z1 z2 _ " Sub-Collections: ", *tempsr*, ";;
      RETURN END.

   (setjobsoletes) %set Obsoletes Document(s) field in JWORK
   header%
   PROCEDURE (obsoletestr);                                   5B6B14
      LOCAL TEXT POINTER z1, z2;
      REF obsoletestr; %address of string containing list of
      Obsoleted Documents%
      %---------------%
      FIND jwp1 >
         (["Obsoletes Document(s): "] < ["O] SP ^z1 > [";]
         ^z2/
         (["RFC# "] / ["Sub-Collections: "] / ["Keywords: "] /
         ["Distribution: "] / ["Author(s): "]) [";] ^z1 ^z2);
      ST z1 z2 _ " Obsoletes Document(s): ", *obsoletestr*, ";
      ;
      RETURN;
      END.

   (setjunrecorded) %set or clear UNRECORDED attribute in
   JWORK heade % PROCEDURE (onoff);                           5B6B15
      %IF onoff = TRUE, set unrecorded attribute, otherwise
      clear it%
      LOCAL TEXT POINTER z1,z2;
      %---------------%
      FIND jwp1 >
         (["(Unrecorded) "] ^z2 < ["(] SP > ^z1 /
         (["Title: "] < ["T] / ["Author(s): "] < ["A]) SP >
         ^z1 ^z2);
      IF onoff THEN ST z1 z2_ " (Unrecorded) "
      ELSE ST z1 z2 _ NULL;
```

```
   RETURN END.

(setjupdates) %set Updates Document(s) field in JWORK
header% PROCEDURE (updatestr);                              5B6B16
   LOCAL TEXT POINTER z1, z2;
   REF updatestr; %address of string containing list of
   Updated Documents%
   %---------------%
   FIND jwp1 >
      ([^Updates Document(s): ^] < [^U] SP ^z1 > [^;] ^z2/
      ([^Obsoletes Document(s): ^] / [^RFC# ^] /
      [^Sub-Collections: ^] / [^Keywords: ^] /
      [^Distribution: ^] / [^Author(s): ^]) [^;] ^z1 ^z2);
   ST z1 z2 _ " Updates Document(s): ", *updatestr*, "; ;
   RETURN;
   END.

(setjtitle)%set Title field in JWORK header%
PROCEDURE(titlestr);                                        5B6B17
   REF titlestr; %address of string containing desired
   title%
   LOCAL TEXT POINTER z1,z2;
   %---------------%
   %remove " from the string%
      FIND SF(*titlestr*);
      LOOP CASE READC OF
         = "": %replace it with a single quote%
            BEGIN
            FIND ^z2 < CH > ^z1;
            ST z1 z2 _ "";
            FIND z2 >;
            END;
         = ENDCHR: EXIT LOOP;
         ENDCASE ;
   FIND jwp1 >
      ([^Title: .H1=^]  < [^T] SP ^z1 > [^;] ^z2/
       [^Author(s): ^] < [^A] SP ^z1 ^z2>);
   ST z1 z2 _ " Title: .H1=", "",  *titlestr*, "", ";;
   RETURN END.

(setcacc) %set access field in catalog entry%
PROCEDURE (entrystid, repstr, ptr1, ptr2);                  5B6B18
   %----------%
   LOCAL repcnt;
   LOCAL TEXT POINTER z1, z2;
   REF repstr, ptr1, ptr2;
   %----------%
   IF NOT &ptr1 THEN getcacc (entrystid, 0, $z1, $z2)
   ELSE FIND ptr1 ^z1 ptr2 ^z2;
   IF *repstr* = *nullfield* THEN %delete the field%
      IF (FIND z1 < [$PT] $NP > ^z1 [^AccessList:^] z2 [^;]
      ^z2) THEN
         ST entrystid _ SF(entrystid) z1, z2 SE(entrystid)
      ELSE NULL
   ELSE
      BEGIN
```

```
            repcnt _ repstr.L;
            IF *repstr* [repstr.L] = "; THEN BUMP DOWN repcnt;
            IF NOT (FIND z1 < [$PT] $NP > ["AccessList:"]) THEN
               ST entrystid _ SF(entrystid) z1, " AccessList: ",
               *repstr* [1 TO repcnt], ";, z2 SE(entrystid)
            ELSE
               ST entrystid _ SF(entrystid) z1, *repstr* [1 TO
               repcnt], z2 SE(entrystid);
            END;
         RETURN;
         END.

      %....Status....%
         (jidstatus) %generate status string for an ident entry%
         PROCEDURE (idstr REF, entrystr, typid, statstr REF);   5B6C1
            %This procedure inserts the status of the entry passed
            into the status string.%
            LOCAL TEXT POINTER ptr1, ptr2, ptr3, ptr4;
            LOCAL f1, f2;
            LOCAL STRING cadstr[300];
            *statstr* _ "Ident: ", *idstr*, CR, LF;
            getinam(entrystr, 0, $ptr1, $ptr2);
            *statstr* _ *statstr*, "Name: ", ptr1 ptr2, CR, LF;
            IF typid = indtyp THEN
               BEGIN
               getiorg(entrystr, 0, $ptr1, $ptr2);
               *statstr* _ *statstr*, "Organization: ", ptr1 ptr2,
               CR, LF;
               IF getisorg(entrystr, 0, $ptr1, $ptr2) THEN
                  *statstr* _ *statstr*, "Secondary Organization(s):
                  ",
                     ptr1 ptr2, CR, LF;
               END
            ELSE
               BEGIN
               IF expdtst(entrystr, 0) THEN
                  *statstr* _ *statstr*, "Expand", CR, LF
               ELSE *statstr* _ *statstr*, "Unexpanded", CR, LF;
               getimem(entrystr, 0, $ptr1, $ptr2);
               *statstr* _ *statstr*, "Membership: ", ptr1 ptr2, CR,
               LF;
               geticord(entrystr, 0, $ptr1, $ptr2);
               *statstr* _ *statstr*, "Coordinator: ", ptr1 ptr2,
               CR, LF;
               IF typid = orgtyp THEN
                  BEGIN
                  getityp(entrystr, 0, $ptr1, $ptr2);
                  *statstr* _ *statstr*, "Organization Type: ", ptr1
                  ptr2, CR, LF;
                  END;
               END;
            IF getigrps (entrystr, 0, $ptr1, $ptr2) THEN
               *statstr* _ *statstr*, "Groups: ", ptr1 ptr2, CR, LF;
            %Put Online and Network addresses in cadstr%
               *cadstr* _ NULL;
               f2 _ getinlhost(entrystr, 0, $ptr3, $ptr4);
```

```
            IF (f1 _ getiuser(entrystr, 0, $ptr1, $ptr2)) OR f2
            THEN
               BEGIN
               *cadstr* _ *cadstr*, "  Online(NLS)", CR, LF;
               IF f1 THEN *cadstr* _ *cadstr*, "    User: ", ptr1
               ptr2, CR, LF;
               IF f2 THEN *cadstr* _ *cadstr*, "    Host: ", ptr3
               ptr4, CR, LF;
               END;
            f2 _ getihost(entrystr, 0, $ptr3, $ptr4);
            IF (f1 _ getinma(entrystr, 0, $ptr1, $ptr2)) OR f2
            THEN
               BEGIN
               *cadstr* _ *cadstr*, "  Network", CR, LF;
               IF f1 THEN *cadstr* _ *cadstr*, "    User: ", ptr1
               ptr2, CR, LF;
               IF f2 THEN *cadstr* _ *cadstr*, "    Host: ", ptr3
               ptr4, CR, LF;
               END;
         getiadd(entrystr, 0, $ptr1, $ptr2);
         IF cadstr.L > 0 OR ptr2[1] > ptr1[1] THEN
            BEGIN
            *statstr* _ *statstr*, "Mail Addresses: ", CR, LF;
            IF ptr2[1] > ptr1[1] THEN
               BEGIN
               *statstr* _ *statstr*, "  Hardcopy Address: ";
               cetcapp($ptr1, $ptr2, &statstr);
               *statstr* _ *statstr*, CR, LF;
               END;
            *statstr* _ *statstr*, *cadstr*;
            END;
         getiverify(entrystr, 0, $ptr1, $ptr2);
         *statstr* _ *statstr*, ptr1 ptr2, CR, LF;
         IF getiphone(entrystr, 0, $ptr1, $ptr2) THEN
            BEGIN
            *statstr* _ *statstr*, "Phone: ", ptr1 ptr2, CR, LF;
            END;
         IF getifunction(entrystr, 0, $ptr1, $ptr2) THEN
            BEGIN
            *statstr* _ *statstr*, "Function: ", ptr1 ptr2, CR,
            LF;
            END;
         IF identwheel AND geticapability(entrystr, 0, $ptr1,
         $ptr2) THEN
            BEGIN
            *statstr* _ *statstr*, "Capabilities: ", ptr1 ptr2,
            CR, LF;
            END;
         IF getisubcol(entrystr, 0, $ptr1, $ptr2) THEN
            BEGIN
            *statstr* _ *statstr*, "Sub-Collection: ", ptr1 ptr2,
            CR, LF;
            END;
         IF getidelivery(entrystr, 0, $ptr1, $ptr2) THEN
            BEGIN
            *statstr* _ *statstr*, "Delivery: ", ptr1 ptr2, CR,
```

```
        LF;
        END;
    IF getimcmnts(entrystr, 0, $ptr1, $ptr2) THEN
        BEGIN
        *statstr* _ *statstr*, "Comments: ", ptr1 ptr2, CR,
        LF;
        END;
    RETURN;
    END.


(jstatus) %generate a submission status string%
PROCEDURE (status);                                          5B6C2
    % jstatus documentation
        It works by calling a set of getj routines, which are
        roughly complementary to the setj routines including
        the fact that they expect jwp1 to be set to the first
        character of the JWORK origin statement.
        Each getj routine either null's the passed string if
        the field does not exist or fills it with the value
        of the field.
        The results are simply appended and separated by
        EOL's to form a longer string which is printed out to
        the user.
        %
    LOCAL stid;
    LOCAL TEXT POINTER z1, z2;
    LOCAL STRING str[1500];
    REF status;  %address of string into which the status
    message is to be stored%
    %--------------------%
    *status*
        *[addcrlf ( getjnumber ( $str ), $sjnumber )]*,
        *[addcrlf ( getjaxcess ( $str ), $sjaccess )]*,
        *[addcrlf ( getjtitle ( $str ), $sjtitle )]*,
        *[addcrlf ( getjcomment ( $str ), $sjcomment )]*,
        *[addcrlf ( getjauthor ( $str ), $sjauthor )]*,
        *[addcrlf ( getjaction ( $str ), $sjaction )]*,
        *[addcrlf ( getjinfo ( $str ), $sjinfo )]*,
        *[addcrlf ( getjsubcol ( $str,0 ), $sjsubcol  )]*;
    *status* _ *status*, %this is broken to avoid an L10
    STACK OVERFLOW%
        *[addcrlf ( getjkeyw ( $str ), $sjkeywords )]*,
        *[addcrlf ( getjexpedite ($str), $sjhandling)]*,
        *[addcrlf ( getjunrecorded ($str), $sjrecording)]*,
        *[addcrlf ( getjrfc ( $str ), $sjrfc )]*,
        *[addcrlf ( getjobsoletes ( $str ), $sjobsoletes )]*,
        *[addcrlf ( getjupdates ( $str ), $sjupdates )]*,
        *[addcrlf ( getjlink ( $str ), $sjlink )]*;
    getjtype( $str );
    CASE *str*[1] OF  %type of source%
        = 'M, = 'F:
            BEGIN  %determine what status to give%
            % Is it a file?%
            IF FIND SF(jwp1) [^"Clerk: "] [^"Origin: "] [^'<] ^z1
            [^'>] < CH > ^z2 THEN
```

```
                *status* _ *status*, CR, LF, *sjfile*, z1 z2,
                CR, LF
             ELSE  %see if it is a structure%
                IF (stid _ getnxt(jwp1)) NOT= endfil THEN %
                There is substructure%
                   IF getnxt(stid) = endfil THEN
                      IF NOT FIND SF(stid) *nullsource* ENDCHR
                      THEN
                         *status* _ *status*, CR, LF,
                         *sjmessage*, SF(stid) SE(stid), CR, LF
                      ELSE NULL  %user has not supplied source
                      yet%
                   ELSE %show some text from first and last
                   branches%
                      BEGIN
                      FIND SF(stid) ^z1 $15CH ^z2;
                      IF getftl(stid) THEN % a branch%
                         *status* _ *status*, CR, LF, "BRANCH
                         AT: ", '", z1 z2, '", CR, LF
                      ELSE
                         BEGIN
                         *status* _ *status*, CR, LF, "GROUP
                         AT: ", '", z1 z2, '", CR, LF;
                         stid _ getail(stid);
                         FIND SF(stid) ^z1 $15CH ^z2;
                         *status* _ *status*, "  [THROUGH]  ",
                         '", z1 z2, '", CR, LF;
                         END;
                      END;
                END;
          = 'H:  %hardcopy%
             *status* _ *status*,
                *[addcrlf ( getjhcloc ( $str ), $sjhardcopy
                )]*;
          = 'S: %secondary distribution%
             IF (stid _ getnxt(jwp1)) NOT= endfil THEN % There
             is substructure%
                *status* _ *status*, CR, LF, *sjforward*,
                SF(stid) SE(stid), CR, LF;
          ENDCASE;
       RETURN (&status);
       END.

    (addcrlf) %support routine for jstatus%
    PROCEDURE (string, heading);                              5B6C3
       %*addcrlf*
          This routine is used by jstatus for inserting CRLF's
          between fields.
          Basically, if it is handed a NULL string, then it
          does nothing, but if handed a non-empty string, it
          appends an CRLF.
          %
       REF string, heading;
       %----------------%
       IF string.L > 0 THEN *string* _ *heading*, SP, *string*,
       CR, LF;
```

```
        RETURN(&string);
        END.

    %....Get contents of Journal header fields....%
        %
        This is a set of routines which return values of fields in
        the Journal header .
        Generally, they assume jwp1 is a pointer to the first
        character of the statement containing the header, and
        search that statement for the presence of the field.
        If the field is not found, the string address is set to
        NULL.
        Otherwise, the contents of the field is copied into the
        passed string, and its address is returned.
        In cases where the getj routines do unusual things, they
        are documented separately
        %
        (getjaction) %get the contents of the Action Distribution
        field from the JWORK header%
        PROCEDURE (string);                                    5B6D8
           REF string;  LOCAL TEXT POINTER z1, z2;
           IF FIND jwp1 > [^"Action Distribution: "] [^/] ^z1 [^;]
           ^z2 _z2 THEN
              *string* _ z1 z2
           ELSE *string* _ NULL;
           RETURN(&string);
           END.

        (getcacc) %fetch access field from catalog entry%
        PROCEDURE (entrystid, fldstr, ptr1, ptr2);             5B6D9
           %----------%
           LOCAL retval;
           LOCAL TEXT POINTER z1, z2, z3;
           %----------%
           retval _ FALSE;
           makeptr (entrystid, $z3);
           IF NOT fndungtxt ($"AccessList:", $z3, $z1, $z2)
           OR NOT (FIND z2 > $NP ^z1 [^;] < CH $NP ^z2) THEN
              getcend (entrystid, $z1, $z2)
           ELSE retval _ TRUE;
           stptset (fldstr, ptr1, ptr2, $z1, $z2);
           RETURN (retval);
           END.

        (getjauthor) %get the contents of the Author field from the
        JWORK header%
        PROCEDURE (string);                                   5B6D10
           REF string;  LOCAL TEXT POINTER z1, z2;
           FIND jwp1 > [^"Author(s):"] [^/] ^z1 [^;] ^z2 _z2;
           *string* _ z1 z2;
           RETURN(&string);
           END.

        (getjaxcess) %get the contents of the Access List field
        from the JWORK header%
        PROCEDURE (string);                                   5B6D11
```

```
   REF string;  LOCAL TEXT POINTER z1, z2;
   IF rdprvsts (jwp1.stfile) = $psprivate THEN
      *string* _ "PRIVATE"
   ELSE *string* _ NULL;
   RETURN(&string);
   END.


(getjclerk) %get the contents of the Clerk field from the
JWORK header%
PROCEDURE (string);                                          5B6D12
   REF string;  LOCAL TEXT POINTER z1, z2;
   FIND jwp1 > ["Clerk: "] ^z1 [".] ^z2 _z2;
   *string* _ z1 z2;
   RETURN(&string);
   END.

(getjcomment) %get the contents of the Comment field from
the JWORK header%
PROCEDURE (string);                                          5B6D13
   REF string;  LOCAL TEXT POINTER z1, z2;
   *string* _ NULL;
   IF ( FIND jwp1 > ["

   "] [".GCR;"]  ^z1 ) AND ( POS z1 # SE(z1) ) THEN
      *string* _ z1 SE(z1);
   RETURN(&string);
   END.

(getjexpedite) %get the Expedite attribute from the JWORK
header%
PROCEDURE (string);                                          5B6D14
   REF string;  LOCAL TEXT POINTER z1, z2;
   *string* _ NULL;
   IF FIND jwp1 > ["(Expedite)"] THEN *string* _
   "Expedite";
   RETURN(&string);
   END.

(getjunrecorded) %get the Unrecorded attribute from the
JWORK header%
PROCEDURE (string);                                          5B6D15
   REF string;  LOCAL TEXT POINTER z1, z2;
   *string* _ NULL;
   IF FIND jwp1 > ["(Unrecorded)"] THEN *string* _
   "Unrecorded";
   RETURN(&string);
   END.

(getjhcloc) %get the contents of the Hard Copy Location
field from the JWORK header%
PROCEDURE (string);                                          5B6D16
   REF string;  LOCAL TEXT POINTER z1, z2;
   *string* _ NULL;
   IF FIND jwp1 > ["Hard Copy--Location: "] ^z1 [";] ^z2
   _z2 THEN
```

```
      *string* _ z1 z2;
   RETURN(&string);
   END.

(getjinfo) %get the contents of the Information-only
Distribution field from the JWORK header%
PROCEDURE (string);                                          5B6D17
   REF string;  LOCAL TEXT POINTER z1, z2;
   IF FIND jwp1 > [^"Information-only Distribution: "] [^/]
   ^z1 [^;] ^z2 _z2 THEN
      *string* _ z1 z2
   ELSE *string* _ NULL;
   RETURN(&string);
   END.

(getjlink) %get the contents of the Insert Link field from
the JWORK header%
PROCEDURE (string);                                          5B6D18
   REF string;  LOCAL TEXT POINTER z1, z2;
   IF FIND jwp1 > [^"Insert Link at: "] ^z1 [^;] ^z2 _z2
   THEN
      *string* _ z1 z2
   ELSE *string* _ NULL;
   RETURN(&string);
   END.

(getjkeyw) %get the contents of the Keywords field from the
JWORK header%
PROCEDURE (string);                                          5B6D19
   REF string;  LOCAL TEXT POINTER z1, z2;
   IF FIND jwp1 > [^"Keywords: "] ^z1 [^;] ^z2 _z2 THEN
      *string* _ z1 z2
   ELSE *string* _ NULL;
   RETURN(&string);
   END.


(getjnumber) %get the contents of the Number field from the
JWORK header%
PROCEDURE (string);                                          5B6D20
   REF string;  LOCAL TEXT POINTER z1, z2;
   IF NOT FIND jwp1 > [^";;;;"][^(] ('J/'j) ^z1 $D ^z2 THEN
      err($"Bad Journal Work File");
   *string* _ z1 z2;
   RETURN(&string);
   END.

(getjtype) %get the TYPE of the item from the JWORK header%
PROCEDURE (string);                                          5B6D21
   REF string;  LOCAL TEXT POINTER z1, z2;
   IF NOT FIND jwp1 > [^";;;; "] ^z1 CH ^z2 THEN
      err($"Bad Journal Work File");
   *string* _ z1 z2;
   RETURN(&string);
   END.
```

```
(getjobsoletes) %get the contents of the Obsoletes
Documents field from the JWORK header%
PROCEDURE (string);                                          5B6D22
   REF string;  LOCAL TEXT POINTER z1, z2;
   IF FIND jwp1 > [^"Obsoletes Document(s): "] ^z1 [^;] ^z2
   _z2 THEN
      *string* _ z1 z2
   ELSE *string* _ NULL;
   RETURN(&string);
   END.

(getjrfc) %get the contents of the RFC field from the JWORK
header%
PROCEDURE (string);                                          5B6D23
   REF string;  LOCAL TEXT POINTER z1, z2;
   IF FIND jwp1 > [^"RFC# "]  ^z1 [^;] ^z2 _z2 THEN
      *string* _ z1 z2
   ELSE *string* _ NULL;
   RETURN(&string);
   END.

(getjsubcol) %get the contents of the Sub-Collections field
from the JWORK header%
PROCEDURE (string);                                          5B6D24
   REF string;  LOCAL TEXT POINTER z1, z2;
   IF (FIND jwp1 > [^"Sub-Collections: "]  ^z1 [^;] ^z2 _z2
   ) THEN
      *string* _ z1 z2
   ELSE *string* _ NULL;
   RETURN(&string);
   END.

(galjsubcol)PROCEDURE(string, idfnum);                       5B6D25
   %*galjsubcol-- get all jsubcollection *
      This is the getj routine which returns the
      subcollections which the document belongs to.
      It is a little more complex than most getj routines,
      because the subcollection membership is computed from
      more than one field, and redundant entries must be
      deleted.
      The algorithm is:
         Get value of sub-collection field in header
         If this is an RFC document, add NWG and NIC to
         subcollections
         If there are any groups on the distribution list
         for this document, enter them into te
         subcollection list (this uses the routine
         getgpids).
      As usual, idfile is the file number of the identfile
      or zero.
      %
   LOCAL TEXT POINTER z1, z2, z3;
   LOCAL STRING tempsr[150], temp1sr[30];
   REF string;
   IF (FIND jwp1 > [^"Sub-Collections: "]  ^z1 [^;] ^z2 _z2
   ) THEN
```

```
            *string* _ z1 z2
         ELSE *string* _ NULL;
         IF rfcflg THEN *string* _ "NWG ", *string*;
         IF rfcflg AND NOT FIND BETWEEN z1 z2([."NIC".]) THEN
            *string* _ "NIC ", *string*;
         IF FIND jwp1 > [."Distribution: ".] [./.] ^z1 THEN
            BEGIN
               *tempsr* _ NULL;
               getgpids($z1, $tempsr, idfnum); %Extract all group
               references from ident list%
               %Now fix up Sub-collection Field to reflect
               Distribution Groups%
                  FIND SF(*tempsr*) ^z1;
                  WHILE (FIND z1 $NP ^z2 1$PT ^z1) DO
                     BEGIN
                     *temp1sr* _ +z2 z1;
                     IF NOT FIND SF(*string*) [*temp1sr*.] THEN
                        *string* _ *string*, SP, *temp1sr*;
                     END;
            END;
         RETURN(&string);
         END.

      (getjtitle) %get the contents of the Title field from the
      JWORK header%
      PROCEDURE (string);                                        5B6D26
         REF string;  LOCAL TEXT POINTER z1, z2;
         IF FIND jwp1 > [."Title: ".] [.".] -"" ^z1 _z1 [."."] ^z2 _z2
         THEN
            *string* _ z1 z2
         ELSE *string* _ NULL;
         RETURN(&string);
         END.

      (getjupdates) %get the contents of the Updates Documents
      field from the JWORK header%
      PROCEDURE (string);                                        5B6D27
         REF string;  LOCAL TEXT POINTER z1, z2;
         IF FIND jwp1 > [."Updates Document(s): ".] ^z1 [.;.] ^z2
         _z2 THEN
            *string* _ z1 z2
         ELSE *string* _ NULL;
         RETURN(&string);
         END.

   %....field conversion routines....%
      (convjsubcol) %convert the contents of the Sub-Collections,
      Author, and Distribution field into a real Sub-Collections
      field for the JWORK header%
      PROCEDURE (idfile);                                        5B6E1
         % convjsubcol documentation
            This routine updates the sub-collections field for
            this document.
            The sub-collections membership is computed from the
            current sub-collection, author, and Distribution
            fields, and redundant entries are deleted.  If this
```

```
      is an RFC document, add NWG and NIC to
      subcollections.  If there are any groups on the
      distribution list for this document, enter them into
      the sub-collection list (this uses the routine
      getgpids).
      %
   LOCAL rfcf;
   LOCAL TEXT POINTER z1, z2, a1, a2;
   LOCAL STRING
      string[300], tempsr[300], temp1sr[50], temp2sr[50],
      identry[1000];
   %----------------%
   *string* _ NULL;
   IF (FIND jwp1 > ["Sub-Collections: "]   ^z1 [";] ) THEN
       getgpids ($z1, $string, idfile);
   IF rfcf _ ( FIND jwp1 > ["RFC # "] ) THEN
      IF NOT FIND SF(*string*) ["NWG"] THEN
         *string* _ *string*, " NWG";
   IF rfcf AND NOT FIND BETWEEN z1 z2(["NIC"]) THEN
      IF NOT FIND SF(*string*) ["NIC"] THEN
         *string* _ *string*, " NIC";
   getjauthor($tempsr);
   %Now fix up Sub-collection Field to reflect authors%
      FIND SF(*tempsr*) ^z1;
      WHILE (FIND z1 $NP ^z2 1$PT ^z1) DO
         BEGIN
         *temp1sr* _ +z2 z1; %an author%
         ckident($temp1sr, $identry, idfile); %get his
         ident entry%
         lgetsubcoll($identry, $temp2sr, 0, 0); %get his
         subcollections%
         %process each element of his sub-coll list%
            FIND SF(*temp2sr*) ^a1;
            WHILE (FIND a1 $NP ^a2 1$PT ^a1) DO
               BEGIN
               *temp1sr* _ +a2 a1; %a sub-coll%
               IF NOT FIND SF(*string*) [*temp1sr*] THEN
                  *string* _ *string*, SP, *temp1sr*;
               END;
         END;
   IF FIND jwp1 > ["Distribution: "] ["/] ^z1 THEN  %FIX
   THIS LATER FOR ACTION, INFO DIST LIST%
      BEGIN
         *tempsr* _ NULL;
         getgpids($z1, $tempsr, idfile); %Extract all group
         references from ident list%
         %Now fix up Sub-collection Field to reflect
         Distribution Groups%
            FIND SF(*tempsr*) ^z1;
            WHILE (FIND z1 $NP ^z2 1$PT ^z1) DO
               BEGIN
               *temp1sr* _ +z2 z1;
               IF NOT FIND SF(*string*) [*temp1sr*] THEN
                  *string* _ *string*, SP, *temp1sr*;
               END;
      END;
```

```
    IF string.L > 0 THEN setjsubcol($string);
    RETURN;
    END.

(convjdist) %convert Action Distribution and
Information-only Distribution fields into a single
Distribution field with comments appended to recipient
idents.%
PROCEDURE;                                                    5B6E2
   LOCAL TEXT POINTER z1,z2, z3, z4;
   LOCAL STRING dist[500], idstr[25], comm[300], temp[400];
   IF FIND jwp1 > ["Action Distribution: "] ^z2 < ["A] CH
   ^z1 > z2 ["/] $NP ^z2 [";] ^z4 < CH $NP > ^z3 THEN
   %process action list%
      BEGIN %pointers z1 through z4 should be set up as in
      the following example:
         <z1> Action Distribution: / <z2>CHI DIA DCW<z3>
      ;<z4>  %
      *temp* _ z2 z3;  %membership%
      ST z1 z4 _ NULL;  %delete action list%
      %add ( [ ACTION ] ) to each ident; must agree with
      format checked in (nls, jnlde1, distd1)%
         FIND SF(*temp*) ^z1;
         WHILE (FIND z1 $NP $(",) $NP ^z2 1$(LD / "^ / "& /
         "-) ^z1) DO
            BEGIN
            *idstr* _ +z2 z1;
            IF FIND z1 $NP "( ^z2 ([")] ^z1 ^z3 _z3
            /[ENDCHR] ^z1 ^z3) THEN
               *comm* _ z2 z3
            ELSE *comm* _ NULL;
            *dist* _ *dist*, *idstr*, "(, " [ ACTION ] ",
            *comm*, "), SP;
            END;
      END;
   IF FIND jwp1 > ["Information-only Distribution: "] ^z2 <
   ["I] CH ^z1 > z2 ["/] $NP ^z2 [";] ^z4 < CH $NP > ^z3
   THEN %process information-only list%
      BEGIN %pointers z1 through z4 should be set up as in
      the following example:
         <z1> Information-only Distribution: / <z2>RWW<z3>
      ;<z4>  %
      *temp* _ z2 z3;  %membership%
      ST z1 z4 _ NULL;  %delete info list%
      %add ( [ INFO-ONLY ] ) to each ident%
         FIND SF(*temp*) ^z1;
         WHILE (FIND z1 $NP $(",) $NP ^z2 1$(LD / "^ / "& /
         "-) ^z1) DO
            BEGIN
            *idstr* _ +z2 z1;
            IF FIND z1 $NP "( ^z2 ([")] ^z1 ^z3 _z3
            /[ENDCHR] ^z1 ^z3) THEN
               *comm* _ z2 z3
            ELSE *comm* _ NULL;
            *dist* _ *dist*, *idstr*, "(, " [ INFO-ONLY ]
            ", *comm*, "), SP;
```

```
                END;
            END;
         IF dist.L THEN
            BEGIN
            FIND jwp1 > [" Author(s): "] [";] ^z1 ^z2;
            ST z1 z2 _ " Distribution: /", *dist*, ";;
            END;
         RETURN END.

%Procedures concerned with locking/unlocking Journal%
   (unlkjo)PROC(type);                                          5B6F1
      %Type indates flag to be unlocked... -1 means all%
      IF type NOT IN [-1, 1] THEN err($"Illegal Flag Number");
      IF type # 1 THEN flagut(0, $rstfg);
      IF type # 0 THEN flagut(1,$rstfg);
      RETURN;
      END.
%....Journal submission....%
   (jsubmit) %do the Journal submission or secondary
   distribution%
   PROCEDURE;                                                   5B6G1
      %This procedure submits an item to the Journal by
      creating a properly formatted file in the directory
      <tejournal>.%
      LOCAL  rnumstid, numstid, fl, traping, capsav, connflag,
      stid, orgstr;
      LOCAL TEXT POINTER z1, z2, z3, z4, z5, z6, where;
      LOCAL STRING
         number[20],      %journal number%
         rfcnum[20],      %RFC number%
         itemtype[20],    %item type%
         linkaddr[150],   %link to location for link insert%
         fnamestr[150],   %file name string%
         goodids[500],    %list of good idents%
         badids[500],     %list of bad idents%
         workstr[500],    %work string%
         accesslist [10000], %access list%
         dpassw[15],      %password string%
         dirnam[50],      %directory name%
         dfilstr[150],    %tejournal file name%
         mfname[150];    %message file name%
      REF fl, orgstr;
      IF nojournalflag THEN err($"Journal Not Enabled on this
      System");
      jolock(); %Check to make sure it will work -- does not
      return unless things are cool%
      %set stid's to null%
         rnumstid _ numstid _ orgstid;
      capsav _ 0;
      traping _ connflag _ FALSE;
      &orgstr _ 0;
      INVOKE (initrap) ;
      IF NOT idfno THEN idfno _ loaidfil();
      % Get string to be used for origin statement
      restoration. %
         &orgstr _ getstring(2000, $dspblk);
```

```
      *orgstr* _ SF(jwp1) SE(jwp1);
   %check for non-null source%
      stid _ getsub(jwp1);
      IF getnxt(stid) = endfil AND ( FIND SF(stid)
      *nullsource* ENDCHR ) THEN
         err($"Nothing to send!");
   %check ident lists%
      getjauthor($workstr);
         IF *workstr* # *initsr* THEN %must check it%
            BEGIN
            *goodids* _ *workstr*;
            ckidlist($goodids, $badids, idfno);
            IF badids.L THEN %tell user the bad news%
               BEGIN
               IF goodids.L THEN setjauthor($goodids)
               ELSE setjauthor($initsr);
               *badids* _ "Mail not sent because of unknown
               ident(s): ", *badids*;
               err($badids);
               END;
            IF *goodids* # *workstr* THEN
            setjauthor($goodids);
            END;
      getjaction($workstr);
         IF workstr.L THEN %must check it%
            BEGIN
            *goodids* _ *workstr*;
            ckidlist($goodids, $badids, idfno);
            IF badids.L THEN %tell user the bad news%
               BEGIN
               setjaction($goodids);
               *badids* _ "Mail not sent because of unknown
               ident(s): ", *badids*;
               err($badids);
               END;
            IF *goodids* # *workstr* THEN
            setjaction($goodids);
            END;
      getjinfo($workstr);
         IF workstr.L THEN %must check it%
            BEGIN
            *goodids* _ *workstr*;
            ckidlist($goodids, $badids, idfno);
            IF badids.L THEN %tell user the bad news%
               BEGIN
               setjinfo($goodids);
               *badids* _ "Mail not sent because of unknown
               ident(s): ", *badids*;
               err($badids);
               END;
            IF *goodids* # *workstr* THEN
            setjinfo($goodids);
            END;
   %set up strings for later use%
      getjrfc($rfcnum);
      getjnumber($number);
```

```
            IF NOT rfcnum.L AND NOT number.L THEN  %kludge for
            now until fix background guy%
               BEGIN
               *number* _ "1234";
               setjnumber($"1234");
               END
            ELSE IF *number* # "1234" THEN %check it%
               BEGIN
               getjauthor($workstr);
               IF NOT FIND SF(*workstr*) [*initsr*] THEN
                  *workstr* _ *initsr*, SP, *workstr*;
               conjdir(TRUE);
               numstid _ openlock(0, jflname($"tcnumbers"));
               IF rfcnum.L THEN %check rfc number and journal
               number%
                  BEGIN
                  rnumstid _ openlock(0,
                  jflname($"Rfcnumbers"));
                  ckrfcnum(1, $rfcnum, $number, $workstr,
                  rnumstid, numstid);
                  END
               ELSE ckcnum(1, $number, $workstr, numstid);
               conjdir(FALSE);
               END;
         getjlink($linkaddr);
         getjtype($itemtype);
      %fill in time and date (of submission) fields%
         *datesr* _ NULL;
         dtfrmt(-1, $datesr);
         IF NOT FIND jwp1 > [";;;;"] [ "(J" / "(j" ] [";] ^p1
         _p1 ["Author(s):"] ["/] ^p3 [";] ^p4 _p4 THEN
            err($"Journal Header improperly formatted [mail
            not sent]");
         IF rfcnum.L THEN
            ST jwp1 _ SF(p1) p1, SP, *datesr*, "; .HJOURNAL=",
            "", "NWG/RFC# ", *rfcnum*, ".SPLIT;",  p3 p4, SP,
            *datesr*, "  ", *number*, "",  p1 SE(p1)
         ELSE
            ST jwp1 _ SF(p1) p1, SP, *datesr*, "; .HJOURNAL=",
            "", p3 p4, SP, *datesr*, "  ", *number*, "",  p1
            SE(p1);
      %build access list if private document%
         IF rdprvsts (jwp1.stfile) = $psprivate THEN
            BEGIN
            accesslist.L _ 0;
            clnidlst (getjaction ($workstr), $accesslist);
            clnidlst (getjinfo ($workstr), $accesslist);
            clnidlst (getjauthor ($workstr), $accesslist);
            clnidlst (getjclerk ($workstr), $accesslist);
            setjaxcess ($accesslist);
            END;
      %convert subcollections and distribution list -- should
      be done when tejournal file is loaded by background%
         convjdist();
         convjsubcol(idfno);
      %decide between storage as message or file%
```

```
      IF FIND jwp1 > [";;;; "] ^p1 CH ^p2 THEN
         IF rdprvsts (jwp1.stfile) = $psprivate
         OR getnxt (getnxt (jworkstid)) # endfil
         % OR NOT jtypchk (2000, jworkstid) % THEN
            ST p1 p2 _ "F
         ELSE ST p1 p2 _ "M;
   DROP (initrap) ;
   %conn. to TEJOURNAL directory and update the workfile
   into it%
      IF NOT number.L OR *number* = "1234" THEN  %build
      deferred-number file name%
         BEGIN
         FIND SF(*datesr*) [1$D] ^z2 < D ^z1 > [":] ^z5 <CH
         ^z4 $D ^z3 z5 > $D ^z6;
         *dfilstr* _ "D, z1 z2, z3 z4, z5 z6, *initsr*,
         ".NLP";
         END
      ELSE  %build pre-assigned number file name%
         *dfilstr* _ "J, *number*, ".NLN";
      *dirnam* _ NULL;
      &fl _ flntadr(jworkstid.stfile);
      jfnstr(fl.florig, $fnamestr);
      traping _ TRUE;
      capsav _ trapcc();
      connflag _ TRUE;
      condir(IF jdebug THEN $"KELLEY" ELSE $"TEJOURNAL",IF
      jdebug THEN $"KFJ" ELSE $jnlpsw,$dirnam,$dpassw);
      updtfl(jworkstid.stfile, newversion, $dfilstr);
      condir($dirnam,$dpassw,$dirnam,$dpassw);
      connflag _ FALSE;
      IF traping:=FALSE THEN notrapcc(capsav);
      fl.flnoclos _ FALSE;
      close(jworkstid.stfile);
      jworkstid _ 0;
      % Dispose of orgstr space. %
         freestring(&orgstr:=0, $dspblk);
      % Zap the Journal workfile. %
         INVOKE (jwsigtrap, jsubok) ;
         jworkstid _ orgstid;
         jworkstid.stfile _ open(0, $fnamestr);
         resetf(jworkstid.stfile);
         (jsubok):  %initialize the file%          5B6G1W18E
         &fl _ flntadr(jworkstid.stfile);
         fl.flnoclos _ TRUE;
         FIND SF(jworkstid) ^jwp1;
         initjwork();  %zap the Journal workfile%
         % commented out for now
         IF FIND SF(jworkstid) "INCOMPLETE" ^jwp1
         THEN ST SF(jworkstid) jwp1 _;
         %
   INVOKE (closetrap) ;
   IF number.L AND *number* # "1234" THEN
      BEGIN
      conjdir(TRUE);
      IF rfcnum.L THEN
         BEGIN %release RFC number%
```

```
            usedcnum($rfcnum, rnumstid);
            END;
         usedcnum($number, numstid); %release used journal
         number%
         conjdir(FALSE);
         END;
      IF linkaddr.L THEN
         BEGIN
         CASE *itemtype*[1] OF
            = "H: *mfname* _ "(Hard Copy -- Journal, ",
            *number*, ",)";
            ENDCASE *mfname* _ "( Journal, ", *number*, ",
            1:w)" ;
         %now insert the link%
            FIND SF(*linkaddr*) ^z1 SE(*linkaddr*) ^z2;
            caddexp($z1, $z2, lda(), $where);
               %where should the starting point of the
               expression be???? we leave it undefined%
            IF where = endfil THEN
               BEGIN
               *linkaddr* _ "Unable to insert link ",
               *mfname*, " at ", *linkaddr*;
               dismes(1, $linkaddr);
               END
            ELSE
               BEGIN
               FIND SF(*mfname*) ^z1 SE(*mfname*) ^z2;
               dismes(1, $linkaddr);
               cinstex($where, $z1, $z2, TRUE);
               END;
         END;
      dismes(1, $"Completed");
      IF idfno THEN close(idfno := 0);
      IF numstid.stfile THEN close(numstid.stfile := 0);
      IF rnumstid.stfile THEN close(rnumstid.stfile := 0);
      RETURN;
      (jwsigtrap) CATCHPHRASE;                          5B6G1AE
         BEGIN
         INVOKE (contcatch);
         CASE SIGNALTYPE OF = aborttype:
            BEGIN %must create it%
            jworkstid _ openwk(0, $fnamestr);
            TERMINATE;
            END;
         ENDCASE;
         CONTINUE;
         END;
      (initrap) CATCHPHRASE;                            5B6G1AF
         BEGIN
         CASE SIGNALTYPE OF = aborttype:
            BEGIN %must create it%
            conjdir(FALSE);
            IF numstid.stfile THEN close(numstid.stfile := 0);
            IF rnumstid.stfile THEN close(rnumstid.stfile :=
            0);
            IF traping := FALSE THEN notrapcc(capsav);
```

```
            IF connflag := FALSE THEN %connect back to user's
            directory%
               condir($dirnam,$dpassw,$dirnam,$dpassw);
            % Check orgstr; if empty, simply release string.
            If not, reset the origin statement to its original
            state. %
               IF &orgstr THEN
                  BEGIN
                  IF orgstr.L THEN
                     BEGIN
                     % Restore original origin statement. %
                        ST jwp1 _ *orgstr*;
                     END;
                  freestring(&orgstr, $dspblk);
                  END;
            END;
         ENDCASE;
         CONTINUE;
         END;
      (closetrap) CATCHPHRASE;                              5B6G1AG
         BEGIN
         INVOKE (contcatch) ;
         CASE SIGNALTYPE OF = aborttype:
            BEGIN %must create it%
            conjdir(FALSE);
            IF numstid.stfile THEN close(numstid.stfile := 0);
            IF rnumstid.stfile THEN close(rnumstid.stfile :=
            0);
            IF idfno THEN close(idfno := 0);
            IF traping := FALSE THEN notrapcc(capsav);
            END;
         ENDCASE;
         CONTINUE;
         END;
      END.

   (secdist)  %execute forward journal item request%
   PROCEDURE(number, distlist, actionflag);                 5B6G2
      %Accepts a string containing a catalog number, and a
      string containing an identlist and a flag (true if
      action, false if info-only).
         Distributes the indicated Journal document to the
         persons on the list%
      LOCAL char, jcstid, fl, traping, capsav, connflag;
      REF number, distlist, fl;
      LOCAL TEXT POINTER z1, z2, z3, z4;
      LOCAL STRING
         badids[500],     %list of bad idents%
         dpassw[15],      %password string%
         dirnam[50],      %directory name%
         dfilstr[150],    %tejournal file name%
         idstr[25],       %to hold an ident%
         comm[300],       %to hold the comment associated with
         an ident%
         fnamestr[150],   %file name string%
         tempsr[400];     %temporary work string%
```

```
        jcstid _ orgstid;
        capsav _ 0;
        traping _ connflag _ FALSE;
        INVOKE (closcatch) ;
        IF NOT (jcstid _ gtcatent (&number, 0, 0, FALSE)) THEN
           err ($"No such document");
        IF NOT vrprvacc (0, 0, jcstid, $initsr) THEN
           err ($"Private document; access denied to you");
        %check distribution list%
           ckidlist($distlist, $badids, idfno);
           IF badids.L THEN %tell user the bad news%
              BEGIN
              *badids* _ "Illegal ident(s) [mail not forwarded]:
              ", *badids*;
              err($badids);
              END;
        %put in ACTION or INFO-only designation%
           IF actionflag THEN %process as action list%
              *dirnam* _ " [ ACTION ] "
           ELSE %process as information-only list%
              *dirnam* _ " [ INFO-ONLY ] ";
           *tempsr* _ *distlist*;
           *distlist* _ NULL;
           %add ( [ ACTION ] )  or ( [INFO-ONLY] ) to each
           ident; must agree with format checked in (nls,
           jnlde1, distd1)%
              FIND SF(*tempsr*) ^z1;
              WHILE (FIND z1 $NP $(',) $NP ^z2 1$(LD / '^ / '& /
              '-) ^z1) DO
                 BEGIN
                 *idstr* _ +z2 z1;
                 IF FIND z1 $NP '( ^z2 ([')] ^z1 ^z3 _z3
                 /[ENDCHR] ^z1 ^z3) THEN
                    *comm* _ z2 z3
                 ELSE *comm* _ NULL;
                 *distlist* _ *distlist*, *idstr*, '(, *dirnam*,
                 *comm*, '), SP;
                 END;
           %terminate with semicolon%
              *distlist* _ *distlist*, '; ;
        ST jworkstid _ ";;;; S ", SF(jcstid) SE(jcstid);
        z2 _ getsub(jcstid);
        z2 _ ccopsta(jworkstid, levdown, z2, FALSE, 0);
        z2 _ cisstr(z2, &distlist, $sucdir);
        cisstr(z2, $initsr, $sucdir);
        %update to TEJOURNAL%
           *dfilstr* _ 'S, *number*, ".NLN";
           *dirnam* _ NULL;
           &f1 _ flntadr(jworkstid.stfile);
           jfnstr(f1.florig, $fnamestr);
           traping _ TRUE;
           capsav _ trapcc();
           connflag _ TRUE;
           condir(IF jdebug THEN $"IRBY" ELSE $"TEJOURNAL",IF
           jdebug THEN $"BLI" ELSE $jnlpsw,$dirnam,$dpassw);
           updtfl(jworkstid.stfile, newversion, $dfilstr);
```

```
            condir($dirnam,$dpassw,$dirnam,$dpassw);
            connflag _ FALSE;
            IF traping:=FALSE THEN notrapcc(capsav);
            fl.flnoclos _ FALSE;
            close(jworkstid.stfile);
            jworkstid _ 0;
            % Zap the Journal workfile. %
               INVOKE (jwcreate, jsubok) ;
               jworkstid _ orgstid;
               jworkstid.stfile _ open(0, $fnamestr);
               resetf(jworkstid.stfile);
               (secdok):  %initialize the file%          5B6G2S16E
               &fl _ flntadr(jworkstid.stfile);
               fl.flnoclos _ TRUE;
               initjwork();  %zap the Journal workfile%
               % commented out for now
               FIND SF(jworkstid) ^jwp1;
               ST jwp1 jwp1 _ "COMPLETED";
               %
         dismes(1, $"Completed");
         close(jcstid.stfile := 0);
         RETURN;
         (closcatch) CATCHPHRASE;                        5B6G2W
            BEGIN
            INVOKE (contcatch);
            CASE SIGNALTYPE OF = aborttype:
               BEGIN %close any open files%
               conjdir(FALSE);
               IF traping := FALSE THEN notrapcc(capsav);
               IF connflag := FALSE THEN %connect back to user's
               directory%
                  condir($dirnam,$dpassw,$dirnam,$dpassw);
               close(jcstid.stfile := 0);
               END;
            ENDCASE;
            CONTINUE;
            END;
         (jwcreate) CATCHPHRASE;                         5B6G2X
            BEGIN
            INVOKE (contcatch);
            CASE SIGNALTYPE OF = aborttype:
               BEGIN %must create it%
               jworkstid _ openwk(0, $fnamestr);
               TERMINATE;
               END;
            ENDCASE;
            CONTINUE;
            END;
         END.

      (contcatch) CATCHPHRASE;                           5B6G3
         BEGIN
         CONTINUE;
         END;
   %....Network Journal Procedures....%
      (njs) %network journal submission%
```

```
PROCEDURE (acs);                                                    5B6H1
   %----------%
   LOCAL sysmsg REF, ct, nxtstid, substid, injfn,
   titlestid, boundstid, tally, tailstrt, tailend;
   LOCAL STRING
      infile [40], control [200], authlist [200], distlist
      [200], clerk [20], title [200], errstr [500], level
      [30], workfile [40];
   LOCAL TEXT POINTER tp1, tp2, tp3, tp4, tp5;
   REF acs;
   %----------%
   %avoid scheduler pit%
      R1 _ 400000B;
      R2 _ 303B;
      !spriw();
   %open, read, and close control file%
      DROP (initrap) ;
      INVOKE (ctltrap) ;
      R1 _ acs [7]; ! HRRZS 1; injfn _ R1;
      IF NOT sysopen (injfn, read, chrtyp, $errstr) THEN
         hiortn (cat, $"Control file can't be opened: ",
         $errstr, 0);
      IF NOT isread (injfn, $control, CR) THEN
         hiortn (cat, $"Control file can't be read", 0, 0);
      sysclose (injfn, $errstr);
   %determine conversion type%
      ct _ 'S;
      IF FIND SF(*control*) ['';] ^tp1 _tp1 ^tp2 CH ^tp3 tp2
      THEN
         BEGIN
         CASE (ct _ READC .A (177B-SP)) OF
            ='S, %insert sequential%
            ='H, %heuristic w/o prev right justfiy%
            ='J, %heuristic w/ prev right justify%
            ='A, %insert assembler w/ structure%
            ='M: %insert assembler w/o structure%
               NULL;
            ENDCASE
               hiortn (cat, $"Valid conversion types are A,
               M, S, H, and J", 0, 0);
         ST tp1 tp3 _ NULL;
         END;
      ct _ 'S; %force sequential conversion for now%
   %verify and save clerk, author(s), addressee(s)%
      INVOKE (idtrap) ;
      IF NOT FIND SF(*control*) $(SP/',) ^tp1 1$(LD/'-)
      ^tp2 ['/] ^tp3 _tp3 ^tp4 [CR] ^tp5 _tp5 THEN
         hiortn (cat, $"Ident list format error", 0, 0);
      *clerk* _ + tp1 tp2;
      *initsr* _ *clerk*;
      *authlist* _ + tp1 tp3;
      *distlist* _ + tp4 tp5;
      IF NOT njsverids ($clerk, $authlist, $distlist,
      $errstr)
         THEN hiortn (cat, $"No such ident(s): ", $errstr,
         0);
```

```
      hioco (ok, $acs);
   %fetch name of sequential text file%
      INVOKE (jfntrap) ;
      R1 _ acs [7]; ! HRRZS 1; injfn _ R1;
      jfnstr (injfn, $infile);
   %initialize work file%
      *workfile* _ "<SYSTEM>[SEND-MAIL].", *initsr*,
      ";1;P707000";
      INVOKE (ini2trap, okay) ;
      jworkstid _ orgstid;
      jworkstid.stfile _ open (0, $workfile);
      initjwork ();
      FIND SF(jworkstid) ^jwp1;
      (okay):                                              586H1L7
      INVOKE (jfntrap) ;
      setjauthor ($authlist);
      setjaction ($distlist);
   %insert text of message/file%
      IF (substid _ getsub (jworkstid)) # jworkstid THEN
         cdelgro (substid, getail (substid), FALSE, 0);
      level.L _ 0;
      CASE ct OF
         ="S: %insert sequential%
            inseq (jworkstid, $level, $infile, tenfil);
         ="H: %heuristic insert sequential w/o prev right
         justify%
            inseqn (jworkstid, $level, $infile, FALSE);
         ="J: %heuristic insert sequential w/ prev right
         justify%
            inseqn (jworkstid, $level, $infile, TRUE);
         ="A: %insert assembler w/ Structure%
            inseq (jworkstid, $level, $infile, assfil);
         ="M: %insert assembler w/o Structure%
            inseq (jworkstid, $level, $infile, macfil);
         ENDCASE;
   %extract the title (if any)%
      titlestid _ boundstid _ tailstrt _ tally _ 0;
      IF ((nxtstid _ getnxt (jworkstid)) # endfil) THEN
         WHILE ((nxtstid # jworkstid) AND ((tally _
         tally+1) <= 15)) DO
            BEGIN
            IF FIND SF(nxtstid) ^tp1 [":] ^tp2 _tp2 $SP
            ^tp3 ([CR/EOL] < CH / SF(nxtstid)) $SP ^tp4
            THEN
               IF (tp2 [1] - tp1 [1]) <= title.M THEN
                  BEGIN
                  *title* _ + tp1 tp2;
                  IF *title* = "RE"
                  OR *title* = "TITLE"
                  OR *title* = "SUBJECT" THEN
                     BEGIN
                     titlestid _ nxtstid;
                     *title* _ tp3 tp4;
                     IF title.L THEN setjtitle ($title);
                     END;
                  END;
```

```
                IF FIND SE(nxtstid) "- - - -" THEN
                   BEGIN
                   boundstid _ nxtstid;
                   WHILE (NOT getftl (nxtstid)) DO
                      BEGIN
                      nxtstid _ getsuc (nxtstid);
                      IF FIND SE(nxtstid) "-------" THEN
                         BEGIN
                         tailstrt _ nxtstid;
                         WHILE (NOT getftl (nxtstid)) DO
                            nxtstid _ getsuc (nxtstid);
                         tailend _ nxtstid;
                         EXIT LOOP 2;
                         END;
                      END;
                   EXIT LOOP;
                   END;
                IF NOT FIND SF(nxtstid) [PT] THEN
                   BEGIN
                   boundstid _ nxtstid;
                   EXIT LOOP;
                   END;
                nxtstid _ getsuc (nxtstid);
                END;
             INVOKE (donetrap, done) ;
             IF tailstrt THEN
                cdelgro (tailstrt, tailend, FALSE, 0);
             IF boundstid THEN
                cdelgro (getnxt (jworkstid), boundstid, FALSE, 0);
             IF titlestid THEN cdelsta (titlestid, FALSE, 0);
             (done): INVOKE (jfntrap) ;                          5B6H1N7
          %null document?%
             IF getnxt (jworkstid) = endfil THEN
                cisstr (jworkstid, $nullsource, $"d");
          %journalize%
             INVOKE (subtrap) ;
             jsubmit ();
             hiortn (ok, $"Journalized", 0, 0);
          RETURN;
          (subtrap) CATCHPHRASE(:sysmsg);                        5B6H1R
             BEGIN
             CASE SIGNALTYPE OF = aborttype:
                BEGIN
                hiortn (cat, $"Submission error: ", &sysmsg, 0);
                END;
             ENDCASE;
             CONTINUE;
             END;
          (ctltrap) CATCHPHRASE(:sysmsg);                        5B6H1S
             CASE SIGNALTYPE OF
                = aborttype: hiortn (cat, $"Control file error: ",
                &sysmsg, 0);
             ENDCASE;
          (idtrap) CATCHPHRASE(:sysmsg);                         5B6H1T
             CASE SIGNALTYPE OF
                = aborttype: hiortn (cat, $"Ident verification
```

```
            error: ", &sysmsg, 0);
         ENDCASE;
      (jfntrap) CATCHPHRASE(:sysmsg);                          5B6H1U
         CASE SIGNALTYPE OF
            = aborttype: hiortn (cat, $"Work file init. error:
            ", &sysmsg, 0);
         ENDCASE;
      (ini2trap) CATCHPHRASE;                                  5B6H1V
         BEGIN
         CASE SIGNALTYPE OF = aborttype:
            BEGIN
            jworkstid _ openwk (0, $workfile);
            TERMINATE ;
            END;
         ENDCASE;
         CONTINUE;
         END;
      (donetrap) CATCHPHRASE;                                  5B6H1W
         BEGIN
         CASE SIGNALTYPE OF = aborttype:
            BEGIN
            TERMINATE ;
            END;
         ENDCASE;
         CONTINUE;
         END;
      END.
   (njsverids) %verify ident lists%
   PROCEDURE (clerk, authlist, distlist, errlst);              5B6H2
      %----------%
      LOCAL idfileno;
      LOCAL STRING ident [30];
      LOCAL TEXT POINTER tp1, tp2;
      REF clerk, authlist, distlist, errlst;
      %----------%
      idfileno _ open (0, jflname ($"Identfile"));
      INVOKE (vidtrap) ;
      errlst.L _ 0;
      %verify clerk%
         IF NOT oldid (&clerk, idfileno) THEN
            *errlst* _ *errlst*, *clerk*, SP;
      %verify authors%
         FIND SF(*authlist*) ^tp2;
         WHILE (FIND tp2 > $(SP/",) ^tp1 1$(LD/"-) ^tp2) DO
            BEGIN
            *ident* _ tp1 tp2;
            IF NOT oldid ($ident, idfileno) THEN
               *errlst* _ *errlst*, *ident*, SP;
            END;
      %verify addressees%
         FIND SF(*distlist*) ^tp2;
         WHILE (FIND tp2 > $(SP/",) ^tp1 1$(LD/"-) ^tp2) DO
            BEGIN
            *ident* _ tp1 tp2;
            IF NOT oldid ($ident, idfileno) THEN
               *errlst* _ *errlst*, *ident*, SP;
```

```
            END;
         IF errlst.L THEN BUMP DOWN errlst.L;
         close (idfileno := 0);
         RETURN (NOT errlst.L);
         (vidtrap) CATCHPHRASE;                                    5B6H2P
            BEGIN
            CASE SIGNALTYPE OF = aborttype:
               BEGIN
               IF idfileno THEN
                  sigclose (idfileno := 0);
               END;
            ENDCASE;
            CONTINUE;
            END;
         END.

      (hioco) %co-routine return to calling fork%
      PROCEDURE (outcome, acs);                                    5B6H3
         %----------%
         LOCAL STRING filename [40], error [200];
         REF acs;
         %----------%
         *filename* _ "NJS.SRC";
         acs [7] _ sgtjfn (100000000000B, $filename, $error);
         RETURN;
         R1 _ outcome;
         R2 _ 0;
         R4 _ $acs [7]; ! MOVE 3,7;
         !haltf();
         ! EXCH 3,7; ! MOVEM 3,(4);
         RETURN;
         END.
   % miscellaneous  %
      (jtypchk) PROC (maxchars, jstid); % check number of chars
      in a jworkfile  %                                            5B6I1
         % jstid is statement preceeding the body of the
         submssion %
         % Returns TRUE if <= max %
         LOCAL  stid, size;
         size _ 0;
         stid _ jstid;
         WHILE (stid _ getnxt(stid)) # endfil AND size <=
         maxchars DO
            size _ size + getstsize(stid);
         RETURN(IF size > maxchars THEN FALSE ELSE TRUE);
         END.

% IDENTSUPPORT ROUTINES %
   INCLUDE <nine,identsupport, sendmail>
FINISH
```

Useroptions subsystem
  You can alter how you interact with NLS to fit your own equipment, use patterns, and style by specifying the parameters controlled by Useroptions. The effects of Useroptions commands hold for future NLS sessions, until you use the commands again to change them specifically. Unless stated otherwise, each command takes effect immediately. The following commands are available (including Universal commands) in Useroptions.
% FIRST SEARCHES BRANCH. Index in <nls,>.
  commands
  ##<useroptions>## 1A
  CONTROLFUNCTION
  ##<controlfunctions>## 1B
  defaults
  ##<reset>## 1C
    %backlinks: <helpd, how>
  RMODE
  ##<recognition>## 1D
  RMODE2
  ##<recognition>## 1E
  Space (for tabs?)...
    Command syntax in NLS 8.5 for the "Spaces (for tabs)" command.
  ##<spaces>## 1F
    Backspace: Spaces (for tabs?) Backspace...
      Command syntax in NLS 8.5 for the "Spaces (for tabs) On" command.
    ##<spaces>## 1F1

commands 2
  Control (characters for terminal) DEVICES OK (control character)
  CONTROLFUNCTIONS (character(s)) CONTENT (echo as) CONTENT OK
  DEVICES = Ti (terminal), Nvt, Lineprocessor,
            Imlac, Execuport, 33-tty, 35-tty, or 37-tty
    The Useroptions command "Control" allows you to assign certain command CONTROLFUNCTIONS (like Command Accept, Repeat, etc.) to any special characters you choose. You can write one set of definitions for each kind of terminal (DEVICES) you use. 2A
    DEVICES = Ti (terminal), Nvt, Lineprocessor,
              Imlac, Execuport, 33-tty, 35-tty, or 37-tty
      Nvt means "Network Virtual Terminal". Specifying Imlac, or Lineprocessor will set up definitions only for the use of Display terminals. All others set up definitions for Typewriter terminals. When you go into TNLS from EXEC on one of those display terminals, you'll get the character set for Ti; going into TNLS via Simulate command gets you the set for the terminal you specify. 2A1
    CONTROLFUNCTIONS 2A2
      Ca = Command Accept
      ##<nls, universal ca>## 2A2A
      Cd = Command Delete
      ##<nls, universal cd>## 2A2B
      Rpt = <REPEAT>
      ##<nls, repeat>## 2A2C
      Insert = <ENTER>
      ##<nls, okinsert>## 2A2D
      Bc = Backspace Character

##<nls, universal backspace !character>## 2A2E
Bw = Backspace Word
##<nls, backspace !word>## 2A2F
Bs = Backspace Statement--erases whole TYPEIN field:
<CTRL-P> is the default.
Currently not defined for Lineprocessors. 2A2G
Litesc = Literal Escape
"Take the character which follows
as literal text instead of its special
function." 2A2H
Ignore = causes the characters you specify to be ignored
when they are encountered as input. This was
specifically written for use with noisy lines
that tend to generate DEL characters, etc. 2A2I
Sc = Shift Character (for an upper-case-only terminal
being used for upper/lower case). 2A2J
Sw = Shift Word (for an upper-case-only terminal
being used for upper/lower case). 2A2K
Tab = <CTRL-I> as the default.
##<nls, universal tab>##

effects

Users often change NLS control characters to avoid characters that mean something else on their local system. Some Control Functions have standard unalterable characters. All have other default assignments which will be overridden by any other special characters you define with this command. When the command asks you "character(s)" you may type as many keys for each character as you like. After "(echo as)" specify a single character that will print when you use the control character. <CTRL-N> can be used to specify no character. Repeat the command to set definitions for other terminals and other control functions. You can return to the defaults with the Reset Control command. You may see your current list of definitions with the Show Control command. 2A3

Backspace (and) Echo (with backspace symbol)/Overprint OK 2B

Currentcontext (length) NUMBER OK

The Useroptions command "Currentcontext" changes the NUMBER of characters that surround the character you are on as shown in response to typing a slash. The default value is 7 characters. There are Show and Reset Currentcontext commands. See address slash. 2C

Display (area) Right / Wraparound ...

The Useroptions "Display (area)" command allows you to specify the length of lines of text in video terminal windows (display areas) independent of the size of the window. There are Reset and Show Display commands. See also the Useroptions Output command. 2D

Right: Display (area) Right
(margin is column) CONTENT OK

The Useroptions "Display (area) Right" command allows you to specify the maximum length of the visible line by typing the number of the character position or "column" you wish to act as the right margin. The default is 72. If it is set to 0, the right margin will be automatically set to two or three characters less than the maximum for the display station you are using. If the right margin is larger than the size of your display window, the Display (area) Wraparound command allows you to specify the

character position on the screen where the line will be broken.
2D1

Wraparound: Display (area) Wraparound
(margin is column) CONTENT OK
The Useroptions command "Display (area) Wraparound" allows specification of the character position on the screen where the line will be broken for display right margins larger than the size of your display window. The right margin will be used to determine which words should appear in the next line. The number you give for Wraparound will only apply in the middle of lines with visible characters past that number and will start a new line exactly at that character even if it is in the middle of a word. The default wraparound number is 0 which means don't wraparound.
2D2

% Not implemented in NLS 9
Entry Subsystem/Program ...
The Useroptions command "Entry" allows you to specify the subsystem or program you will be using when you enter NLS.

Entry Subsystem SUBSYSTEM OK
Entry Program CONTENT OK

There are also Show and Reset Entry commands in Useroptions.
Exclude (subsystem/program) CONTENT OK
The Useroptions command "Exclude (subsystem/program)" allows you to delete subsystems and programs that are available to you when you enter NLS. For CONTENT specify the subsystem or program you wish to delete. This is the opposite of the Useroptions "Include" command. The "Reset Default" command will set your included sybsystems and programs to the system default.
2F
Execute (command in) SUBSYSTEM [Universal command].
##<execute>##
2G
External (Names Link File Address) CONTENT OK
The Useroptions command "External" is used to search for an external name. For CONTENT specify the address of a file with named statements containing links. See externalname. See also: Base Set External.
2H
Feedback FEEDSPECS OK
The Useroptions command "Feedback" turns noise words off and on. The default is "on". For your typewriter terminal sessions, you can specify how many letters of the command words and noise words appear. The default is 50. Also for typewriter terminal sessions, you can indent the beginning of each command specification. The default is no indenting. There are also Reset Feedback and Show Feedback commands. The Reset Feedback Mode command turns noise words on.
2I
FEEDSPECS = Terse, Verbose, Length CONTENT, or Indenting CONTENT.

Verbose, which is the default, gives you noisewords. Terse shuts off noisewords. Length and Indenting affect typewriter terminals only. To limit both noise words (if they're on) and completion of command words upon recognition (independently), use Length and give the number of characters you want for each; otherwise, all characters up to 50 will appear. To indent the beginning of each command, use Indenting and give the number of spaces to be indented for CONTENT; otherwise commands will not be

indented. 2I1
TNLS example:
USER C: Feedback C: Terse OK:
USER C: Goto C: Base OK:
BASE C: Append C: Statement A: 2
A: 1
T: and
BASE C:
Filereturn (ring entries) CONTENT OK
The Useroptions command "Filereturn" allows you to change the maximum size of your File Return Ring. The default is 10. The maximum allowed is 25. For CONTENT specify the number of entries for the ring. The commands won't take effect until you create a new window in DNLS or until your next NLS session. They will remain in effect for subsequent sessions until you change the size again. Each location remembered on your ring takes up space and too many could slow you down or cause problems. For this reason we recommend no more than 10 except for special cases. There are also "Return", "Show Return" and "Reset Filereturn" commands. 2J
Goto SUBSYSTEM OK [Universal command].
##<goto>## 2K
Help [Universal command]
##<nls, help>## 2L
Herald HERALDSPECS OK (for typewriters)
The Useroptions command "Herald" allows you to specify the number of characters of each subsystem name that will print as a herald, or to choose to have a star * print as the herald for every subsystem. There are Reset Herald and Show Herald commands. 2M
HERALDSPECS = Terse, Verbose, or Length.
For HERALDSPECS, you may choose terse which makes the herald for every subsystem a star *. Verbose (the default) makes the herald the first few letters of the current subsystem name. For Length, you may specify the number of letters of the name to be printed when Verbose is the chosen HERALDSPECS. The "Reset Herald Mode" command makes the herald "verbose" with four letters (such as USER when in Useroptions). 2M1
Include (subsystem/program) CONTENT OK
The Useroptions "Include" command allows you to choose which subsystems and programs are automatically available to you when you enter NLS. When specifying a program, precede the program name by the directory name. Use the Exclude command to delete subsystems and programs from your list of inclusions. Use the Show Default command to see your current default subsystem and program status. There are no default Programs included after resetting. 2N
programs available
##<programs, available>##
Jump [Universal command].
##<nls, universal !jump>## 2O
% The following command should be deleted out of the NLS 9 code, and is not documented.
Leveladjust (prompting) Off/On OK
Name (delimiter defaults) OK (left delimiter) CONTENT (right) CONTENT OK
The Useroptions command "Name" sets the statement name delimiters to the characters you specify for CONTENT. These will be in effect whenever you create a file or use the Reset Name delimiters command.

Only one character is allowed for the left delimiter and one for the right. There are Show and Reset Name commands. The Reset Name command returns your delimiters to NULL NULL. See name-delimiters.

2Q

Output Quickprint Right (margin is column) CONTENT OK
The Useroptions "Output" command allows you to specify the character position or "column" which will act as the right margin when using the Base Output Quickprint command. The default is 72. There are also Show Output, Reset Output, and Display (area) commands in Useroptions.

2R

Printoptions PRINTSPECS CONTENT OK
The Useroptions command "Printoptions" allows you to specify the number of spaces for level indenting and tab stop settings. These work for display as well as typewriter terminals and output printer devices. For typewriter terminal users, you can set the Left and Right margins used for Print commands. You can specify Bottom margin and Page size used at your terminal when pagination (viewspec E) is on in the Print commands. For display terminals, see also: display. There are Show and Reset Printoptions commands.

PRINTSPECS:

2S

Bottom: Printoptions Bottom (margin is line) CONTENT OK
The Useroptions command "Printoptions Bottom" allows you to specify for CONTENT a number for the number of lines from the top of the page down to where the printing will stop for that page. The default is 63. This takes effect immediately upon using the TNLS Print commands whenever viewspec E is on. See also: pagination, viewspec capital-E.

2S1

Indenting: Printoptions Indenting (per level) CONTENT OK
The Useroptions command "Printoptions Indenting" allows you to control the indenting of levels of NLS heirarchical structure. For CONTENT type a number for the number of spaces you want each level to be indented from the preceding level. The default is 3. This takes effect immediately. See also: level.

2S2

Left: Printoptions Left (margin is column) CONTENT OK
The Useroptions command "Printoptions Left" allows you to specify a Number for the column where your printing should start at the left margin of the page. Specify a number for CONTENT. The default is 0. This affects both paginated output (when viewspec E is on) and regular TNLS output immediately upon using the TNLS Print commands. See also: pagination, viewspec capital-E.

2S3

Page: Printoptions Page (size is (lines)) CONTENT OK
The Useroptions command "Printoptions Page" allows you to specify a number for CONTENT for the number of lines from the top of the page to the bottom of the page (the place below the page numbers where dashes mark a page break). The default is 66. This takes effect immediately upon using the Print commands whenever viewspec E is on. See also: pagination, viewspec capital-E.

2S4

Right: Printoptions Right (margin is column) CONTENT OK
The Useroptions command "Printoptions Right" allows you to specify a number for the column where your printing should stop at the right magin of the page. Specify a number for CONTENT. The number default is 72. This affects both paginated output (when viewspec E is on) and regular TNLS output immediately upon using the TNLS Print commands. There is also a Useroptions

"Output Quickprint Right (margin)" command. See also: pagination, viewspec capital-E. 2S5

Tab: Printoptions Tab (stop settings) CONTENT OK
The Useroptions command "Printoptions Tab" allows you to set the column locations where TAB will take you. You are allowed 10 tabstops per line. A tabstop cannot be set in column 1.
Use one of the following forms for CONTENT:
(1) t t t t t t t t
where the t's are seperated by spaces to the proper position for the tab stops. In TNLS you will want to enter a <CR> first.

(2) 8,16,24,32,40,48,56,64,72
Where the number represents the column position (starting from 1 not 0) for the next tab stop.
The default tabstops are 9,17,25,33,41,49,57,65,73,81,89,97,105. 2S6

Notes on tabs.
Tabstops cannot be set specifically for one online file in this version of Useroptions. Instead, the tabstops you set will affect any and all files you read. See Publication Tabs for Output Processor directives to specify the stops in specific "Output Printed" files. Tabs set by directives will not appear correctly in online or "Output Quickprinted" files unless the reader uses the Useroptions Printoptions Tab command to set his tabstops to be the same as those set in the TABSTOPS directive. When a statement formatted for one level is moved to a lower level, it is automatically indented 3 spaces. This can cause a tab to be moved past its intended stop and unintentionally go to the next stop (destroying your format). To avoid this problem, see the Useroptions Space (for tabs) command. 2S6A

Prompt PMODE OK
PMODE = Off or Partial or Full
The Useroptions command "Prompt" allows you to turn prompts off, to show just the basic alternatives in commands (Partial), or to show all options as well as alternatives (Full). The Reset Prompt command sets it to Full. To see what kind of prompting is currently set, use Useroptions Show Prompt command. 2T

Quit [Universal command].
##<nls, universal !quit>## 2U

Recognition (mode) RMODE OK
RMODE = Anticipatory
or Demand
or Fixed
or Terse (secondary mode) RMODE2
RMODE2 = Anticipatory, Demand, Fixed, or Terse
The Useroptions command "Recognition" allows you to change your recognition mode. Your choices for RMODE are--Fixed, Anticipatory, Demand, and Terse. With Terse, choose one of the above for the secondary recognition mode. The Reset Recognition command will set it to Terse, Terse. There is a Show Recognition command. See command recognition. 2V

Reset
The Useroptions "Reset" command returns your useroptions to the new user's system default. You can reset a particular useroption or use the "Reset All" command. See the description for the command

that sets each useroption to determine the new user's default for it. The "Reset Default" command resets the "Include" and "Exclude" Useroptions commands. 2W

default useroptions

To see the default useroptions, use the "Reset All" command to set them to the default, and the "Show Default" command to see them. 2W1

Return (ring entries) CONTENT OK

The Useroptions command "Return" allows you to change the maximum size of your statement return ring. The default is 10. The maximum allowed is 25. For CONTENT specify the number of entries for the ring. Each location remembered on your ring takes up space and too many could slow you down or cause problems. For this reason we recommend no more than 10 except for special cases. There are Useroptions "Show Return" and "Reset Return" commands. There is also a Useroptions "Filereturn" command. 2X

Show

The Useroptions "Show" command will show you the status of your current useroptions. You can Show a particular useroption or Show All. There is also a Reset command for each useroption. The "Show Default" command is for both the "Include" and "Exclude" Useroptions commands. For the list of useroptions, see commands. 2Y

syntax

Show All OK
Show Backspace (mode) OK
Show Currentcontext OK
Show Default (Subsystems and Programs) OK
Show Display (parameters) OK
Show External (Names Link) OK
Show Leveladjust OK
Show Name (delimiter defaults) OK
Show Output (parameters) OK
Show Programmer's (Template File) OK
Show Quickprint (parameters) OK
Show Return (ring sizes) OK
Show Spaces (for tabs status) OK
Show Viewspecs OK

Spaces (for tabs) OK/Off OK/On (justify to) Right (termination characters besides <CA> and TAB) CONTENT OK

The Useroptions command "Spaces (for tabs)" provides you with three alternatives. If you use "Spaces OK", the proper number of spaces to reach the next tab stop will be entered into your file when you push the tab key. The next character typed after the tab will appear in the column containing the tab stop. "Spaces On" inserts spaces and every character typed after the tab will backspace the cursor or typing head one space. At a typewriter terminal, the character is remembered, but not typed at this point. When 1) another tab, 2) a real carriage return, 3) a termination character, or 4) your OK is typed, the characters you input since the last tab will be typed and you will be 1) at the next tab stop, 2) at the beginning of the next line, 3) after the tab stop, or 4) ready for the next command. In this way, your tabular material is right justified to the tab stop. Use "Spaces Off" if you wish to have the actual character <TAB> entered in your file. 2Z

To guarantee proper spacing

limit the length of the tabbed line to be 71 (or 1 minus your

default right margin). In addition, subtract three for each level down the statement will ultimately be located below the top level of the file.

<BC>, <BW> and right justification.

BC and BW will backspace characters and words as if all the spaces and other characters had been entered without using tabs. For example, BC after typing a tab will cause your cursor or typing head to move back over one of the spaces entered by the tab. In this way, you can manually "right justify" tabular material such as sums of numbers by backspacing the number of characters to be typed before typing them. The "Spaces (for tabs) On" command will make this happen automatically.

Special features for typewriter terminals.

To insure that what you see on your terminal is what will be in the file, an unrecorded carriage return will be issued whenever you start to enter literal characters. Also, BC and BW will cause the type head to physically move back if they are entered during your first line of text or after a line of text preceded by a carriage return. When you backspace over a carriage return, the line preceding the carriage return is re-typed leaving your type head at the end of the preceding line. For this to work properly, you must be sure your BC, BW, and <CR> characters are set as follows for the type terminal your are using: 2Z2A

<BC> and <BW>

The button or control character you hit to backspace characters and words must be set to echo one character. This is because whenever you hit your BC or BW key, a character (usually slash) is normally echoed moving your type head forward. Then your type head will back over it before backing over the actual character or word to be erased. Your format will not look right when you enter the table if your BC or BW is not set to echo any characters or set to echo two characters. See Control. 2Z2A1

<CR> or <RETURN>

Your <CTRL-D> character is called <CA> or <OK> and is normally set to be <CR> or RETURN for typewriter terminals. Use the Useroptions "Control" command to set it to some other character or no character other than <CTRL-D>. Otherwise, if your RETURN key acts as your <CA> or <OK>, to enter a real carriage return (<CR>) you must precede it with a literal escape character and the four characters "<EOL>" will be typed, thus messing up your table. See Control. 2Z2A2

% There is a see also BC that points to this branch.

<BC>, <BW> and column overflow

When you type enough characters to fill all the positions in front of the tab stop except one, your bell will ring. If you continue to type more characters, all of the characters you input since the last tab will be typed preceded by a space and they will, of course, not be right justified. If you type BC after some characters after a tab while you are in the automatic backspace mode, the character or word you just typed will be deleted and your cursor or print head will move forward one space. If you type your BW key, you will be spaced all of the way back to the tabstop and be removed from the automatic

backspace mode. Otherwise BC and BW will backspace as if all the spaces and other characters had been entered without using tabs. For more information about backspacing in "Spaces (for tabs)" modes, see BC.
Changing tab stops
##<printoptions !tab>##
Startup (Commands Branch Address) CONTENT OK
The Useroptions command "Startup" allows you to specify certain commands to take place automatically every time you enter NLS. For CONTENT give the address of a statement or branch of commands that can be executed. Thereafter, whenever you enter NLS, the commands in the branch at that address will be executed as in the Process STRUCTURE command. <CTRL-O> halts processing. The "Reset Startup" command deletes any startup address. See also the "Include" command for automatic userprogram loading. 2A@
For automatic user-program or user-subsystem loading,
##<include>##
Processing commands
##<base, process>##
Syntax [Universal command].
##<nls, universal !syntax>## 2AA
Viewspecs Default VIEWSPECS OK
The Useroptions command "Viewspecs" allows you to select which particular viewspecs you want to be in force when you begin your next session. The Base command Set Viewspecs will change viewspecs, but they will then revert to the "Default" set next time you log in. The Base command "Reset Viewspecs" will return viewspecs immediately to that same status. Useroptions Reset Viewspecs command will return your "default" viewspecs to hjupmwyACEHJLP. There is also a Show Viewspecs command in Useroptions. 2AB
% backlinks: <base, reset !viewspecs>
% This command has been added in NLS 9, but it should not have been. It is not documented, and should be deleted out of the code.
Viewspecs Pompting On/Off OK
? for showing current alternatives [Universal command].
##<nls, universal !questionmark>##
; for talking [Universal command].
##<nls, universal !semicolon>##
< for showing subsystem stack [Universal command].
##<nls, universal !stack>##
> for showing current subsystem [Universal command].
##<nls, universal !stack>##
REPEAT <CTRL-B> [Universal command].
##<nls, !repeat>## 2AH
<CTRL-Q> HELP button [Universal command].
##<nls, ctrl-q>##
<CTRL-S> SYNTAX button [Universal command].
##<nls, notation>##
% rafi's documentation %
PREFACE 3A
The useroptions is a data structure that dictates the behaviour of the NLS system as the user sees it. The USEROPTIONS subsystem is a means to manipulate this data structure and adjust the behaviour of the system to the individual preferences. Note that the useroptions influences NLS and all subsystems running under it 3A1

This document lists and explains in detail all the information in the useroptions and the various possibilities to manipulate it. 3A2

In the following pages each command will be explained and the the syntax given. We adopt the following conventions (which allow precise yet informal definition of the syntax):.plexnum 101;.irest 3; 3A3

Command words appear in upper case. 3A3A

Noise words appear in lower case. 3A3B

Alternatives appear as statements of same level in a plex. 3A3C

An upper case word preceded by an atsign (@) refers to a rule defined elsewhere in the document. 3A3D

Commands are terminated with a colon (:)..irest 0; 3A3E

COMMANDS IN USEROPTIONS 3B

BACKSPACE 3B1

overview 3B1A

This command enables the user to define a mode for his backspacing. Two modes exist: OVERPRINT and ECHO and both apply to typewriter type terminals only. 3B1A1

In the OVERPRINT mode the typing head will move back appropriately when a backspace character is hit (applys to both backspace character and backspace word). This mode is usefull while typing in formatted text and thus retaining the format when typing. In the ECHO mode (default) a special backspace symbol will be typed and the typing head will keep on moving. Avoid using the OVERPRINT mode for terminal that do not have the capability of moving the typing head backwords. 3B1A2

In the OVERPRINT mode When you backspace over a carriage return, the line preceeding the carriage return is re-typed leaving your type head at the end of the preceeding line. In order for this to work properly, you must be sure your BC, and BW characters should be set (by your USERPROFILE) to echo with a single character. Your format will not look right when you enter a table if your BC or BW are not set to echo any characters or set to echo more than one character. 3B1A3

Syntax 3B1B

BACKSPACE and OVERPRINT OK: 3B1B1

BACKSPACE and ECHO with backscpace symbol OK: 3B1B2

CURRENTCONTEXT 3B2

Overview 3B2A

This command allows to change the number of characters that sorround the character you are on as shown in response to typing a slash (for typewriter terminals only). Default value is 7. 3B2A1

Syntax 3B2B

CURRENTCONTEXT length NUMBER OK: 3B2B1

DISPLAY 3B3

Overview 3B3A

The Display (area) Right command allows you to specify the maximum length of the visible line of text in video terminal windows (independently of the size of the window) by typing the number of the character position or "column" you wish to act as the right margin. The default is 72. If it is set to 0, the right margin will be automatically set

to two or three characters less than the maximum for the display station you are using. If the right margin is larger than the size of your display window, the Display (area) wraparound command allows you to specify the character position on the screen where lines should be broken. The right margin will be used to determine which words should appear in the next line. The number you give for Wraparound will only apply in the middle of lines with visible characters past that number and will start a new line exactly at that character even if it is in the middle of a word. The default wraparound number is 0 which means no wraparound. 3B3A1

Syntax 3B3B

DISPLAY area RIGHT margin is column NUMBER OK: 3B3B1

DISPLAY area WRAPAROUND margin is column NUMBER OK: 3B3B2

EXCLUDE 3B4

Overview 3B4A

This command allows you to delete subsystems and programs that are available to you (previously INCLUDEd) when you enter NLS. This command will take effect on your next session with NLS. 3B4A1

Syntax 3B4B

EXCLUDE PROGRAM @PROGNAME OK: 3B4B1

EXCLUDE SUBSYSTEM @SUBSYSNAME OK: 3B4B2

PROGNAME := any program currently loaded 3B4B4

SUBSYSNAME := any subsystem currently attached 3B4B5

FILERETURN 3B5

Overview 3B5A

This command allows to change the size of your File Return Ring. The default is 10 (maximum allowed is 25). The command will not take effect until you create a new window in DNLS or until your next NLS session. They will remain in effect for subsequent sessions until you change the size again. Each location remembered on your ring takes up space and too many could slow you down or cause problems. For this reason we recommend no more than 10 except for special cases. 3B5A1

Syntax 3B5B

FILERETURN ring entries NUMBER OK: 3B5B1

INCLUDE 3B6

Overview 3B6A

This command allows to include subsystems and programs that will be automatically available to you when you enter NLS. You may include only programs that are currently loaded or subsystems currently attached to you. Use the Exclude command to delete subsystems and programs from your list of inclusions. The default subsystems are Base, Sendmail, Useroptions, and Programs. 3B6A1

Syntax 3B6B

INCLUDE PROGRAM @PROGNAME OK: 3B6B1

INCLUDE SUBSYSTEM @SUBSYSNAME OK: 3B6B2

LEVELADJUST 3B7

Overview 3B7A

This command turns the LEVEL-ADJUST prompting off and on. When Leveladjust is "Off", the LEVEL-ADJUST field in

commands is unavailable; you cannot change levels when inserting, moving, and copying statements, etc. When Leveladjust is "On", the feature does appear in appropriate commands. 3B7A1

Syntax 3B7B

LEVELADJUST prompting @SWITCH OK: 3B7B1

SWITCH := 3B7B3
ON
OFF

NAME 3B8

Overview 3B8A

This command allows the setting of statement name-delimiters to the characters you specify. These will be in effect whenever you create a file or use the BASE Reset Name delimiters command. Only one character is allowed for the left delimiter and one for the right. Default delimiters are NULL NULL. 3B8A1

Syntax 3B8B

NAME delimiter defaults 3B8B1

left delimiter: CHARACTER right delimiter CHARACTER OK: 3B8B2

OUTPUT 3B9

Overview 3B9A

This command allows the setting of printing parameters for the various OUTPUT commands (excluding Quickprint). The following parameters can be set: 3B9A1

LEFT margin
RIGHT margin
BOTTOM margin
PAGE size
INDENTING per level
TAB stops settings

TAB stops settings text is of form: 3B9A2

c c c c c c 3B9A2A

where c stands for any character except SPACE, and represents the position for the tab setting, or of the form: 3B9A3

13,27,55,80

where the numbers represent the column position (starting from 1 not 0) for the next tab stop. If the latter form is terminated by "..." the tabsettings will be extrapolated from the last one set, by equal intervals as defined by the last two settings 3B9A4

Note: The use of TABs in the tab text is prohibitted. In the second form spaces may be used except within the digits of a number. 3B9A5

Syntax 3B9B

OUTPUT @PRINTSPEC NUMBER OK: 3B9B1

OUTPUT TAB stop setting @TABTEXT OK: 3B9B2

PRINTSPEC := 3B9B4

LEFT margin is column
RIGHT margin is column
BOTTOM margin is line
PAGE size is (lines)

```
            INDENTING per level
         TABTEXT := A text as defined above in the Overview.     3B9B5
   output                                                         3B10
      Overview                                                    3B10A
         This command allows the setting of printing parameters for
         the OUTPUT QUICKPRINT command . The following parameters
         can be set:                                             3B10A1
            LEFT margin
            RIGHT margin
            BOTTOM margin
            PAGE size
            INDENTING per level
            TAB stops settings
         For explanation of tab stop settings see the OUTPUT command
         (in USEROPTIONS).                                       3B10A2
      Syntax                                                      3B10B
         OUTPUT @PRINTSPEC NUMBER OK:                           3B10B1
         OUTPUT TAB stop setting @TABTEXT OK:                   3B10B2
   QUIT                                                           3B11
      Overview                                                    3B11A
         This command terminates your session with the USEROPTION
         subsystem and takes you back to BASE. Exection of this
         command involves writing out all the data for future use.
         Failing to execute this command will result in the loss of
         all the updated data.                                   3B11A1
      Sayntax                                                     3B11B
         QUIT OK:                                             3B11B1
   RESET                                                          3B12
      Overview                                                    3B12A
         This command allows to reset a specific useroption
         parameter to the system default or reset all parameters.
                                                                 3B12A1
      Syntax                                                      3B12B
         RESET ALL OK:                                        3B12B1
         RESET @RESETSPEC OK:                                 3B12B2

         RESETSPEC :=                                          3B12B4
            BACKSPACE mode
            CURRENTCONTEXT length                            3B12B4B
            DEFAULT Subsystems and Programs                  3B12B4C
            DISPLAY area RIGHT margin
            DISPLAY area WRAPAROUND margin
            EXTERNAL Names Link File Address                 3B12B4F
            FILERETURN ring size
            NAME delimiters
            OUTPUT @PRNTRESET                                3B12B4I
            QUICKPRINT @PRNTRESET
            RETURN ring size                                 3B12B4K
            SPACES for tabs to OFF
            STARTUP Commands Branch Address                  3B12B4M
            VIEWSPECS DEFAULTS                               3B12B4N
            VIEWSPECS PROMPTING                              3B12B4O
         PRNTRESET :=                                          3B12B5
            LEFT margin
            RIGHT margin
            BOTTOM margin
```

USERPROFILE). 3B15A2
The process of right justification is described in the follwing. The proper number of spaces to reach the next tab stop will be entered into your file when you type the tab key. Every character typed after the tab will backspace the cursor or typing head one space. At a typewriter terminal, the character is remembered, but not typed at this point. When another tab, a real carriage return, a termination character, or your OK is typed the characters you input since the last tab will be typed and you will be at the next tab stop, at the beginning of the next line, after the tab stop, or ready for the next command as appropriate. 3B15A3
When you type enough characters to fill all the positions in front of the tab stop except one, your bell will ring. If you continue to type more characters, all of the characters you input since the last tab will be typed preceded by a space and they will, of course, not be right justified. 3B15A4

Syntax 3B15B
SPACES for tabs OK: 3B15B1
SPACES for tabs OFF OK: 3B15B2
SPACES for tabs ON justify to LEFT OK: 3B15B3
SPACES for tabs ON justify to RIGHT termination characters TEXT OK: 3B15B4

STARTUP 3B16

Overview 3B16A
This command allows you to specify certain commands to take place automatically every time you enter NLS. For LINK give the ADDRESS of a statement or branch of commands that can be executed. Thereafter, whenever you enter NLS, the commands in the branch at that address will be executed as in the Process STRUCTURE command OF BASE. 3B16A1

Syntax 3B16B
STARTUP commands branch address LINK OK: 3B16B1

LINK := any legal NLS link 3B16B3

VIEWSPECS 3B17

Overview 3B17A
This command allows you to select which particular VIEWSPECS you want to be in force when you begin your next NLS sessions and whether you will be prompted for viewspecs. 3B17A1
The Base command Set Viewspecs will change viewspecs, but they will then revert to the "Default" set next time you log in. The BASE command Reset Viewspecs will return Viewspecs immediately to those defined by Useroptions. Useroptions "Default" viewspecs are hjnpuwzACEGJLP. 3B17A2
Setting "Viewspecs Prompting" to Off will inhibit NLS prompting for VIEWSPECS whenever it usually does. The BASE commands Set (or Reset) Viewspecs will still be available. The command takes effect immediatelly. 3B17A3

Syntax 3B17B
VIEWSPECS DEFAULT @VUSPECS OK: 3B17B1
VIEWSPECS PROMPTING @SWITCH OK: 3B17B2

```
        VUSPECS := legal NLS viewspecs string                    3B1784
    Miscellanous                                                   3B18
      System defaults are:                                         3B18A
        Backspace mode: ECHO                                       3B18A1
        Current context length: 7                                  3B18A2
        Default subsystems: BASE, USEROPTIONS, PROGRAMS, SENDMAIL.
                                                                   3B18A3
        Display area                                               3B18A4
          Right margin: 72
          Wraparoun margin: 0 (no wraparound)
        External name link: None.                                  3B18A5
        Filereturn ringsize: 10                                    3B18A6
        Leveladjust prompting: ON                                  3B18A7
        Name delimiters                                            3B18A8
          Left: NULL Right: NULL
        Output and Quickprint                                      3B18A9
          Left margin: 0
          Right margin: 72
          Bottom margin: 60
          Page size: 66
          Indenting per level: 3
          Tab stop setting: 9,17,25,...
        Retrun ringsize: 10                                        3B18A10
        Spaces for tab: OFF (Justify to LEFT)                      3B18A11
        Startup commands branch: None.                             3B18A12
        Viewspecs                                                  3B18A13
          Default: hjnpuwzACEHJLP ( all lines all levels )
          Prompting: ON
%source-code%                                                            4
  %grammar%                                                             4A
    FILE uogram % (arcsubsys,cgcml,) (arcsubsys,cgcml9,) to
    (reinine,useroptions.cml,) %
    % COMPILING INSTRUCTIONS %
      INCLUDE <nine, nls-grammar, flags !subsystems>
    % DECLARATIONS %
      INCLUDE <nine, nls-grammar, declarations !universal>
      DECLARE COMMAND WORD % typein selector %
        "TYPEIN" = 127 SELECTOR TYPEIN = TEXT;
      DECLARE COMMAND WORD
        % useroptions %
          "ALL" = 70,
          "BACKSPACE" = 71,
          "BOTTOM" = 72,
          "CURRENTCONTEXT" = 73,
          "DEFAULT" = 74,
          "DISPLAY" = 75,
          "ECHO" = 76,
          "ENTRY" = 77,
          "EXCLUDE" = 78,
          % "EXTERNAL" = 59, %
          % "FILERETURN" = 67, %
          "INCLUDE" = 79,
          "INDENTING" = 80,
          "LEFT" = 81,
          "LEVELADJUST" = 82,
          % "NAME" = 32, %
```

```
            % "OFF" = 63, %
            % "ON" = 62, %
            "OUTPUT" = 83, "QUICKPRINT" = 83, % these are the same %
            "OVERPRINT" = 84,
            "PAGE" = 85,
            "PRINTOPTIONS" = 86,
            "PROGRAMMER'S" = 87,
            "PROMPTING" = 88,
            % "RETURN" = 62, %
            "RIGHT" = 89,
            "SPACES" = 90,
            "STARTUP" = 91,
            % "VIEWSPECS" = 57, %
            "WRAPAROUND" = 92,
            "DIRECTORY" = 112,
         % userprofile %
            "ANTICIPATORY" = 93,
            "CONTROL" = 94,
            "DEMAND" = 95,
            "NOISEWORD" = 96,
            "FIXED" = 97,
            "FULL" = 98,
            "HERALD" = 99,
            "LENGTH" = 100,
            "MODE" = 101,
            "PARTIAL" = 102,
            "PROGRAM" = 103,
            "PROMPT" = 104,
            "RECOGNITION" = 105,
            "SUBSYSTEM" = 106,
            "TERSE" = 107,
            "TTY" = 108,
            "VERBOSE" = 109,
            "COMMAND-WORD" = 110,
            "CONDITIONAL" = 111;
      DECLARE VARIABLE par1, par2, par3, par4;
      DECLARE FUNCTION
         % useroptions %
            xuobckspc, xuocurcon, xuodflt, xuodisplay, xuoentry,
            xuoextn, xuoinit, xuolevel, xuonamed, xuooutput,
            xuoprogtemp, xuoprtdir, xuoquickprint, xuoreset,
            xuoringsize, xuosget, xuoshow, xuospfortab, xuostup,
            xuoterm, xuoviewspecs,
         % userprofile %
            xupcontchar, xupcmndwrd, xupdfltool, xupinit, xuphandle,
            xupherald, xupnoisewrd,  xupprompt, xuprecognition,
            xupreset, xupshow, xupstup, xupterm,                4A3E2A
         % common %
            xureset, xushow;
   % rules %
      % useroptions %
         printparam =
            (
            par1 _
               (
                  ("RIGHT" / "LEFT") <"margin is column"> /
```

```
                "BOTTOM" <"margin is line"> /
                "PAGE" <"size is (lines)"> /
                "INDENTING" <"per level">
            )
         par2 _ LSEL(#"INTEGER")
         )
         /
         (
         par1 _ "TAB" <"stop settings">
         par2 _ LSEL(#"TEXT")
            %TAB text is of form
            c       c       c       c       c               c       ,
            where c stands for any char except SP, or of the form
            13,27,55,80 where the number represents the column
            position (starting from 1 not 0) for the next tab
            stop. If the text is terminated by "..." the
            tabsetting will be extrapolated from the last tab
            defined by equal intervals defined by the last to
            settings%
         );
      prntset =
         par2 _
            ( ("RIGHT" / "LEFT" / "BOTTOM")
               <"margin">
            / "PAGE" <"size">
            / "INDENTING" <"per level">
            / "TAB" <"stop settings">
            );
      swtch =
         ("ON" / "OFF"!L2!);
   % userprofile %
      devopt =                                                    4A4B1
         (
              "DISPLAY"! 1 !
            / "TYPEWRITER"! 2!
         );
      function =                                                  4A4B2
         (
              "CA"!1!
            / "CD"!L2 2!
            / "RPT"!3!
            / "OPTION"!12!
            / "BC"!4!
            / "BW"!L2 5!
            / "BS"!L2 6!
            / "LITESC"!7!
            / "IGNORE"!8!
            / "SC"!9!
            / "SW"!L2 10!
            / "TAB"!11!
         );
   % COMMON RULES %
      INCLUDE <nine, nls-grammar, rules !universal>
% commands % SUBSYSTEM useroptions KEYWORD "USEROPTIONS"
   INITIALIZATION
      uoinit=
```

```
        % useroptions %
           xuoinit()
        % userprofile %
           xupinit();
TERMINATION
   uoterm =
        % useroptions %
           xuoterm()
        % userprofile %
           xupterm();
% useroptions %
   backspace COMMAND =                                              4A5C1
      "BACKSPACE" <"and">
         par1 _ ("OVERPRINT" / "ECHO" <"with backspace
         symbol">)
         CONFIRM
         xuobckspc( par1 );
   curcon COMMAND = %current content length%                        4A5C2
      "CURRENTCONTEXT" <"length">
         %number of chars to be printed for each side of CM in
         slash command%
         par1 _ LSEL(#"INTEGER")
         CONFIRM
         xuocurcon(par1);
   display COMMAND = %display area parameters%                      4A5C3
      "DISPLAY" <"area">
         par1 _ ( "RIGHT" / "WRAPAROUND" )
         <"margin is column">
         par2 _ LSEL(#"INTEGER")
         CONFIRM
         xuodisplay(par1, par2);
   dflt COMMAND = % include / exclude %                             4A5C4
      par1 _ ( "INCLUDE" / "EXCLUDE"!L2! )
         <"subsystem/program">
         par2 _ LSEL(#"OLDFILENAME")
         CONFIRM
         xuodflt( par1, par2);
   entry COMMAND = %entry subsystem/program%                        4A5C5
      "ENTRY"!L2!
      param _ "SUBSYSTEM"
         subslist _ xuosget()  %get names of default substems%
         param2 _  CW: subslist
         CONFIRM
      xuoentry(param, param2);
   extn COMMAND = %external names link file address%                4A5C6
      "EXTERNAL"!L2! <"Names Link File Address">
         par1 _ LSEL(#"LINK")
         CONFIRM
         xuoextn( par1 );
   levl COMMAND = %level adjust prompting%                          4A5C7
      "LEVELADJUST" <"prompting">
         par1 _ swtch
         CONFIRM
         xuolevel(par1);
   name COMMAND = %name delimiter defaults%                         4A5C8
      "NAME" <"delimiter defaults">
```

```
        CONFIRM
        CLEAR
        <"left delimiter"> par1 _ LSEL(#"CHARACTER")
        <"right delimiter"> par2 _ LSEL(#"CHARACTER")
        CONFIRM
        xuonamed(par1,par2);
printoptions COMMAND = %printing parameters%                    4A5C9
   "PRINTOPTIONS"!L2!
        printparam
        CONFIRM
        xuooutput(par1, par2);
%+PROGRAMMERS%                                                  4A5C10
programmers COMMAND = %programmer's template file%              4A5C11
   "PROGRAMMER'S"!L2! <"Template File Name">
        par1 _ LSEL(#"LINK")
        CONFIRM
        xuoprogtemp( par1 );
%+PROGRAMMERS%                                                  4A5C12
output COMMAND =                                                4A5C13
   "OUTPUT"
        (
           ( "QUICKPRINT" %quickprint parameters%
             % printparam should be used here when all
             alternatives are implemented %
             par1 _ "RIGHT" <"margin is column">
             par2 _ LSEL(#"INTEGER")
             CONFIRM
             xuoquickprint(par1, par2)
           )
        /
           ( "DIRECTORY" <"Default for print/quickprint">
             par1 _ LSEL(#"WORD")
             xuoprtdir( par1 )
           )
        );
reset COMMAND =                                                 4A5C14
   "RESET"!L2!
        ( par2 _ NULL par3 _ NULL
        % useroptions %
           (par1 _
              ( "BACKSPACE" <"mode">
              / "CURRENTCONTEXT" <"length">
              / "DEFAULT"!L2! <"Subsystems and Programs">
              / "DISPLAY" <"area">
                 par2 _ ( "RIGHT" / "WRAPAROUND" ) <"margin">
                    #"DISPLAY"
              / "ENTRY"!L2!
                 par2 _ "SUBSYSTEM"
                    #"ENTRY"
              / "EXTERNAL" <"Names Link">
              / "NAME" <"delimiters">
              / "OUTPUT"
                 ( "QUICKPRINT" par2 _ "RIGHT" <"margin">
                 / par2 _ "DIRECTORY" <"default"> )
                 #"QUICKPRINT"
              / "PROGRAMMER'S"!L2! <"Template File">
```

```
                / "PRINTOPTIONS"!L2!
                   prntset #"PRINTOPTIONS"
                / "SPACES"!L2! <"for tabs to OFF">
                / "STARTUP" <"Commands Branch Address">
                / "VIEWSPECS"
                   par2 _ ("DEFAULT" / "PROMPTING" )
                   #"VIEWSPECS"
                / ("RETURN"!L2! / "FILERETURN" )  <"ring size">

                )
             CONFIRM
             xuoreset(par1, par2))
          / % userprofile %
             (par1 _
                ( "CONTROL"!L2! <"characters for terminal">
                   (
                        par2 _ devopt <"function"> par3 _ (
                      function / "ALL" )
                      / par2 _ "ALL" par3 _ NULL
                   ) #"CONTROL"
                / "NOISEWORD" !L2!
                   par2 _ ("MODE" / "LENGTH" ) #"NOISEWORD"
                                                        4A5C14A3A2A
                / "COMMAND-WORD" !L2!
                   par2 _ ("MODE" / "LENGTH" ) #"COMMAND-WORD"
                                                        4A5C14A3A3A
                / "HERALD"
                   par2 _ ("MODE" / "LENGTH") #"HERALD"
                                                        4A5C14A3A4A
                / "PROMPT" <"mode">
                / "RECOGNITION" <"mode">
                )
             CONFIRM                                    4A5C14A3B
             xupreset(par1, par2, par3))                4A5C14A3C
          / % common %
             ( "ALL"
             CONFIRM                                    4A5C14A4B
             xureset())                                 4A5C14A4C
          ); % end of RESET %
       ringsize COMMAND = %return/filereturn ring size%      4A5C15
          par1 _ ("RETURN"!L2! / "FILERETURN" )  <"ring entries">
             par2 _ LSEL(#"INTEGER")
             CONFIRM
             xuoringsize(par1, par2);
       show COMMAND =                                        4A5C16
          "SHOW"
             ( par2 _ NULL
             % useroptions %
                (par1 _
                   ( "BACKSPACE" <"mode">
                   / "CURRENTCONTEXT"
                   / "DEFAULT"!L2! <"Subsytems and Programs">
                   / "DISPLAY" <"parameters">
                   / "ENTRY"!L2! <"subsytem">
                   / "EXTERNAL" <"Names Link">
                   / "LEVELADJUST"
```

```
                / "NAME" <"delimiter defaults">
                / "OUTPUT"
                / "PROGRAMMER'S"!L2! <"Template File">
                / "PRINTOPTIONS"!L2!
                / "RETURN"!L2! <"ring sizes">
                / "SPACES" !L2! <"for tabs status">
                / "STARTUP" <"commands branch address">
                / "VIEWSPECS"
                )
             CONFIRM
             xuoshow(par1))
          / % userprofile %
             (par1 _
                ( "CONTROL"!L2! <"characters for terminal">
                par2 _ devopt #"CONTROL"
                / "NOISEWORD" !L2! <"mode">
                / "COMMAND-WORD" !L2! <"mode">
                / "HERALD" <"mode">
                / "PROMPT" <"mode">
                / "RECOGNITION" <"mode">
                )
             CONFIRM                                          4A5C16A3B
             xupshow(par1,par2))                              4A5C16A3C
          / % common %
             ( "ALL"
             CONFIRM                                          4A5C16A4B
             xushow())                                        4A5C16A4C
       ); % end of SHOW %
space COMMAND = %space for tabs/backspace/termination%
                                                              4A5C17
   "SPACES" !L2! <"for tabs">
      par2 _ NULL
      par3 _ NULL
      par1 _
         (
            "OFF" CONFIRM
          / CONFIRM #"ON"
          / "ON" !L2! <"justify to">
             (
                  par2 _ "LEFT"
               / par2 _ "RIGHT" CLEAR
                  <"termination characters besides <CA>
                  and TAB"> par3 _ LSEL( #"TEXT" )
             )
             CONFIRM #"ON"
         )
      xuospfortab(par1, par2, par3);
startup COMMAND =                                             4A5C18
   "STARTUP"!L2! <"Commands Branch Address">
      par1 _ LSEL(#"LINK")
      CONFIRM
      xuostup( par1 );
vuspecs COMMAND =                                             4A5C19
   "VIEWSPECS"
      (
         ( par1 _ "DEFAULT" par2 _ viewspecs() )
```

```
                /
                ( par1 _ "PROMPTING" par2 _ swtch )
            )
            CONFIRM
            xuoviewspecs(par1, par2);
      % userprofile %
         contchar COMMAND =  % control character %                    4A5D1
            "CONTROL"!L2! <"characters for terminal">
            par1 _ devopt                                             4A5D1B
            CONFIRM                                                   4A5D1C
            CLEAR <"control function">                                4A5D1D
            par2 _ function                                           4A5D1E
            <"character(s)"> par3 _ LSEL (#"TYPEIN")
            <"echo as"> par4 _ LSEL(#"TYPEIN")
            CONFIRM                                                   4A5D1H
            xupcontchar(par1, par2, par3, par4);                      4A5D1I
         noisewrd COMMAND =                                            4A5D2
            "NOISEWORD" !L2!
            (
               "MODE" par1 _ ( "VERBOSE" / "TERSE" ) par2 _ NULL
               / par1 _ "LENGTH" <"to be"> par2 _ LSEL(#"INTEGER")
            )
            CONFIRM                                                   4A5D2D
            xupnoisewrd(par1, par2);                                  4A5D2E
         cmndwrd COMMAND =                                             4A5D3
            "COMMAND-WORD" !L2!
            (
               "MODE" par1 _ ( "VERBOSE" / "TERSE" ) par2 _ NULL
               / par1 _ "LENGTH" <"to be"> par2 _ LSEL(#"INTEGER")
            )
            CONFIRM                                                   4A5D3D
            xupcmndwrd(par1, par2);                                   4A5D3F
         herald COMMAND =                                              4A5D4
            "HERALD" !L2!
            par2 _ NULL                                               4A5D4B
            (
               "MODE" par1 _ ( "VERBOSE" / "TERSE" )
               / par1 _ "LENGTH" <"to be"> par2 _ LSEL(#"INTEGER")
            )
            CONFIRM                                                   4A5D4E
            xupherald(par1, par2);                                    4A5D4F
         prompt COMMAND = % prompt mode %                              4A5D5
            "PROMPT"
            par1 _ ("CONDITIONAL" / "FULL" / "OFF" / "PARTIAL")
                                                                      4A5D5B
            CONFIRM                                                   4A5D5C
            xupprompt(par1);                                          4A5D5D
         recognition COMMAND =                                         4A5D6
            "RECOGNITION" <"mode">
            (
               par2 _ NULL                                          4A5D6B1
               par1 _ "TERSE" / "ANTICIPATORY" / "DEMAND" / "FIXED"
                                                                    4A5D6B2
            )
            CONFIRM                                                   4A5D6D
            xuprecognition(par1, par2);                               4A5D6E
```

```
    INCLUDE <nine, nls-grammar, commands !universal>
    END.
FINISH of uogram
    %%                                                    4A6A
```

```
%BE%                                                                    4B
   FILE uo %(arcsubsys,x110,) (arcsubsys,1109,) to
   (reinine,useroptions.subsys,)%
   % useroption/userprofile x-routines and other support routines %
      NOMESS
   % DECLARATIONS %
      % package table %
         (useroptions) _ (                                              4B3A1
            % useroptions %
               $"XUOBCKSPC" , $xuobckspc,
               $"XUOCURCON" , $xuocurcon,
               $"XUODFLT" , $xuodflt,
               $"XUODISPLAY" , $xuodisplay,
               $"XUOENTRY", $xuoentry,
               $"XUOEXTN" , $xuoextn,
               $"XUOINIT" , $xuoinit,
               $"XUOLEVEL" , $xuolevel,
               $"XUONAMED" , $xuonamed,
               $"XUOOUTPUT" , $xuooutput,
               $"XUOPROGTEMP" , $xuoprogtemp,
               $"XUOPRTDIR" , $xuoprtdir,
               $"XUOQUICKPRINT" , $xuoquickprint,
               $"XUORESET" , $xuoreset,
               $"XUORINGSIZE" , $xuoringsize,
               $"XUOSGET", $xuosget,
               $"XUOSHOW" , $xuoshow,
               $"XUOSPFORTAB" , $xuospfortab,
               $"XUOSTUP" , $xuostup,
               $"XUOTERM" , $xuoterm,
               $"XUOVIEWSPECS" , $xuoviewspecs,
            % userprofile %
               $"XUPCONTCHAR" , $xupcontchar,
               $"XUPINIT" , $xupinit,
               $"XUPNOISEWRD" , $xupnoisewrd,
               $"XUPCMNDWRD" , $xupcmndwrd,
               $"XUPHERALD" , $xupherald,
               $"XUPPROMPT" , $xupprompt,
               $"XUPRECOGNITION" , $xuprecognition,
               $"XUPRESET" , $xupreset,
               $"XUPSHOW" , $xupshow,
               $"XUPTERM" , $xupterm,
            % common %
               $"XURESET" , $xureset,
               $"XUSHOW" , $xushow,
               0,0);
      % command words %
         DECLARE CONSTANT
            % useroptions %
               all = 70,
               backspace = 71,
               bottom = 72,
               currentcontext = 73,
               default = 74,
               display = 75,
               echo = 76,
               entry = 77,
```

```
            exclude = 78,
            external = 59,
            filereturn = 67,
            include = 79,
            indenting = 80,
            left = 81,
            leveladjust = 82,
            name = 32,
            off = 63,
            on = 62,
            output = 83, quickprint = 83,  % these are the same %
            overprint = 84,
            page = 85,
            printoptions = 86,
            program = 103,
            progrmrs = 87,
            prompting = 88,
            right = 89,
            spaces = 90,
            startup = 91,
            subsystem = 106,
            viewspecs = 57,
            wraparound = 92,
            directory = 112,
         % userprofile %
            ca = 1 ,
            cd = 2 ,
            rpt = 3 ,
            bc = 4 ,
            bw = 5 ,
            bs = 6 ,
            litesc = 7 ,
            ignore  = 8 ,
            sc = 9 ,
            sw = 10 ,
            tab = 11 ,
            option = 12 ,
            anticipatory = 93 ,
            control = 94 ,
            demand = 95 ,
            noisewrd = 96 ,
            fixed = 97 ,
            full = 98 ,
            herald = 99 ,
            lngth = 100 ,
            mode = 101 ,
            partial = 102 ,
            prompt = 104 ,
            recognition = 105 ,
            terse = 107 ,
            verbose = 109 ,
            cmndwrd  = 110 ,
            conditional = 111 ;
      % general use %
         DECLARE CONSTANT
            in = 201,
```

```
        out = 202,
        away = 203;
    DECLARE STRING str[200], longstr[400], msg[1000];
    DECLARE uarray[5000B], uarend;  % keep those together %
% userprofile data structures etc %
    REF toolst, cntchr, profil;
%%                                                    483E
```

```
% useroptions routines %
   % x-routines %                                                    4B4A
      (xuobckspc) % BACKSPACE x-routine %
      PROCEDURE (bsmode REF);                                       4B4A1
         %
            FUNCTION
               set new backspace mode
            ARGUMENTS
               bsmode - command word - new bsmode
            PROCEDURE VALUE
               TRUE always
         %
         % set the bsmode %
            CASE ELEM #bsmode#[1] OF
               = echo: bckspc _ FALSE;
               = overprint: bckspc _ TRUE;
            ENDCASE;
         RETURN;
         END.
      (xuocurcon) % CURRENT CONTEXT x-routine %
      PROCEDURE (length);                                           4B4A2
         %
            FUNCTION:
               set new currentcontext length (in tslshchars)
            ARGUMENTS:
               length - integer - new length (must be positive)
            PROCEDURE VALUE
               FALSE if length illegal TRUE otherwise
         %
         IF length < 1 THEN ABORT( 0, $"illegal length");
         % set the length %
            tslshchars _ length;
         RETURN;
         END.
      (xuodflt) % INCLUDE/EXCLUDE x-routine %
      PROCEDURE (type REF, link REF);                               4B4A3
         %
            FUNCTION:
               x-routine for INCLUDE, EXCLUDE commands.
            ARGUMENTS:
               type - command word - exclude / include
               link - literal - name of prog/subsys.
            PROCEDURE VALUE
               TRUE always
            ALGORITHM:
               The new name is included in the list (if it is not
               there already) and then:
                  (a) include: Do nothing (already done).
                                                                4B4A3A4A1
                  (b) exclude: delete the name from list.
                                                                4B4A3A4A2
         %
         LOCAL i, found, index, pt REF;
         LOCAL TEXT POINTER tp1, tp2;
         LOCAL STRING sysname[100];
         % create the name %
```

```
      &pt _ ELEM #link#[2];
      tp1 _ pt;
      tp1[1] _ pt[1] + 1; % remove link delim %
      tp2 _ pt[d2sel];
      tp2[1] _ pt[d2sel+1] - 2;  % remove trailing
      delimiters %
      *sysname* _ + tp1 tp2;
   IF sysname.L = 0 THEN ABORT( 0, $"NULL name illegal");
   % search "sysname" in list %
      found _ FALSE ;
      FOR i _ 1 UP UNTIL > subslist.L DO
         IF *sysname* = *[ELEM #subslist#[i]]* THEN %
         sysname found %
            BEGIN
               found _ TRUE ;
               index _ i;
               EXIT;
            END;
   IF NOT found THEN %append "sysname" to list %
      BEGIN
         #subslist# !_ *sysname* ;
         index _ subslist.L ;
      END;
   % perform the exclusion %
      CASE ELEM #type#[1] OF
         = exclude:
            IF index = 1 THEN dismes(1, $"BASE cannot be
            excluded")
            ELSE #subslist#[index] _;
      ENDCASE;
   RETURN;
   END.
(xuodisplay) % Display parameters  x-routine %
PROCEDURE (param REF, margin);                          4B4A4
   %
      FUNCTION:
         set display parameters
      ARGUMENTS:
         param - list - command word (right/wraparound)
         margin - integer - new margin set ( mutst be
         positive )
      PROCEDURE VALUE
         FALSE if illegal margin TRUE otherwise
   %
   IF margin < 1 THEN ABORT( 0, $"illegal margin");
   % set the margin %
      CASE ELEM#param#[1] OF
         = right: udpcolmax _ margin;
         =wraparound: udpwrapcol _ margin;
      ENDCASE;
   RETURN;
   END.
(xuoentry)   % CL: ;  set entry subsystem or program  %
PROCEDURE (type REF, entname REF );                     4B4A5
   % Procedure description
      FUNCTION
```

```
            Set entry subsystem name or prgram variable
         ARGUMENTS
            type:  command word variable -- "SUBSYSTEM" (106)
            or "PROGRAM"(103)
            entname:  string variable -- name of subsystem or
            program
         RESULTS
            none
         NON-STANDARD CONTROL
            none
         GLOBALS
            none
         %
      % Declarations %
      % Set entry subsystem name or prgram variabl %
         CASE ELEM #type#[cwtype] OF
            = subsystem:  %subsystem%
               *entsubsystem* _ *entname*;
            = program:  %program%
               err($"Not implemented.");
            ENDCASE;
      % Return %
         RETURN;
      END.

   (xuoextn) % EXTERNAL name link x-routine %
   PROCEDURE (link REF);                                    4B4A6
      %
         FUNCTION:
            set external name link address
         ARGUMENTS:
            link - list - containing the link typein
         PROCEDURE VALUE
            FALSE if link too long TRUE otherwise
      %
      LOCAL pt REF;
      % check validity of the link %
         &pt _ ELEM #link#[2] + 4;
         getlink( &pt );
      IF pt.L > enlfstr.M THEN ABORT( 0, $"link too long");
      % copy to permanent location %
         *enlfstr* _ *pt*;
      RETURN;
      END.
   (xuoinit) % initialize useroption %
   PROCEDURE;                                               4B4A7
      %
         FUNCTION:
            initialize the USEROPTIONS subsys
         PROCEDURE VALUE
            FALSE if useroption file cannot be openned TRUE
            otherwise
      %
      LOCAL uojfn;
      % check accessibility of data %
         IF NOT (uojfn _ uopen(1B11 %old% ))THEN %no file
```

```
            exists%
               IF NOT (uojfn _ uopen(2B11 %new% ))THEN %will not
               be able to write%
                  ABORT(0, $"local data inaccesible");
            sysclose( uojfn, $lit );
         RETURN;
         END.
      (xuolevel) % LEVELADJUST x-routine %
      PROCEDURE (switch REF);                                    4B4A8
         %
            FUNCTION:
               set level adjust prompting
            ARGUMENTS:
               switch - command word - ON / OFF
            PROCEDURE VALUE
               TRUE always
         %
         nolevadj _ % switch %
            CASE ELEM#switch#[1] OF
               = on:  FALSE;
               = off: TRUE;
            ENDCASE FALSE;
         RETURN;
         END.
      (xuonamed) % NAME DELIMITERS x-routine %
      PROCEDURE (delleft REF, delright REF);                     4B4A9
         %
            FUNCTION:
               set new name delimiters
            ARGUMENTS:
               dellft - list(literal) - left delimiter
               delright - list(literal) - right delimiter
            PROCEDURE VALUE
               TRUE always
         %
         LOCAL pt REF;
         % set left delimiter %
            &pt _ ELEM #delleft#[2] + 4;
            dfnmdl _ IF pt.L = 0 THEN 0 ELSE *pt*[1];
         % set right delimiter %
            &pt _ ELEM#delright#[2] + 4;
            dfnmdr _ IF pt.L = 0 THEN 0 ELSE *pt*[1];
         RETURN;
         END.
      (xuooutput) % OUTPUT parameters x-routine %
      PROCEDURE (param REF, length REF);                         4B4A10
         %
            FUNCTION:
               set one of the output parameters
            ARGUMENTS:
               param - command word - which parameter
               length - REFed variable:
                  contains the number to be set for all
                  parameters but TAB. contains a list (literal)
                  text selection for tab settings
            PROCEDURE VALUE
```

```
            TRUE always
        %
        LOCAL end, da REF;
        % set the parameter %
           CASE ELEM#param#[1] OF
              = right: colmax _ &length;
              = left: tpoffset _ &length;
              = bottom: linmax _ &length;
              = page: pgsize _ &length;
              = indenting: indcnt _ &length;
              = tab:
                 BEGIN
                    tabset( ELEM#length#[2] + 4, $stdtab );
                    % update display areas %
                    end _ (&da _ $dpyarea) + dal*dacnt;
                    DO IF da.daexis THEN
                       BEGIN
                          da.datab0 _ stdtab;
                          da.datab1 _ stdtab[1];
                          da.datab2 _ stdtab[2];
                       END
                    UNTIL (&da _ &da+dal) >= end;
                 END;
           ENDCASE;
        RETURN;
        END.
    (xuoprogtemp) % PROGRAMMER'S  Template File %
    PROCEDURE (link REF);                                          4B4A11
        %
           FUNCTION:
              set programmer's template file
           ARGUMENTS:
              link - list - containing the link typein
           PROCEDURE VALUE
              FALSE if link too long TRUE otherwise
        %
        LOCAL pt REF;
        % check validity of the link %
           &pt _ ELEM #link#[2] + 4;
           getlink( &pt );
        IF pt.L > enlfstr.M THEN ABORT( 0, $"link too long");
        % copy to permanent location %
           *tmpfile* _ *pt*;
        RETURN;
        END.
    (xuoprtdir) % output DIRECTORY %
    PROCEDURE (dir REF);                                           4B4A12
        %
           FUNCTION:
              set output spooler directory
           ARGUMENTS:
              dir - list - containing the directory name
           PROCEDURE VALUE
              FALSE if no such directory
        %
        LOCAL pt1 REF, pt2 REF;
```

```
        LOCAL STRING locstr[99];
        % check validity of the directory %
           &pt1 _ ELEM #dir#[tppair];
           &pt2 _ &pt1 + d2sel;
           *locstr* _ pt1 pt2;
           transdir( $locstr ); % calls err if illegal %
        IF locstr.L > prtdir.M THEN ABORT( 0, $"directory name
        too long");
        % copy to permanent location %
           *prtdir* _ *locstr*;
        RETURN;
        END.
     (xuoquickprint) % QUICKPRINT parameters x-routine %
     PROCEDURE (param REF, length REF);                          4B4A13
        %
           FUNCTION:
              set one of the quickprint parameters
           ARGUMENTS:
              param - command word - which parameter
              length - REFed variable:
                 contains the number to be set for all
                 parameters but TAB. contains a list (literal)
                 text selection for tab settings
           PROCEDURE VALUE
              TRUE always
        %
        % set the parameter %
           CASE ELEM#param#[1] OF
              = right: uqpcolmax _ &length;
              = left: uqptpoffset _ &length;
              = bottom: uqplinmax _ &length;
              = page: uqppgsize _ &length;
              = indenting: uqpindcnt _ &length;
              = tab: tabset( ELEM#length#[2] + 4, $uqpstdtab );
           ENDCASE;
        RETURN;
        END.
     (xuoreset) % RESET x-routine %
     PROCEDURE (what REF, subcmd REF);                           4B4A14
        %
           FUNCTION:
              reset a specific variable to system default
           ARGUMENTS:
              what - command word - what parameter to reset
              subcmd - command word - subcommand applicable in
              some cases only
           PROCEDURE VALUE
              TRUE always
        %
        LOCAL command;
        command _ ELEM #subcmd#[1];
        % reset the parameter %
           CASE ELEM#what#[1] OF
              = all: uorstall();
              = backspace: bckspc _ FALSE;
              = currentcontext: tslshchars _ defcurcon;
```

```
                = display:
                   CASE command OF
                      = right: udpcolmax _ defdcolmax;
                      = wraparound: udpwrapcol _ defwrpcol;
                   ENDCASE;
                = default: uorstdflt();
                = entry:
                   CASE command OF
                      = subsystem: entsubsystem.L _ 0;
                         %no entry subsystem%
                      = program: err($"Not implemented");
                   ENDCASE;
                = external: enlfstr.L _ 0; %empty string%
                = name: (dfnmdl, dfnmdr) _ ( 0,0 )
                = cwreturn: srrsize _ defsrrsize;
                = filereturn: frrsize _ deffrrsize;
                = printoptions:
                   CASE command OF
                      = right: colmax _defcolmax;
                      = left: tpoffset _ defoffset;
                      = page: pgsize _ defpgsize;
                      = bottom: linmax _ deflinmax;
                      = indenting: indcnt _ defindcnt;
                      = tab: (stdtab, stdtab[1], stdtab[2],
                      stdtab[3]) _ (deftb1, deftb2, deftb3,
                      deftb4);
                      = directory: *prtdir* _ *dfprtdir*;
                   ENDCASE;
                = progrmrs:  % programmer's template %
                   *tmpfile* _ *dflttemp*;
                = output, = quickprint:
                   CASE command OF
                      = right: uqpcolmax _ defqcolmax;
                      = left: uqptpoffset _ defqoffset;
                      = page: uqppgsize _ defqpgsize;
                      = bottom: uqplinmax _ defqlinmax;
                      = indenting: uqpindcnt _ defqindcnt;
                      = tab: (uqpstdtab, uqpstdtab[1],
                      uqpstdtab[2], uqpstdtab[3]) _ (defqt1,
                      defqt2, defqt3, defqt4);
                      = directory: *prtdir* _ *dfprtdir*;
                   ENDCASE;
                = spaces: rjtchr.L _ rtjtab _ spftab _ 0;
                = startup: #stuplist#[1] _ ""; %empty string%
                =viewspecs:
                   CASE command OF
                      = default: (stdvsp, stdvsp[1]) _ (defvs1,
                      defvs2);
                      = prompting: novspec _ FALSE;
                   ENDCASE;
             ENDCASE;
          RETURN;
          END.
       (xuoringsize) % RETURN/FILERETURN x-routine %
       PROCEDURE (which REF, length);                    4B4A15
          %
```

```
            FUNCTION:
               set new length to the return rings
            ARGUMENTS:
               which - command word - which ring (statement
               return or file return
               length - integer - new ringsize (positive only )
            PROCEDURE VALUE
               FALSE if illegal ringsize TRUE otherwise
         %
         IF length < 1 THEN ABORT( $"Illegal ringsize" );
         % set the ringsize %
            CASE ELEM #which#[1] OF
               = cwreturn: srrsize _ length;
               = filereturn: frrsize _ length;
            ENDCASE;
         RETURN;
         END.
      (xuosget)   % CL: ;  return list of default subsystems to
      FE %
      PROCEDURE (rtnlist REF);                                    4B4A16
         % Procedure description
            FUNCTION
               This procedure is used by the Entry Subsystem
               command.  It returns a list of the defualt
               subsystems.
            ARGUMENTS
               rtnlist--REF-pointer to result list, used for
               returning results to the FE.
            RESULTS
               proc-value
            NON-STANDARD CONTROL
               none
            GLOBALS
               none
            %
         % Declarations %
            (stradr) LOCAL REF;                                   4B4A16B1
            (index) _ 1;                                          4B4A16B2
            (tp1) TEXT POINTER;                                   4B4A16B3
            (workstring) STRING [80];                             4B4A16B4
            (loclist) LIST [4];                                   4B4A16B5
         % Copy names of default subsystems to return list %
            FOR index UP UNTIL > subslist.L DO
               BEGIN  %check each subsystem for directory name%
               &stradr _ ELEM #subslist#[index];
               IF FIND SE(*stradr*) ["",] > CH $SP ^tp1 THEN
                  %subsystem name includes directory, omit it
                  from string sent to FE%
                     *workstring* _ tp1 SE(*stradr*)
               ELSE *workstring* _ *stradr*;  %no directory name%
               #loclist# !_ *workstring*;
               END;
            #rtnlist#[1] _ LIST(COPY #loclist#);
            #loclist# _ ;
         % Return %
            RETURN;
```

```
    END.

(xuoshow) % useroption SHOW x-routine %
PROCEDURE( what REF );                                          4B4A17
   %
      FUNCTION:
         show the user a specified useroption variable
      ARGUMENTS:
         what - command word - what to show
      PROCEDURE VALUE
         TRUE always
   %
   LOCAL pt REF;
   % create the message %
      &pt _ CASE ELEM#what#[1] OF
         = all: shoall();
         = currentcontext: shocurcon();
         = backspace: shobckspc();
         = default: shodflt();
         = display: shodisplay();
         = entry: shoentry();
         = external: shoextrn();
         = leveladjust: sholevadj();
         = name: shonamdel();
         = output, = quickprint: shoquick();
         = progrmrs:  shoprgrms();
         = printoptions: shooutput();
         = cwreturn: shoreturn();
         = spaces: shospaces();
         = startup: shostup();
         = viewspecs: shovsp();
      ENDCASE $"?";
   % print it out %
      typelit( 1, &pt );
   RETURN;
   END.
(xuospfortab) % SPACES for tab x-routine %
PROCEDURE (switch REF, justify REF, terminate REF);    4B4A18
   %
      FUNCTION
         set the variables associated with SPACES for tabs
      ARGUMENTS
         switch - command word -set spaces on or off
         justify - command word - if spaces on set
         justification to left or right
         terminate - literal - termination characters (up
         to 5) to terminate justification
      PROCEDURE VALUE
         FALSE if termination string is empty TRUE
         otherwise
   %
   LOCAL pt REF;
   %set the switch %
      CASE ELEM#switch#[1] OF
         = off: rjtchr.L _ rtjtab _ spftab _ 0;
         = on:
```

```
                BEGIN
                   spftab _ TRUE;
                   CASE ELEM #justify#[1] OF
                      = left:
                         BEGIN
                            rtjtab _ FALSE;
                            RETURN;
                         END;
                      = right: NULL;
                   ENDCASE % no justification specified %
                      RETURN;
                   % the user wants right justification at this
                   point %
                   &pt _ ELEM#terminate#[2] + 4;
                   IF pt.L = 0 THEN ABORT( $"NULL termination
                   characters illegal" );
                   rtjtab _ TRUE;
                   *rjtchr* _ *pt*[1 TO MIN( 5, pt.L )];
                END;
             ENDCASE;
          RETURN;
          END.
       (xuostup) % STARTUP x-routine %
       PROCEDURE (link REF);                                    4B4A19
          %
             FUNCTION:
                create a new link for startup command branch
             ARGUMENTS:
                link -literal - new branch address
             PROCEDURE VALUE
                TRUE always
          %
          LOCAL pt REF;
          % establish validity of link %
             &pt _ ELEM #link#[2] + 4;
             getlink( &pt );
          % store link permanently %
             #stuplist#[1] _ *pt*;
          RETURN;
          END.
       (xuoterm) % useroption termination %
       PROCEDURE;                                               4B4A20
          %
             FUNCTION:
                Termination rule. collect the entire data and
                write out on user's options file
             PROCEDURE VALUE
                TRUE always
          %
          LOCAL uojfn, length, frstadrs, numpage, tmp;
          LOCAL LIST uolist[3];
          LOCAL pcpport;
          % get access to file %
             IF NOT (uojfn _ uopen( 1B11 %old%)) THEN %need new%
                IF NOT (uojfn _ uopen( 2B11 %new%)) THEN
                   err($"cannot write useroption data");
```

```
        % collect the data, encode, and move it to output pages
        %
           frstadrs _ ( ($uarray/1000B) +1)*1000B;
           tmp _ makedesc(unull, 0, FALSE);
           blkxfr( $uoversion, frstadrs, $uoend - $uoversion +
           1);
           #uolist# _
              *enlfstr*,
              COPY #stuplist#[1],
              LIST( COPY #subslist#),
              *tmpfile*,
              *prtdir*,
              *entsubsystem*,
              tmp, tmp, tmp, tmp, tmp, tmp, tmp, tmp, tmp, tmp,
              tmp, tmp, tmp, tmp ;
                 % the NULL elements are provided so that
                 additional items can be added to the
                 useroptions without turning the world around.
                 list size is 20 elements %
           OPENPORT wlist(frstadrs + 1000B, 7000B : [pcpport]);
           length _ encuopt($uolist, clistt, pcpport);  % build
           the PCP88 data structure %
        % write out data to file %
           numpage _ length/1000B + 2;
           uomap(uojfn, out, numpage, frstadrs/1000B);
           uomap(uojfn, away, numpage, frstadrs/1000B);
              % Why this is necessary is not clear, but
              procedure seems to work only when it is present! %
        % turn on archive "don't delete" bit in FDB %
           chnfdb(uojfn, 17B, 1B10, 1B10);
        % cleanup lists and restore core status %
           NULL-LISTS;
           sysclose( uojfn, $lit );
           IF NOT length THEN err($"too long useroption data");
        RETURN;
        END.
     (xuoviewspecs) % VIEWSPECS x-routine %
     PROCEDURE (type REF, value REF);                    4B4A21
        %
           FUNCTION:
              set new viewspecs defaults
           ARGUMENTS:
              type - command word - viewspecs/prompting
              value -pointer - command word for prompting or
              pointer to block of viewspecs
           PROCEDURE VALUE
              TRUE always
        %
        % set the viewspecs variable %
           CASE ELEM#type#[1] OF
              = default: (stdvsp, stdvsp[1]) _ (value,
              value[1]);
              = prompting:
                 CASE ELEM#value#[1] OF
                    = on: novspec _ FALSE;
                    = off: novspec _ TRUE;
```

```
                ENDCASE;
            ENDCASE;
        RETURN;
        END.
    % support routines %
      (getlink) % get link string %
      PROCEDURE ( linkstr REF );                                    4B4B1
        %
          FUNCTION:
            check and parse a link
          ARGUMENTS:
            linkstr - address - adddress of string to be
            parsed
          PROCEDURE VALUE
            TRUE always
          RESULTS:
            return the part of linkstr that is relevant in
            linkstr itself
        %
        LOCAL adstr[40];
        LOCAL TEXT POINTER tp1,tp2;
        % parse the link %
          FIND SF( *linkstr* ) ^tp1;
          lnkprs( $tp1, $adstr );
        % store lelevant information in linkstr %
          tp1 _ adstr[ls]; tp1[1] _ adstr[ls+1];
          tp2 _ adstr[le]; tp2[1] _ adstr[le+1];
          *linkstr* _ tp1 tp2;
        RETURN;
        END.
      (shoall) % SHOW ALL %
      PROCEDURE;                                                    4B4B2
        %
          FUNCTION:
            create a string with all user options
          PROCEDURE VALUE
            address of created string
        %
        % create the string %
          % commented out until bug in compiler is fixed
          *lit* _ *[shocurcon()]*,
          *[shodisplay()]*,
          *[shoreturn()]*,
          *[sholevadj()]*,
          *[shooutput()]*,
          *[shoquick()]*,
          *[shospaces()]*,
          *[shobckspc()]*,
          *[shovsp()]*,
          *[shonamdel()]*,
          *[shostup()]*,
          *[shoextn()]*,
          *[shodflt()]*;
          %
          *lit* _ *[shocurcon()]*;
          *lit* _ *lit*, *[shodisplay()]*;
```

```
            *lit* _ *lit*, *[shoreturn()]*;
            *lit* _ *lit*, *[sholevadj()]*;
            *lit* _ *lit*, *[shooutput()]*;
            *lit* _ *lit*, *[shoquick()]*;
            *lit* _ *lit*, *[shospace()]*;
            *lit* _ *lit*, *[shobckspc()]*;
            *lit* _ *lit*, *[shovsp()]*;
            *lit* _ *lit*, *[shonamdel()]*;
            *lit* _ *lit*, *[shostup()]*;
            *lit* _ *lit*, *[shoextn()]*;
            *lit* _ *lit*, *[shodflt()]*;
            *lit* _ *lit*, *[shoprgrms()]*;
            *lit* _ *lit*, *[shoentry()]*;
         RETURN( $lit );
         END.
      (shobckspc) % SHOW BACKSPACE %
      PROCEDURE;                                                    4B4B3
         %
            FUNCTION:
               create a string to show backspace status
            PROCEDURE VALUE
               address of created string
         %
         % create the string %
            *str* _ "
            Backspace status: ", *[ IF bckspc THEN $"overprint"
            ELSE $"Echo with backspace symbol" ]*;
         RETURN( $str );
         END.
      (shocurcon) % SHOW CURRENT CONTEXT %
      PROCEDURE;                                                    4B4B4
         %
            FUNCTION:
               create a string to show current context
            PROCEDURE VALUE
               address of created string
         %
         *str* _ "
         currentcontext length = ", STRING( tslshchars );
         RETURN( $str );
         END.
      (shodflt) % SHOW DEFAULT subsys %
      PROCEDURE;                                                    4B4B5
         %
            FUNCTION:
               create a string to show default subsystems and
               programs
            PROCEDURE VALUE
               address of created string
         %
         LOCAL i;
         % set initial message %
            *str* _ "
            Default programs/subsystems: ";
         % append default  progs/subsys %
            FOR i _ 1 UP UNTIL > subslist.L DO
```

```
            *str* _ *str*, *[ELEM#subslist#[i]]*, " ; ";
      RETURN( $str );
      END.
   (shodisplay) % SHOW DISPLAY parameters %
   PROCEDURE;                                                    4B4B6
      %
         FUNCTION:
            create a string to show diplay parameters
         PROCEDURE VALUE
            address of created string
      %
      % create the string %
         *str* _ "
         display parameters:
               right margin = ", STRING(udpcolmax),  "
               wraparound margin = ", STRING(udpwrapcol);
      RETURN( $str);
      END.
   (shoentry)   % CL: ;  show entry subsystem and program %
   PROCEDURE;                                                    4B4B7
      % Procedure description
         FUNCTION
            Create a string with entry subsystem and entry
            program name.  ENTRY PROGRAM NOT YET IMPLEMENTED.
         ARGUMENTS
            none
         RESULTS
            proc-value
         NON-STANDARD CONTROL
            none
         GLOBALS
            none
         %
      % Declarations %
         (subname) STRING [40];                                  4B4B7B1
      % create the string %
         IF entsubsystem.L THEN
            *subname* _ *entsubsystem*  % entry subsystem%
         ELSE *subname* _ "BASE";  %default entry subsystem%
         *str* _ "
         Entry subsystem: ", *subname*;
      % Return %
         RETURN($str);
      END.

   (shoextn) % SHOW EXTERNAL link %
   PROCEDURE;                                                    4B4B8
      %
         FUNCTION:
            create a string to show the external name link
            address
         PROCEDURE VALUE
            address of created string
      %
      *str* _ "
      External name link: ", *[ IF enlfstr.L = 0 THEN $"None."
```

```
      ELSE $enlfstr ]*;
      RETURN( $str );
      END.
   (sholevadj) % SHOW LEVELADJUST promptimg %
   PROCEDURE;                                                    4B4B9
      %
         FUNCTION:
            create a string to show level adjust prompting
         PROCEDURE VALUE
            address of created string
      %
      *str* _ "
      Level adjust prompting: ", *[ IF nolevadj THEN $"OFF"
      ELSE $"ON" ]* ;
      RETURN( $str );
      END.
   (shonamdel) % SHOW NAME DELIMITERS %
   PROCEDURE;                                                    4B4B10
      %
         FUNCTION:
            create a string to show name delimiters
         PROCEDURE VALUE
            address of created string
      %
      LOCAL STRING str1[5], str2[5];
      % create string for each delimiter %
         IF dfnmdl=0 THEN *str1* _ "NULL" ELSE *str1* _
         dfnmdl;
         IF dfnmdr=0 THEN *str2* _ "NULL" ELSE *str2* _
         dfnmdr;
      % create entire string %
         *str* _ "
         name delimiters: left: ", *str1*, " right: ", *str2*
         ;
      RETURN( $str );
      END.
   (shooutput) % SHOW OUTPUT parameters %
   PROCEDURE;                                                    4B4B11
      %
         FUNCTION:
            create a string to show all output parameters
         PROCEDURE VALUE
            address of created string
      %
      LOCAL STRING temp[5];
      % create the string %
         *longstr* _ "
         output parameters
         left margin: ", STRING( tpoffset ), "
         right margin: ", STRING( colmax ),  "
         bottom margin: ", STRING(linmax), "
         page size: ", STRING( pgsize ), "
         indenting per level: ", STRING( indcnt ), "
         tab setting: ";
         *longstr* _ *longstr* , *[shotab( $stdtab )]*, "
         default directory: ", *prtdir*;
```

```
      RETURN ($longstr );
      END.
   (shoprgrms) % SHOW PROGRAMMER'S TEMPLATE FILE %
   PROCEDURE;                                                    4B4B12
      %
         FUNCTION:
            create a string to show the startup command branch
            address
         PROCEDURE VALUE
            address of created string
      %
      LOCAL pt REF;
      % create the string %
         *str* _ "
         programmer's template file: ", *[ IF tmpfile.L = 0
         THEN $"None." ELSE $tmpfile ]* ;
      RETURN( $str );
      END.
   (shoquick) % SHOW QUICKPRINT parameters %
   PROCEDURE;                                                    4B4B13
      %
         FUNCTION:
            create a string to show quickprint parameters
         PROCEDURE VALUE
            address of created string
      %
      LOCAL STRING temp[5];
      % create the string %
         *longstr* _ "
         quickprint parameters
         left margin: ", STRING( uqptpoffset ), "
         right margin: ", STRING( uqpcolmax ),  "
         bottom margin: ", STRING( uqplinmax ), "
         page size: ", STRING( uqppgsize ), "
         indenting per level: ", STRING( uqpindcnt ), "
         tab setting: ";
         *longstr* _ *longstr*, *[shotab( $uqpstdtab )]*, "
         default directory: ", *prtdir* ;
      RETURN ($longstr );
      END.
   (shoreturn) % SHOW RETURN ringsizes %
   PROCEDURE;                                                    4B4B14
      %
         FUNCTION:
            create a string to show return ring sizes
         PROCEDURE VALUE
            address of created string
      %
      *str* _ "
      statement return ringsize = ", STRING( srrsize ), "
      file return ringsize = ", STRING( frrsize );
      RETURN( $str );
      END.
   (shospace) % SHOW SPACES for tab %
   PROCEDURE;                                                    4B4B15
      %
```

```
      FUNCTION:
         create string to show spaces for tab status
      PROCEDURE VALUE
         address of created string
   %
   % create status message %
      *str* _ "
      Spaces for tabs: ", *[ IF spftab THEN $"ON
            justified to " ELSE $"OFF" ]*;
   % create message for justification %
      IF spftab THEN
         CASE rtjtab OF
            = TRUE: *str* _ *str*, "right terminated with:
            ", *rjtchr*;
            = FALSE: *str* _ *str*, "left";
         ENDCASE;
   RETURN( $str );
   END.
(shostup) % SHOW STARTUP branch %
PROCEDURE;                                                  4B4B16
   %
      FUNCTION:
         create a string to show the startup command branch
         address
      PROCEDURE VALUE
         address of created string
   %
   LOCAL pt REF;
   % create the string %
      &pt _ ELEM #stuplist#[1];
      *str* _ "
      startup commands address: ", *[ IF pt.L = 0 THEN
      $"None." ELSE &pt ]* ;
   RETURN( $str );
   END.
(shotab) % show tab settings %
PROCEDURE ( tabtbl REF );                                   4B4B17
   %
      FUNCTION:
         create a string to show the tab settings
      ARGUMENTS:
         tabtbl - address - address of table where tabs are
         defined
      PROCEDURE VALUE
         address of created string
   %
   LOCAL curbit, bottm, step, i, dot, tabarray[145];
   % map tab bits into tabarray. bottm is the bottom index
   of the array. Each entry in  tabarray has a single tab
   position in the right order%
      curbit _ 0; bottm _ -1;
      WHILE (curbit _ nxtbit(&tabtbl, curbit, 4 )) > 0 DO
         tabarray[bottm _ bottm + 1] _ curbit;
   % check if any tabs are defined %
      IF bottm = -1 THEN RETURN($"None.");
   % check if definition includes "..." %
```

```
      step _ IF bottm > 1 THEN tabarray[bottm] -
      tabarray[bottm-1] ELSE -1;
      dot _ % TRUE if specification includes "..." %
         bottm > 1 %no dots if less then two tabs defined %
         AND tabarray[bottm] + step > 144 %must cover
         entire range %
         AND tabarray[bottm-2] + step = tabarray[bottm-1];
         % at least two succesive intervals%
   IF dot THEN  %find virtual bottom of array before dots
   should be typed%
      WHILE bottm > 1 AND tabarray[bottm-2] + step =
      tabarray[bottm-1] DO BUMP DOWN bottm;
   %create string %
      str.L _ 0;
      FOR i _ 0 UP UNTIL > bottm DO *str* _ *str*,
      STRING(tabarray[i]), ", ";
      IF dot THEN *str* _ *str*, "..." ELSE BUMP DOWN
      str.L;
   RETURN( $str );
   END.
(shovsp) % SHOW VIEWSPECS %
PROCEDURE;                                              484B18
   %
      FUNCTION:
         create a string to show viewspecs status and
         promptin
      PROCEDURE VALUE
         address of created string
   %
   % create the string for viewspecs %
      *str* _ EOL, "Viewspecs: ";
      curvsp( $stdvsp, $str );
   % create the string for prompting %
      *str* _ *str*, EOL, "Viewspec prompting: ", *[ IF
      novspec THEN $"OFF" ELSE $"ON" ]* ;
   RETURN( $str );
   END.
(tabset) % handle tabsettings %
PROCEDURE (tabstr REF, tabl REF );                      484B19
   %
      FUNCTION:
         map the definition of tab setting into the
         corresponding bits in the tab-table
      ARGUMENTS:
         tabstr -address of string - that describes the
         tabsettings
         tabl -address - of the table where the tab bits
         should be set
      PROCEDURE VALUE
         FALSE if tabstr contains TABs TRUE otherwise
   %
   LOCAL i, form, step;
   LOCAL TEXT POINTER tp1, tp2, tp3, tp4;
   LOCAL STRING num1[10], num2[10];
   % make sure definition is legal %
      IF FIND SF(*tabstr*) [TAB] THEN
```

```
            ABORT(0,$"Do not use TABs in tab specification");
         % find which form of specification did the user use.
         form 1 sets tabs by character position. Form 2 sets tabs
         by explicit column numbers possibly followed by "..." %
            form _ IF FIND % syntax of form 2 %
               SF(*tabstr*) $( $SP 1$D $SP ", ) ( ($SP 1$D) /
               (2$".) ) $SP ENDCHR
            THEN 2 ELSE 1;
         % clear the previously define tabs %
            dassnbit( &tabl, -1, 4); %clear all tabs%
         % insert new definition %
            CASE form OF
               = 1: % form 1 %
                  FOR i _ 1 UP UNTIL > 144 DO
                     CASE READC OF
                        = ENDCHR: RETURN;
                        = SP: NULL; % keep on counting %
                     ENDCASE setbit( &tabl, i, 4);
               =2: % form 2 %
                  BEGIN
                     FIND SF(*tabstr*) ^ tp1;
                     % set the explicitly specified tabs %
                        WHILE (FIND ^tp1 $SP ^tp1 1$D ^tp2) DO %
                        set tab on position %
                           BEGIN
                              *num1* _ tp1 tp2;
                              setbit( &tabl, VALUE( $num1 ), 4);
                              IF NOT FIND ^tp2 [",] ^tp1 THEN
                              EXIT;
                           END;
                     % see if the user used "..." and set the
                     implicitly defined tabs accordingly %
                        IF FIND % two numbers followed by "..." %
                           SE(*tabstr*) $SP 2$". [",] ^tp4 1$D
                           ^tp3 [",] $SP ^tp2 1$D ^tp1 THEN
                           BEGIN % set up the rest of the tabs %
                              *num1* _ tp1 tp2;
                              *num2* _ tp3 tp4;
                              i _ VALUE( $num2 );
                              step _ i - VALUE( $num1 );
                              FOR i _ i + step UP step UNTIL >
                              144 DO
                                 setbit( &tabl, i, 4 );
                           END;
                  END;
            ENDCASE;
         RETURN;
         END.
   % userprofile routines %
      % x-routines %                                                4B5A
         (xupcontchar) % x-routine for CONTROLCHARACTER command %
         PROCEDURE ( device REF, cont REF, char REF, echostr REF);
                                                                    4B5A1
            %
               FUNCTION:
                  x-routine for CONTROLCHARACTER command.
```

```
        ARGUMENTS:
           device - command word - device for which controls
           are specified.
           cont - command word - which control function is
           being changed.
           char - literal item - characters to serve as
           controls.
           echostr - literal item - echostr for the specified
           control.
        ALGORITHM:
           1. create entrylst to contain the new entry  for
           insertion.
           2.if #entrylst# is a default definition call
           rstcontrol to reset the control character and
           return.
           3. if #cntchr# is empty create a list for the
           given device.
           4. search #cntchr# for a list of given device
           (dvcnum). call the list #dvclst#.
           5. if no list for given device exists append one
           to #cntchr#.
           6. search #dvclst# to see if this control function
           is already defined.
           7. if function is defined replace that entry with
           #entrylst# otherwise append #entrylst# to
           #dvclst#.
%
LOCAL dvclst REF, sstr REF, function, dvcnum, index;
LOCAL TEXT POINTER estr;
LOCAL STRING str1[20], str2[20];
LOCAL LIST entrylst[3];
REF cntchr;

dvcnum _ ELEM #device#[1] ;
function _ ELEM #cont#[1] ;
% create the entry for the given device and controlchar
   NOW EXTRACTING CHAR STRING AS BELOW
   #entrylst# _ USE makedesc(uindex,function, FALSE),
   MOVE#char#[2], MOVE#echostr#[2];
%
% extract char and echostr character strings %
   &sstr _ ELEM #char#[tppair];
   estr _ sstr[2];  estr[1] _ sstr[3];
   *str1* _ sstr estr;
   &sstr _ ELEM #echostr#[tppair];
   estr _ sstr[2];  estr[1] _ sstr[3];
   *str2* _ sstr estr;
% create the entry for the given device and controlchar
%
   #entrylst# _ USE makedesc(uindex,function, FALSE),
   *str1*, *str2*;
IF (ELEM #entrylst#[2]).L = 0 THEN ABORT( 0, $"NULL
character illegal" );
IF dflt( $entrylst ) THEN % reset the entry %
   BEGIN
      rstcontrol( dvcnum, function );
```

```
            RETURN;
         END;
      IF lreadb( &cntchr , 1) = unull THEN % list is empty -
      create one%
         BEGIN
            #cntchr#[1] _ LIST( USE makedesc(uindex, dvcnum,
            FALSE), LIST( MOVE #entrylst# ) ) ;
            RETURN;
         END;
      IF NOT (index _ search( &cntchr, 1, dvcnum)) THEN
         BEGIN % create a list for the given device %
            #cntchr# !_ LIST( USE makedesc(uindex, dvcnum,
            FALSE), LIST( MOVE #entrylst# ) ) ;
            RETURN;
         END;
      &dvclst _ ELEM#cntchr#[index];
      IF (index _ search( &dvclst, 2, function)) THEN %replace
      the definition%
         #dvclst#[index] _ LIST( MOVE #entrylst# )
      ELSE % create new entry %
         #dvclst# !_ LIST ( MOVE #entrylst# ) ;
      NULL-LISTS;
      RETURN;
      END.

   (xupnoisewrd) % x-routine for NOISEWORD command %
   PROCEDURE (type REF, length, rtnlst REF);                    4B5A2
      %
         FUNCTION:
            x-routine for FEEDBACK command
         ARGUMENTS:
            type - command word - what parameter to set
            length - integer - length of noisewords-length
      %
      CASE ELEM#type#[1] OF
         = verbose: profil.nsmd _ upverbose;
         = terse: profil.nsmd _ upterse;
         = lngth:
            IF length IN (0,256) THEN profil.nslen _ length
            ELSE ABORT( 0, $"illegal noiseword length" );
      ENDCASE;
      makertn( &rtnlst, profil.nsmd, profil.nslen );
      RETURN( &rtnlst );
      END.

   (xupcmndwrd) % x-routine for COMMAND-WORD command %
   PROCEDURE (type REF, length, rtnlst REF);                    4B5A3
      %
         FUNCTION:
            x-routine for COMMAND-WORD command
         ARGUMENTS:
            type - command word - what parameter to set
            length - integer - length of command words
      %
      CASE ELEM#type#[1] OF
         = verbose: profil.cmdmd _ upverbose;
```

```
            = terse: profil.cmdmd _ upterse;
            = lngth:
               IF length IN (0,15) THEN profil.cmdlen _ length
               ELSE ABORT( 0, $"illegal command-words length" );
         ENDCASE;
         makertn( &rtnlst, profil.cmdmd, profil.cmdlen );
         RETURN( &rtnlst );
         END.

      (xupherald) % x-routine for HERALD command %
      PROCEDURE (type REF, length, rtnlst REF);                    4B5A4
         %
            FUNCTION:
               x-routine for HERALD command
            ARGUMENTS:
               type - command word - what herald parameter to set
               length - integer -  length for herald-length
               command (applicabale only in this case)
         %
         CASE ELEM #type#[1] OF
            = verbose: profil.hldmd _ upverbose;
            = terse: profil.hldmd _ upterse;
            = lngth:
               IF length IN (0,32) THEN profil.hldlen _ length
               ELSE ABORT( 0, $"illegal herald length");
         ENDCASE;
         makertn( &rtnlst, profil.hldmd , profil.hldlen );
         RETURN;
         END.

      (xupinit) % Userprofile initialization %
      PROCEDURE;                                                   4B5A5
         %
            FUNCTION:
               tool initialization rule
         %
         LOCAL numpage, fpage;
         % initialize parameters %
            fpage _ $uarray/1000B+1;
            numpage _ $uarend/1000B - fpage;
         upinit(fpage, numpage);
         RETURN;
         END.

      (xupprompt) % x-routine for PROMPT command %
      PROCEDURE (type REF, rtnlst REF);                            4B5A6
         %
            FUNCTION:
               x-routine for PROMPT command
            ARGUMENTS:
               type - command word - new prompt setting.
         %
         profil.prpt _
            CASE ELEM #type#[1] OF
               = full: upfull;
               = partial: uppartial;
```

```
            = off: upoff;
            = conditional: upconditional;
         ENDCASE upconditional;
      makertn( &rtnlst, profil.prpt, 0 );
      RETURN( &rtnlst );
      END.

   (xuprecognition) % x-routine for the RECOGNITION command %
   PROCEDURE (first REF, second REF, rtnlst REF);          4B5A7
      %
         FUNCTION:
            x-routine for the RECOGNITION command
         ARGUMENTS:
            first - command word - first level recognition
            mode
            second - command word - second level recognition
            mode. Applicable only if first is "TERSE".
      %
      LOCAL tmp;
      CASE ELEM #first#[1] OF
         = terse:
            BEGIN % second level not optional %
            profil.rcg2 _ rcg2dflt;
            tmp _ upterse;
            END;
         = anticipatory: tmp _ upanticipatory;
         = fixed: tmp _ upfixed;
         = demand: tmp _ updemand;
      ENDCASE;
      profil.rcg _ tmp;
      makertn( &rtnlst, tmp, profil.rcg2 );
      RETURN( &rtnlst );
      END.

   (xupreset) % x-routine for RESET command %
   PROCEDURE (what REF, subcmd REF, function REF);         4B5A8
      %
         FUNCTION:
            x-routine for RESET command
         ARGUMENTS:
            what - command word - what to reset.
            subcmd - command word -subcommand needed for some
            commands like:
               RESET FEEDBACK LENGTH or device for RESET
               CONTROL
            function - command word - control function to
            reset
      %
      LOCAL i, dvce;
      CASE ELEM #what#[1] OF
         = noisewrd:
            CASE ELEM #subcmd#[1] OF
               = mode: profil.nsmd _ nsmddflt;
               = lngth: profil.nslen _ nslndflt;
            ENDCASE;
         = cmndwrd:
```

```
            CASE ELEM #subcmd#[1] OF
               = mode: profil.cmdmd _ cmdmddflt;
               = lngth: profil.cmdlen _ cmdlndflt;
            ENDCASE;
         = herald:
            CASE ELEM #subcmd#[1] OF
               = mode: profil.hldmd _ hldmddflt;
               = lngth: profil.hldlen _ hldlndflt;
            ENDCASE;
         = prompt: profil.prpt _ prptdflt;
         = recognition: (profil.rcg , profil.rcg2 ) _ (rcgdflt
         , rcg2dflt);
         = startup:  #uprofil#[2] _ "";
         = control: rstcontrol( ELEM #subcmd#[1], ELEM
         #function#[1] );
         = all: uprsetall();
      ENDCASE;
      RETURN;
      END.

   (xupshow) % x-routine for SHOW command %
   PROCEDURE (what REF, dvce REF);                              4B5A9
      %
         FUNCTION:
            x-routine for SHOW command
         ARGUMENTS:
            what - command word - what to show
            dvce - command word - device code for SHOW CONTROL
      %
      CASE ELEM #what#[1] OF
         = control: *msg* _ *[cntrlmsg( ELEM #dvce#[1] )]*;
         = noisewrd: *msg* _ *[nswdmsg()]*;
         = cmndwrd: *msg* _ *[cmndmsg()]*;
         = herald: *msg* _ *[hrldmsg()]*;
         = prompt: *msg* _ *[prptmsg()]*;
         = recognition: *msg* _ *[rcgmsg()]*;
         = all: *msg* _ *[hrldmsg()]* , *[nswdmsg()]* ,
         *[cmndmsg()]* , *[prptmsg()]* , *[rcgmsg()]* ;
      ENDCASE;
      typelit(1, $msg );
      RETURN;
      END.

   (xupterm) % Userprofile termination %
   PROCEDURE;                                                   4B5A10
      %
         FUNCTION:
            Termination rule. collect the entire data and
            write out on Userprofile file
         PROCEDURE VALUE
            TRUE always
      %
      LOCAL upjfn, length, frstadrs, numpage;
      LOCAL pcpport;
      % get access to file %
         IF NOT (upjfn _ upopen( 1B11 %old%)) THEN %need a new
```

```
            one%
               IF NOT (upjfn _ upopen( 2B11 %new%)) THEN
                  err($"cannot write userprofile data");
         % collect the data, encode, and move it to output pages
         %
            frstadrs _ ($uarray/1000B+1)*1000B;
            numpage _ $uarend/1000B - frstadrs/1000B;
            OPENPORT wlist(frstadrs, numpage*7000B : [pcpport]);
            length _ encuopt($uprofil, clistt, pcpport);  % build
            the PCPB8 data structure %
         % write out data to file %
            numpage _ length/1000B + 1;
            uomap(upjfn, out, numpage, frstadrs/1000B);
            uomap(upjfn, away, numpage, frstadrs/1000B);
               % Why this is necessary is not clear, but
               procedure seems to work only when it is present! %
         % turn on archive "don't delete" bit in FDB %
            chnfdb(upjfn, 17B, 1B10, 1B10);
         % cleanup lists and restore core status %
            #uprofil# _;
            sysclose( upjfn, $lit );
            IF NOT length THEN err($"too long useroption data");
         RETURN;
         END.
      % support routines %
       (cntrlmsg) % SHOW CONTROLCHARACTERS  %
       PROCEDURE (dvcnum);                                    485B1
         %
            FUNCTION:
               create a message to be shown at SHOW  CONTROL
               command
            ARGUMENTS:
               dvcnum - integer - device number code
            ALGORITHM:
               1. three lists are created for printout:
                  (a) funclist: contains the names of the
                  functions                                 4B5B1A3A1
                  (b) chrlst: contains the control characters.
                                                            4B5B1A3A2
                  (c) echolst: contains the echoes for the
                  control functions.                        4B5B1A3A3
                  The order in those lists is important!!!!
               2. search #cntchr# to find if definition for
               device exists.
               3. if definition exists replace entries in
               #chrlst# and #echolst# with user definitions.
               4. print out the lists defined in (1) taking one
               entry at a time from each.
         %
         LOCAL dvclst REF, i, cntentry REF, index, found;
         LOCAL LIST funclst[15], chrlst[15], echolst[15];
         REF cntchr;
         % create lists for printout filled with defaults %
         #funclst# _ "CA" , "CD" , "RPT" , "BC" , "BW" , "BS" ,
         "LITESC" , "IGNORE" , "SC" , "SW" , "TAB" , "OPTION" ;
         #chrlst# _ "<CA>" , "<CD>" , "<^B>" , "<BC>" , "<^W>" ,
```

```
        "<^/>" , "<^V>" , "-" , "-" , "-" , "    " , "<^U>" ;
        #echolst# _ "" , "" , "", "" , "" , "" , "" , "" , "" ,
        "" , "" , "" , "" , "" , "" ;
           % The echo  defaults are initially NULL strings %
        IF (index _ search( &cntchr, 1, dvcnum )) THEN
           BEGIN % replace user definitions in printout lists %
              &dvclst _ ELEM #cntchr#[index];
              FOR i _ 2 UP UNTIL > dvclst.L DO
                 BEGIN
                    &cntentry _ ELEM #dvclst#[i];
                    #chrlst#[ ELEM#cntentry#[1]] _ *[ELEM
                    #cntentry#[2]]* ;
                    #echolst#[ ELEM#cntentry#[1]] _
                    *[ELEM#cntentry#[3]]* ;
                 END;
           END;
        *msg* _ EOL , "func      char(s) echo" ;
        FOR i _ 1 UP UNTIL > chrlst.L DO
           *msg* _ *msg* , EOL , *[ELEM #funclst#[i]]* , TAB ,
           *[ELEM #chrlst#[i]]* , TAB , *[ELEM #echolst#[i]]* ;
        NULL-LISTS;
        RETURN ( $msg );
        END.

     (dflt) % checks for default CONTROLCHARS %
     PROCEDURE (entrylst REF);                                  4B5B2
        %
           FUNCTION:
              Determines whether the user typed in default
              definition fo specific control character
              Returns TRUE if #entrylst# equals default
           ARGUMENTS:
              entrylst - address of a list (see definition of
              the contents of entrylst in procedure xupcontchar)
        %
        LOCAL result, character, pt REF;
        % all defaults controls have less than character %
           IF [ELEM #entrylst#[2]].L >1 THEN RETURN(FALSE);
        % all default echo is NULL %
           IF [ELEM #entrylst#[3]].L # 0 THEN RETURN(FALSE);
        result _ FALSE;
        &pt _ ELEM #entrylst#[2] ;
        character _ IF pt.L = 0 THEN 0 ELSE *pt*[1] ;
        CASE ELEM #entrylst#[1] OF
           = ca: result _ (character = "<CA>) ;
           = cd: result _ (character = "<CD>) ;
           = rpt: result _ (character = "<^B>) ;
           = bc: result _ (character = "<BC>) ;
           = bw: result _ (character = "<^W>) ;
           = bs: result _ (character = 20B) % ^P % ;
           = litesc: result _ (character = "<^V>) ;
           = ignore, =sc, =sw: result _ (character = 0) % NULL %
           ;
           = tab: result _ (character = TAB) ;
           = option: result _ (character = "<^U>) ;
        ENDCASE;
```

```
   RETURN( result);
   END.

(nswdmsg) % SHOW NOISEWORD %
PROCEDURE;                                                    4B5B3
   %
      FUNCTION:
         create a message for SHOW NOISEWORD command
   %
   LOCAL tmp REF;
   CASE profil.nsmd OF
      = upverbose: &tmp _ $"verbose ";
      = upterse: &tmp _ $"terse ";
   ENDCASE;
   *str* _ EOL , "Noise-word mode: " , *tmp* , "(length = "
   , STRING(profil.nslen) , ") ;
   RETURN( $str );
   END.

(cmndmsg) % SHOW COMMANDWORD %
PROCEDURE;                                                    4B584
   %
      FUNCTION:
         create a message for SHOW COMMANDWORD command
   %
   LOCAL tmp REF;
   CASE profil.cmdmd OF
      = upverbose: &tmp _ $"verbose ";
      = upterse: &tmp _ $"terse ";
   ENDCASE;
   *str* _ EOL , "Command-word mode: " , *tmp* , "(length =
   " , STRING(profil.cmdlen) , ") ;
   RETURN( $str );
   END.

(hrldmsg) % SHOW HERALD %
PROCEDURE;                                                    4B5B5
   %
      FUNCTION:
         create a message for SHOW HERALD command
   %
   LOCAL tmp REF;
   CASE profil.hldmd OF
      = upverbose: &tmp _ $"verbose ";
      = upterse: &tmp _ $"terse ";
   ENDCASE;
   *str* _ EOL , "herald mode: " , *tmp* , " (length = " ,
   STRING( profil.hldlen ), ") ;
   RETURN( $str );
   END.

(prptmsg) % SHOW PROMPT %
PROCEDURE;                                                    4B5B6
   %
      FUNCTION:
         create a message for SHOW PROMPT command
```

```
        %
        LOCAL tmp REF;
        CASE profil.prpt OF
           = upfull: &tmp _ $"full";
           = uppartial: &tmp _ $"partial";
           = upoff: &tmp _ $"off";
           = upconditional: &tmp _ $"conditional";
        ENDCASE;
        *str* _ EOL , "prompt: ", *tmp*;
        RETURN( $str );
        END.

    (rcgmsg) % SHOW RECOGNITION %
    PROCEDURE;                                                  4B5B7
        %
           FUNCTION:
              create a message for SHOW RECOGNITION command
        %
        LOCAL tmp REF;
        CASE profil.rcg OF
           = upanticipatory: &tmp _ $"anticipatory";
           = upterse: &tmp _ $"terse";
           = upfixed: &tmp _ $"fixed";
           = updemand: &tmp _ $"demand";
        ENDCASE;
        *str* _ EOL , "recognition mode: ", *tmp*;
        CASE profil.rcg2 OF
           = upanticipatory: &tmp _ $"anticipatory";
           = upterse: &tmp _ $"terse";
           = upfixed: &tmp _ $"fixed";
           = updemand: &tmp _ $"demand";
        ENDCASE;
        IF profil.rcg = upterse THEN *str* _ *str* , EOL,"
        secondary mode: ", *tmp* ;
        RETURN( $str );
        END.

    (rstcontrol) % RESET CONTROL CHARACTER %
    PROCEDURE (device, function);
        %
           FUNCTION:                                          4B5B8A1
              Resets the appropriate entrynum in the control
              character list                                 4B5B8A1A
           ARGUMENTS:                                         4B5B8A2
              device - integer - device number               4B5B8A2A
              function - integer - function index            4B5B8A2B
        %
        LOCAL index, entrynum, dvclst REF;                     4B5B8C
        REF cntchr;                                            4B5B8D
        CASE device OF                                         4B5B8E
           = all:
              BEGIN                                          4B5B8E1A
                 #uprofil#[4] _ LIST( NULL );
                 &cntchr _ ELEM #uprofil#[4];               4B5B8E1A2
                 RETURN;                                    4B5B8E1A3
              END;                                           4B5B8E1B
```

```
            ENDCASE;                                                    4B5B8F
            IF NOT (index _ search( &cntchr, 1, device )) THEN % no
            need to reset % RETURN;                                     4B5B8G
            % process list for specified device %
            CASE function OF                                            4B5B8I
               = all: #cntchr#[index] _; % delete entire definition
               for device %
            ENDCASE                                                     4B5B8J
               BEGIN                                                    4B5B8J1
                  &dvclst _ ELEM #cntchr#[index];                       4B5B8J1A
                  IF (entrynum _ search( $dvclst, 2, function ))
                  THEN                                                  4B5B8J1B
                     % delete the entry % #dvclst#[entrynum] _;
                  IF dvclst.L = 1 THEN % all entries for this device
                  default %                                             4B5B8J1C
                     #cntchr#[index] _;
               END;                                                     4B5B8J2
            IF cntchr.L = 0 THEN % all devices default %                 4B5B8K
               BEGIN                                                    4B5B8K1
                  #uprofil#[4] _ LIST( NULL );
                  &cntchr _ ELEM #uprofil#[4];                          4B5B8K1B
               END;                                                     4B5B8K2
            RETURN;                                                     4B5B8L
            END.
                                                                        4B5B8M
         (search) % searches a list %
         PROCEDURE (list REF, i0, item );
            %
               FUNCTION:                                                4B5B9A1
                  searches a list from the i0-th element and on for
                  a specific "item"                                     4B5B9A1A
                  The list is made up of sublists and each sublist
                  has its first element an index type element. The
                  search attempts to match this index with "item"
                                                                        4B5B9A1B
               ARGUMENTS:                                               4B5B9A2
                  list - list - the list to be searched                 4B5B9A2A
                  i0 - integer -  the starting element;                 4B5B9A2B
                  item - integer - the item to search for               4B5B9A2C
               RESULTS                                                  4B5B9A3
                  returns the index into the list if a match is
                  found and 0 otherwise                                 4B5B9A3A
            %
            FOR i0 UP UNTIL > list.L DO                                 4B5B9C
               IF ELEM #[ELEM #list#[i0]]#[1] = item THEN RETURN( i0
               );                                                       4B5B9C1
            RETURN( FALSE );                                            4B5B9D
            END.
                                                                        4B5B9E
   % common routines %
      % x-routines %                                                    4B6A
         (xureset)   % reset both uo and up %
         PROCEDURE;                                                     4B6A1
            % Procedure description
               FUNCTION
                  Reset both useroptions and userprofile
```

```
            ARGUMENTS
               none
            RESULTS
               none-value
            NON-STANDARD CONTROL
               none
            GLOBALS
               none
            %
         uorstall();
         uprsetall();
         % Return %
            RETURN;
         END.

      (xushow)   % show both uo and up %
      PROCEDURE;                                                    4B6A2
         % Procedure description
            FUNCTION
               Show both useroptions and userprofile
            ARGUMENTS
               none
            RESULTS
               none-value
            NON-STANDARD CONTROL
               none
            GLOBALS
               none
            %
         LOCAL strptr REF;
         % make th message %
            &strptr _ shoall();
            *strptr* _ *strptr*, *[hrldmsg()]* , *[nswdmsg()]* ,
            *[cmndmsg()]* , *[prptmsg()]* , *[rcgmsg()]*, EOL ;
         % display it %
            typelit(1, &strptr);
         % Return %
            RETURN;
         END.

   % support routines %
      (encuopt)   % LB: ;  encode data structure in PCPB-8 format
      %
      PROCEDURE (nlsds REF, type, pcpport % => length %);       4B6B1
         % Procedure description
            FUNCTION
               convert one L10 entitity into the corresponding
               PCPB-8 data structure using WLIST
            ARGUMENTS
               nlsds: value(or address dpending on type) of a L10
               data structure to encode
               type:  pcp type
               pcpport:  WLIST port
            RESULTS
               length:  the length in bytes of the new PCPB8 data
               structure.
```

```
            NON-STANDARD CONTROL
               ABORT if bad PCP type - error code ermptype
            GLOBALS
               none
            %
         % Declarations %
            LOCAL ptr, i, length;
         %encode structure type%
            CASE type OF
               = pcpempty, = pcpboolean, = pcpindex, =
               pcpcharstr:
                  length _ PCALL [pcpport] (type, &nlsds);
               = pcpbitst:
                  % user is responsible to convert the bitstr to
                  32 bit/wrd using the procedure cnv36to32 %
                  length _ PCALL [pcpport] (type, &nlsds);
               = pcpinteger:
                  BEGIN
                  ptr _ &nlsds;  %WLIST needs address of integer%
                  length _ PCALL [pcpport] (type, $ptr);
                  END;
               %  there is not currently a PCPB-8 block type
               = pcpblock:
               %
               = pcplist:
                  BEGIN
                  length _ PCALL [pcpport] (type, nlsds.L);  %set
                  up list%
                  FOR i _ 1 UP 1 UNTIL > nlsds.L DO  %convert
                  elements%
                     length _ encuopt(ELEM #nlsds#[i], utoptype(
                     DESCR #nlsds#[i] ), pcpport);
                  length _ PCALL [pcpport] (wlistend);  %close
                  list%
                  END;
               ENDCASE ABORT(ermptype, $"Illegal PCP data type",
               type);
         % Return %
            RETURN(length);
         END.

      (makertn)   % create profile return list %
      PROCEDURE ( rtnlst REF, val1, val2 );                    486B2
         % Procedure description
            FUNCTION
               create a list with profile return value
            ARGUMENTS
               rtnlst -- address of eturn list
               val1, val2 - return values
            RESULTS
               none
            NON-STANDARD CONTROL
               none
            GLOBALS
               none
            %
```

```
        #rtnlst# _ USE makedesc(uinteg, val1, FALSE), USE
        makedesc(uinteg, val2, FALSE);
        RETURN;
        END.

FINISH of useroptions
   %%                                                    4E7A
```