```
        SHUTDOWN ..
        PROGEND .. END ,.
    /*
    //SCANCOMP JOB 99999211,WCD,MSGLEVEL=(1,1)
    //C EXEC LSPUPGEN,DSSP=OLD,SSP=20,MEM=ICISCAN
    //BAKUP.SYSUT1 DD *
            /* SCAN PROGRAM V1 */ (SUBRG,STRG) ..
            /* ACCEPTS INPUT STRAIGHT FROM ICI, OR THRU ICIFIX.          */
            /* OUTPUT TO STRUCTUR PROGRAM  IS GUARANTED TO START WITH A  */
            /* BLANK-ASTERISK FOLLOWED BY A SMALL IETTER, AND TO END     */
            /* WITH AT LEAST 15 BLANKS.                                  */
        SCAN .. PROC OPTIONS(MAIN) ,.
            DCL
            (ITP,IXL,IXR,K) FIXED BIN(31,0),   /* SCRATCH VARIABLES.     */
            SMOD CHAR(27) VAR,        /* LETTERS OF SECTYPES WANTED.     */
            SECS CHAR(27) VAR,        /* LEGITIMATE SECTION TYPE CHARS.  */
            SERRI(0 .. 1) CHAR(3) STATIC INIT(' E=',' M='),
                 /* FOR ERR/NOERR SECTION DUMP HEADER MSG SWITCH.        */
            ILLEGAL STATIC INIT(0),  /* COUNTER FOR ILLEGAL CHAR STATS.  */
            IERRCNT FIXED BIN(31,0) STATIC INIT(0),      /* STATISTIC -  */
                 /* TOTAL OF ILLEGAL DELIM ERROR MESSAGES.               */
            NOPT INIT(1) STATIC FIXED BIN(31,0),         /* NEXT LINK IN */
                                                         /* OPTION CHAIN */
            CORRBIT STATIC INIT(1)   /* CORRECTION TEXT SKIP SWITCH.     */
            FIXED BIN(15,0),
            NOUT STATIC INIT(0) FIXED BIN(31,0),         /* TOTAL NO. OF */
                                                         /* OUTPUT CHARS */
            CAPCHK CHAR(80),          /* FOR COUNTING UPPERCASE CHARS.   */
                                      /* CAPDEF IS DEFINED ON CAPCHK.    */
            CAPDEF(80) CHAR(1) DEF CAPCHK,     /* FOR COUNTING UPPERCASE */
            NCAPS STATIC FIXED BIN(31,0) INIT(0),        /* STATISTICS - */
                 /* NUMBER OF UPPERCASE CHARS IN DATA, ACCUMULATOR.      */
            NCHARS STATIC FIXED BIN(31,0) INIT(0),       /* STATISTICS - */
                 /* NUMBER OF CHARACTERS OF LOGICAL DATA INPUT, EXCLUDIN */
                 /* TYPIST ENTRIES, AND INCLUDING EXTRA BLANKS.  EXCEPT  */
                 /* FOR EXTRA BLANK LINES.                               */
            M INIT(0) STATIC          /* STRUCTURED OUTPUT RECORD COUNTER */
            FIXED BIN(31,0),          /* FOR CROSS-REFERENCE.            */
            INDXS STATIC INIT(0) FIXED BIN(31,0),        /* STATISTICS - */
                                      /* NUMBER OF INDEX ENTRIES.        */
            TP CHAR(1),               /* CURRENT TYPIST'S LETTER.        */
            CD CHAR(1),               /* TEMP.                           */
            PBIT BIT(1),              /* PUNCTUATION BIT.                */
            QBIT BIT(1),              /* QUOTE BIT (SINGLE OR DOUBLE).   */
            A CHAR(80  ) VAR,         /* INPUT BUFFER.                   */
            B CHAR(3625) VAR,         /* WORK BUFFER.                    */
            F CHAR(80 ) VAR,          /* TEMP.                           */
            STR CHAR(3625) VAR INIT(' '),      /* STRUCTURED OUTPUT BUFR */
            TR CHAR(080) VAR,         /* TRANSLATE BUFFER.               */
            C CHAR(140) VAR,          /* INDEX ENTRY BUFFER.             */
            CC CHAR(1),               /* TEMP FOR DELIMITER CHARACTERS.  */
            DTMP CHAR(2),             /* BASE FOR CTMP AND ETMP OVERLAYS. */
            CTMP CHAR(1) DEF DTMP,    /* TEMP.                           */
            ETMP CHAR(1) DEF DTMP POS(2),      /* TEMP.                  */
            POMAN CHAR(4) VAR,        /* POTENTIAL ROMAN PAGE NUMBER.    */
            ROMAN CHAR(4) VAR,        /* ACTUAL CURRENT ROMAN PAGE NO.   */
            DAY CHAR(2),              /* DAY STRING.                     */
            MNTHDAY CHAR(2),          /* MONTH STRING.                   */
            YRMNDAY CHAR(4),          /* YEAR.                           */
            TDATE CHAR(10),           /* TEMP FOR POTENTIAL DATE AREA,   */
                                      /* TDATA IS DEFINED ON TDATE.      */
            TDATA(10) CHAR(1) DEF TDATE,       /* ACCESS TO TDATE CHARS. */
            DATEMP CHAR(10) VAR,      /* TEMP FOR POTENTIAL DATE STRING. */
            ICE STATIC INIT(0),       /* CE IS DEFINED ON ICE, POS(2).   */
            CE CHAR(1) DEF ICE POS(2),         /* FOR FILLING ICE.       */
            CP CHAR(1) DEF ICP POS(2),         /* FOR PAGE NO. CONVERT   */
```

```
        ICP INIT(0),                    /* TEMP USED IN PAGECHK, CE, ABCVE, */
                                        /* IS DEFINED ON IT BOS(2).         */
        LTT INIT(0),                    /* PTR TO CURRENT RIGHT DELIMITER.  */
        LOS INIT(0),                    /* OLD LAST SCANNED CHAR.           */
        NC INIT(0),                     /* PTR TO LAST CHAR SCANNED IN TR   */
        IX INIT(1),                     /* INDEX TO START OF TR'S LOGICAL   */
                                        /* OVERLAY ON THE B BUFFER.         */
        IEND INIT(1),                   /* 1 MEANS NO END OF FILE YET.      */
        NEWPAGE INIT(1),                /* CURRENT PAGE NUMBER.             */
        J INIT(0) FIXED BIN(31,0),              /* INPUT LINE COUNTER.      */
        ICO,                            /* PREVIOUS DELIMITER CODE.         */
        IC INIT(0),                     /* INIT(0) (NE 1) AVOIDS STRG ERR.  */
        SYSPRINT PRINT ENV(F(133)),
        CLIN STREAM INPUT,              /* CLEAN INPUT FILE.                */
        CLOUT STREAM OUTPUT,            /* CLEANED OUTPUT FILE.             */
        MODE INIT(0),                   /* CURRENT MODE                     */
        MODOLD INIT(0) ,.               /* PREVIOUS MODE                    */
        DCL             /* DCL STMT FOR CONSTANTS.                          */
        CAPS CHAR(26) STATIC     INIT('ABCDEFGHIJKLMNOPQRSTUVWXYZ'),
        SMALLS CHAR(26) INIT('abcdefghijklmnopqrstuvwxyz') STATIC,
        LEGITS CHAR(87) STATIC VAR INIT(
        '0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz
+*/@#$&!()?-=[ ]: ;'',.%|"'),     /* LEGITIMATE CHARS FROM ICI.           */
        IGPCNT STATIC INIT(0) FIXED BIN(31,0),          /* LINE NO. ON  */
                                                        /* CURRENT PAGE */
        NPAGE STATIC INIT(0) FIXED BIN(31,0),           /* OLD PAGE NO. */
        STAR CHAR(1) STATIC INIT('*'),      /* ASTERISK CHAR CONSTANT       */
        ERRBITS(10) BIT(09) STATIC UNALIGNED INIT( '110111111'B,/* -5*/
        '110111111'B, '110111111'B, '110111111'B, '110111111'B,
        /* INDEX -4       INDEX -3       INDEX -2       INDEX -1     */
        '001001011'B, '101000000'B, '101000000'B, '101000000'B,
        /* STRCTRD        L. PAREN       L. BRAKT       D. BRAKT     */
        '001000000'B),                  /* ERRAY IS DEFINED ON ERRBITS.     */
        /* COMMENT          */
        ERRAY(-5 .. 4,9) BIT(1) DEF ERRBITS,
        DLM(9) CHAR(1) STATIC INIT(
        '*','+','=','$','(',')','[',']','%'),
        NEWMODE(-5 .. 4,9) STATIC INIT(     /*
         *  +  =  $  (  )  [  ]  %      DELIMITERS                          */
         0,-5, 0,-5, 1, 0, 2, 0, 0,         /* INDEX IN STRCTRD TEXT        */
         0,-4, 4,-4,-4,-4,-4,-4,-4,         /* INDEX IN COMMENT SEC.        */
         0,-3, 3,-3,-3,-3,-3,-3, 0,         /* INDEX DBL LEFT BRAKET        */
         0,-2, 2,-2,-2,-2,-2, 0, 0,         /* INDEX SNGL LEFT BRAKET       */
         0,-1, 1,-1,-1, 0,-1,-1,-1,         /* INDEX AFTER LEFT PAREN       */
         0,-5, 0,-5, 1, 0, 2, 0, 0,         /* STRUCTURED TEXT MODE         */
         0,-1, 1,-1, 1, 0, 1, 1, 1,         /* LEFT PAREN EXTRAN TEXT       */
         0,-2, 2,-2, 2, 2, 3, 0, 0,         /* LEFT BRAKT EXTRAN TEXT       */
         0,-3, 3,-3, 3, 3, 3, 3, 0,         /* DBL LEFT BRAKT EXTRAN        */
         0,-4, 4,-4, 4, 4, 4, 4, 4),        /* COMMENT SEC EXTRAN TXT       */
        NEWFRK(-5 .. 4,9),              /* MODE/DELIM FREQUENCY TABLE.      */
        OPAR BIT(16),                           /* ICPT DEFINED ON OPAR.    */
        IOPT(16) BIT(1) DEF OPAR ,.             /* OPTION BITS.             */
        ON ERROR SNAP BEGIN ,.          /* GENERAL PURPOSE ERROR HANDLER.   */
           INCODE=ONCODE ,.             /* SAVE UNDISTORTED ONCODE.         */
           PUT FILE(SYSPRINT) DATA ,.           /* DUMP FOR DEBUGGING.      */
           END ,.
        ON ENDFILE(IN) BEGIN ,.  /* FOR EITHER REAL OR SIGNALED END.        */
           A=STAR ,.                    /* INDICATE END OF SECTION.         */
           IEND=0 ,.                    /* INDICATE NO NEXT SECTION.        */
           GOTO DBLATSRCH ,.    END ,.          /* CONTINUE FOR CLEANUP.    */
        ON STRG SNAP PUT FILE(SYSPRINT) DATA ,.         /* FOR DEBUG.       */
        OPEN FILE(SYSPRINT) PRINT LINESIZE(132) PAGESIZE(62) ,.
        NEWFRK=0 ,.                     /* INITIALIZE FREQUENCY TABLE.      */
        GOTO ENTER ,.                   /* ENTRY TO MAIN LOOP.              */
    RESRCH ..                           /* TOP OF MAIN LOOP.                */
        IF NC=LENGTH(TR)                /* IF LAST CHAR IN BUFF IS A DLM    */
```

```
      THEN GOTO MORE ,.          /* THAT HAS ALREADY BEEN FOUND THEN */
      ND=INDEX(SUBSTR(TR,NC+1),STAR ) ,. /* ELSE SRCH FOR NEXT DLM */
      IF ND=0 THEN                       /* IF NONE LEFT IN BUFR   */
   MORE .. DO ,.                         /* THEN GET MORE INPUT.   */
      B=SUBSTR(B,LTT) ,.         /* DELETE ALREADY PROCESSED DATA    */
                                 /* FROM WORK BUFFER EXCEPT LAST DLM */
      IX=LENGTH(B)+1 ,.          /* IX NOW PTS TO WHAT WILL BE THE   */
           /* CHAR IN B WHICH WILL CORRESPOND TO WHAT WILL BE THE     */
           /* 1ST CHAR OF THE TR OVERLAY ON B.                        */
      LTT=1 ,.          /* RESET PTR TO MOST RECENT DELIM SCANNED.    */
      NC=0 ,.           /* RESET PTR TO LAST TR CHAR ALREADY SCANNED. */
   REREAD ..            /* GET PRIMARY INPUT.                         */
      IF IOPT(7) THEN DO ,.
      NPAGESAV=NPAGE ,.          /* SAVE FOR PAGE CHANGE CHECK.       */
      GET FILE(CLIN) EDIT(TP,NPAGE,IPGCNT,J,LEN,A)
      (R(RFORM)) ,.
   RFORM .. FORMAT(A(1),F(4),F(5),F(6),F(4),A(LEN)) ,.
      IF NPAGE NE NPAGESAV THEN IF IOPT(6) THEN CALL PAGELOG ,.
      IF IOPT(3) THEN            /* LIST INPUT LINES OPTION.          */
      CALL LINELOG ,.
      IF IPGCNT LT 0 THEN SIGNAL ENDFILE(IN) ,.   END ,.
        ELSE DO ,.               /* FOR STRAIGHT ICI INPUT.           */
      READ FILE(IN) INTO(A) ,.             /* READ ONE LINE.          */
      IF A=(80) ' ' THEN GOTO REREAD ,.  /* DISREGARD BLANK LINES.    */
      J=J+1 ,.    END ,.                 /* INPUT RECORD COUNTER.   */
      IF J=NOPT THEN DO ,.               /* FOR OPTION CHAIN.       */
   ENTER ..                    /* ENTRY TO MAIN LOOP.                */
      GET FILE(SYSIN) EDIT(OPAR,NOPT,SECS,SMOD)
      (B(16)  ,F(7),A(27),A(30)) ,.
      K=INDEX(SMOD,' ') ,.       /* FIND BLANK AFTER SECTYPE LETTERS */
      SMOD=SUBSTR(SMOD,1,K-1) ,.           /* CHOP OFF BLANKS.        */
      K=INDEX(SECS,' ') ,.       /* SEARCH FOR BLANK AFTER SEC LETRS */
      SECS=SUBSTR(SECS,1,K-1) ,.           /* CHOP OFF EXTRA BLANKS.  */
      IF IOPT(1) THEN SIGNAL ENDFILE(IN) ,.        /* STOP OPTION.    */
      END ,.
      IF IOPT(15) THEN GOTO REREAD ,.    /* THE LINE SKIP OTPION.     */
      IF IOPT(7) THEN GOTO CLEANIN ,.
      IPGCNT=IPGCNT+1 ,.         /* COUNTS LINES ON CURRENT PAGE.     */
      IF IOPT(11) THEN DO ,.     /* ILLEGAL CHARACTER CHECK.          */
   HEXCHKS .. K=VERIFY(A,LEGITS) ,.      /* SRCH FOR ILLEGAL CHAR.    */
        IF K NE 0 THEN DO ,.             /* IF FOUND ONE THEN . .     */
        CP=SUBSTR(A,K,1) ,.    /* GET ILLEGAL CHAR INTO CP &ICP.      */
        PUT FILE(SYSPRINT) SKIP(2) EDIT
        ('***** ILLEGAL CHAR, ''',CP,''', IN LINE',J,ICP)
        (3 A,F(6),F(4)) ,.
        SUBSTR(A,K,1)='?' ,.   /* CHANGE ILLEGAL CHAR TO QUESTION    */
          /* MARK TO CHK FOR OTHER ILLEGAL CHARS ON SAME LINE.        */
        ILLEGAL=ILLEGAL+1 ,.   /* COUNT ILLEGAL CHARS FOR STATS.     */
        GOTO HEXCHKS ,.   END ,.   END ,.
      IF IOPT(3) THEN            /* LIST INPUT LINES OPTION.          */
      CALL LINELOG ,.
      NUL=75 ,.                  /* CURRENTLY PRESUMED NUMBER OF      */
                                 /* SIGNIFICANT CHARACTERS IN LINE.   */
   NULLS .. IXR=INDEX(A,'|') ,. /* SRCH FOR A NULL CHAR (PREFIX).     */
      IF IXR GT 0 THEN DO ,.     /* IF FOUND ONE THEN DO ,.           */
      A=SUBSTR(A,1,IXR-1) CAT SUBSTR(A,IXR+1) ,.   /* DELETE IT.    */
      NUL=NUL-1 ,.               /* COUNT THE DELETION DOWN.          */
      GOTO NULLS ,.   END ,.     /* CHECK IF MORE NULLS.              */
      DTMP=A ,.                  /* TEMP, SEE BELOW SHORTLY.          */
      ITP=0 ,.                   /* INDICATES NO TYPIST ENTRY FOUND   */
                                 /* YET ON THIS LINE, CHECK FOLLOWS.  */
      IF CTMP=STAR THEN          /* 1ST PART OF TYPIST'S ENTRY CHECK  */
      IF VERIFY(ETMP,CAPS)=0     /* AND 2ND PART OF CHECK.            */
        THEN DO ,.               /* PROCESS APPARENT TYPIST'S ENTRY.  */
        IF IPGCNT NE 30 THEN PUT FILE(SYSPRINT) SKIP(1) EDIT
        ('$$$$$$$$$ APPARENT TYPIST ENTRY IS',IPGCNT,
```

```
            ' LINES FROM PREVIOUS LINE,',J,', PREVIOUS PAGE WAS',
            NPAGE) (A) ,.
            ITP=VERIFY(SUBSTR(A,3),'0123456789') ,.   /* SRCH FOR BLNK*/
            IF SUBSTR(A,ITP+2,1) NE ' ' THEN IXL=0 ,.   ELSE IXL=1 ,.
              /* ABOVE STMT GENERATES LOGIC TEMP USED TWICE BELOW.    */
            IF ITP GT 1 THEN DO ,.              /* IF EVEN PARTLY LEGIT   */
                                                /* PAGE NO. EXISTS.       */
            GET STRING(SUBSTR(A,3,ITP-1)) EDIT(IPAGE) (F(ITP-1)) ,.
            IF IPAGE NE NPAGE+1 THEN IF NPAGE NE 0 THEN
            PUT FILE(SYSPRINT) SKIP(1) EDIT
            ('$$$$$$$$ BAD PAGE SEQUENCE',NPAGE,IPAGE,
            ' IN LINE',J) (A) ,.
            NPAGE=IPAGE ,.   ITP=ITP+2 ,.   IPGCNT=0 ,.   TP=ETMP ,.
            IF IOPT(6) THEN CALL PAGELOG ,. /* LOG PAGE CHANGE.       */
            IF IXL=0 THEN IF SUBSTR(A,ITP  ,2)='$@' THEN DO ,.
              /* ABOVE STMT IS FOR 'DOLLAR SIGN-AT SIGN IN TYPIST ENT */
               ITP=ITP+1 ,.   IF IOPT(4) THEN IXL=1 ,.   END ,.
            ELSE ITP=ITP-1 ,.     /* INCLUDE NONBLANK AFTER DIGIT(S). */
            IF IOPT(16) THEN DO ,.
            IF  INDEX(A,'/')=0 THEN CORRBIT=1 ,.   END ,.
            ELSE CORRBIT=1 ,.     /* LEAVE CORRECTION TEXT MODE.      */
          /* FOR ABOVE GRP  YOU GENERALLY ONLY WANT TO LEAVE CORRECT- */
          /* ION TEXT MODE IF THERE ARE NO SLASHES IN NEW PAGE LINE.  */
            END ,.
            IF ITP LE 1 OR IXL=0 THEN
            PUT FILE(SYSPRINT) SKIP(1) EDIT
            ('$$$$$$ BAD TYPIST ENTRY, LINE',J,A) (2 A,X(1)) ,.
            END ,.                      /* END OF TYPIST'S ENTRY PROCESSING */
          IF CORRBIT=0 THEN GOTO REREAD ,.   /* SKIPPING CORRECTION TXT*/
          IXR=INDEX(A,'@@@') ,.     /* SRCH FOR TRIPLE 'AT'.          */
          IF IXR NE 0 THEN DO ,.    /* IF FOUND LINE DELETE THEN DO ,.  */
          F=SUBSTR(A,MIN(IXR+3,LENGTH(A))),. /* GET REMAINDER OF LINE. */
          IF VERIFY(F,' ') NE 0 THEN PUT FILE(SYSPRINT) SKIP(1) EDIT
          ('****** BAD TRIPLE ''AT'' IN REC',J,A) (2 A,X(1)) ,.
          GOTO REREAD ,.   END ,.  /* SKIP FURTHER PROCESSING OF LINE. */
          IF DTMP='@@' THEN SUBSTR(A,1,2)='  ' ,.       /* REPLACES 2   */
               /* 'AT' SIGNS AT BEGINNING OF LINE WITH SPACES.         */
          IF CTMP='@' THEN SUBSTR(A,1,1)=' ' ,.          /* CHANGES ANY  */
                                /* 'AT'S IN 1ST COLUMN TO BLANKS.       */
          IF DTMP='**' THEN DO ,.   CORRBIT=0 ,.         /* IGNORE CHINA */
          GOTO REREAD ,.   END ,.           /* DATA CORRECTION TEXT.    */
        DUBLQUOT ..                 /* DOUBLE QUOTE CLEANUP.          */
          IXR=INDEX(A,'''''') ,.    /* SRCH FOR A DOUBLE QUOTE.       */
          IF IXR GT 0 THEN DO ,.    /* IF FOUND A PAIR . . . . . . .  */
             A=SUBSTR(A,1,IXR-1) CAT '"' CAT SUBSTR(A,IXR+2) ,.
             GOTO DUBLQUOT ,.   END ,.      /* CHECK FOR MORE.         */
          DO IXL=LENGTH(A) TO LENGTH(A)-4 ,. /* SRCH LAST FIVE CHARS    */
          IF SUBSTR(A,IXL,1) NE ' ' THEN DO ,.          /* OF LINE AND  */
          NUL=IXL ,.   GOTO CHOPOFF ,.   END ,.         /* IF FIND NON- */
          END ,.        /* BLANK THEN INCLUDE IT & CHARS TO ITS LEFT.   */
        CHOPOFF ..
             A=SUBSTR(A,ITP+1,NUL-ITP) ,.     /* CHOP OFF TRAILING      */
                        /* BLNKS, AND LEADING TYPIST ENTRY, IF ANY.     */
                        /* ITP WILL BE 0 IF NO TYPIST ENTRY ON LINE.    */
        DBLATSRCH ..    /* REENTRY POINT AFTER ENDFILE(IN) ON UNIT.     */
          IXR=INDEX(A,'@@') ,.      /* SEARCH FOR DOUBLE 'AT'.        */
          IF IXR NE 0 THEN DO ,.    /* IF FIND DOUBLE 'AT' THEN DO . . .*/
          IXL=IXR ,.   IXR=IXL+2 ,.         /* PTRS TO GUESS WHERE.     */
        BLNKSRCH ..     /* SRCH BACKWARDS FOR FIRST BLANK PRECEDING     */
                        /* DOUBLE 'AT' UNTIL START OF LINE.             */
          IXL=IXL-1 ,.              /* DECREMENT CHAR PTR.              */
          IF IXL GT 0 THEN IF SUBSTR(A,IXL,1) NE ' ' THEN GOTO BLNKSRCH,.
          A=SUBSTR(A,1,IXL) CAT SUBSTR(A,MIN(IXR,LENGTH(A)),
             MAX(0,LENGTH(A)-IXR+1)) ,.     /* DELETE WORD.             */
          GOTO DBLATSRCH ,.   END ,.        /* SEE IF MORE.             */
        ONEAT ..                    /* TOP OF LOOP FOR SINGLE 'AT'S.    */
```

```
      IXR=INDEX(A,'@') ,.          /* SRCH FOR A SINGLE 'AT'.          */
      IF IXR GT 0 THEN DO ,.       /* IF FOUND ONE THEN . . . . . .    */
      IF IXR LT LENGTH(A) THEN F=SUBSTR(A,IXR+1) ,.
      ELSE F='' ,.                 /* AVOIDS STRG PROBLEM.             */
      IF IXR LE 2 THEN A=F ,.    ELSE A=SUBSTR(A,1,IXR-2) CAT F ,.
      GOTO ONEAT ,.   END ,.       /* CHECK FOR MORE SINGLE 'AT'S.     */
   CLEANIN ..
      IF IOPT(14) THEN       DO ,.   LEN=LENGTH(A) ,.
      PUT FILE(CLOUT) EDIT(TP,NPAGE,IPGCNT,J,LEN,A) (R(RFORM)) ,.
      END ,.
      CAPCHK=TRANSLATE(A,          /* FOR UPPERCASE CHAR COUNT STATS.  */
      'AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA',
      'ABCDEFGHIJKLMNOPQRSTUVWXYZ+*/@#$&!()_?[]:"%') ,.
      DO IXL=1 TO LENGTH(A) ,. /* UPPERCASE CHAR COUNTING LOOP.        */
      IF CAPDEF(IXL)='A' THEN NCAPS=NCAPS+1 ,.   END,.
      NCHARS=NCHARS+LENGTH(A) ,.           /* COUNT CLEANED INPUT.    */
      B=B CAT A  ,.                /* ADD CLEANED INPUT TO WORK BUFR.  */
      TR=       TRANSLATE(A,'********' ,'+=$()[]%') ,.
                     /* ABOVE STATEMENT SETS UP THE DELIM BUFFER.       */
      GOTO RESRCH ,.               /* CONTINUE SEARCHING FOR A DLM.    */
      END ,.                       /* END OF INPUT COROUTINE.          */
      ND=ND+NC ,.    /* SET UP ACTUAL PTR TO DLM MARKER IN TR BUFR      */
      CC=SUBSTR(B,IX+ND-1,1) ,.                 /* GET DLM ITSELF FROM B */
      ICO=IC ,.                                 /* SAVE PREVIOUS DIM NO. */
      DO IC=1 TO 9 ,.     (NOSUBRG) ..          /* SRCH FOR NEW DIM NO.  */
      IF CC=DLM(IC) THEN GOTO DLMHIT ,.         /* IF THIS IS THE NUMBER */
      END ,.                                    /* CODE OF THE CHARACTER.*/
   DLMHIT ..           /* CONTINUE INITIAL PROCESSING OF ANY HIT.       */
      LOS=LTT ,.       /* RESET PTR TO 'OLD LAST SCANNED CHAR'.         */
      LTT=IX+ND-1 ,. /* SET UP PTR TO NEW 'LAST SCANNED DLM'.           */
      IF IOPT(9) THEN                           /* FOR DEBUG.            */
      PUT FILE(SYSPRINT) SKIP(1) EDIT
      ('CC=''',CC,'''',' IC=',IC,' MODE=',MODE,' LOS=',LOS)
      (3 A,3 (A,F(5))) ,.
           /* MODE=0=STRUCTURED MODE, MODE GT 0 = AN EXTRANEOUS        */
           /* TEXT MODE, MODE LT 0 = AN INDEX MODE, EXCEPT THAT -5     */
           /* IS 'INDEX IN STRUCTURED TEXT' MODE.                      */
      IF MODE=0 OR MODE=-5 THEN DO ,.     /* IF IN STRUCTURED MODE     */
      IXL=1 ,.           /* FOR EXCLUDING LEFT DELIM UNLESS ASTERISK,    */
                         /* AS IN THE STMT BELOW (ICO=1=ASTERISK).       */
      IF ICO=1 THEN DO ,.          /* IF START/END OF SECTION.          */
      CD=SUBSTR(B,LOS+1,1) ,.  /* SECTION TYPE LETTER.                  */
      IF IEND NE 0 THEN            /* EXCEPT FOR DUMMY LAST SECTION.    */
      IF INDEX(SECS,CD)=0 THEN DO ,.      /* IF NOT LEGAL SECTYPE.     */
      PUT FILE(SYSPRINT) SKIP(2) EDIT(
      '******** ILLEGAL SECTION ''*',CD,''' IN LINE',J) (A) ,.
      SUBSTR(B,LOS+1,1)=TRANSLATE(CD,SMALLS,CAPS) ,.
                         /* ABOVE STATEMENT CONVERTS POSSIBLE CAPITAL    */
                         /* ILLEGAL SECTIONS TO SMALL LETTER SECTIONS.   */
      ISERRI=0 ,.    END ,.        /* TURNS ON ERROR MESSAGE SWITCH.    */
      K=3 ,.             /* INITIAL POINTER FOR DOUBLE BLANK SEARCH.    */
   TWOBLNK ..            /* TOP OF DOUBLE BLANK CLEANING LOOP.          */
      IXL=0 ,.
      IF K LT LENGTH(STR) THEN
      IXL=INDEX(SUBSTR(STR,K),'  ') ,.    /* SRCH FOR A DOUBLE BLNK    */
      IF IXL GT 0 THEN DO ,.    /* IF FOUND DOUBLE BLANKS.             */
         K=K+IXL   ,.                     /* K NOW PTS TO 2ND BLNK OF PAIR.*/
         DTMP=SUBSTR(STR,K-3,2) ,.        /* DTMP=2 CHARS PRECEDING    */
           /* THE PAIR OF BLANKS.  CTMP = 1ST OF THESE, ETMP=2ND.       */
      IF ETMP='.' OR ETMP='?' OR ETMP='!'
      THEN GOTO TWOBLNK ,.      /* OK, ALLOW DOUBLE BLNKS TO LIVE.     */
      PBIT= CTMP='.' OR CTMP='?' OR CTMP='!' ,.
      QBIT=ETMP='''' OR ETMP='"' ,.
      IF NOT (QBIT AND PBIT) THEN
         STR=SUBSTR(STR,1,K-2) CAT SUBSTR(STR,K  ) ,.
      GOTO TWOBLNK ,.   END ,. /* BOTTOM OF DOUBLE BLANK CLEANING.    */
```

```
          IF LENGTH(STR) GT 2 THEN DO ,.
          STR=STR CAT (15) ' ' ,.  /* APPEND 2 BLANKS FOR STRUCTURE,    */
                                   /* AND 13 MORE FOR TAPE BLK.         */
                                   /* THUS AT LEAST 18 BYTES ON TAPE.   */
          IF SUBSTR(STR,3,1)='p' THEN DO ,.  /* IF PLAY SECTION.        */
             /* THIS DOGROUP CHECKS FOR DATES IN P SECTIONS.            */
          K=MIN(10,LENGTH(STR)-4) ,.          /* TEMP USED 3 TIMES.     */
          TDATE=SUBSTR(STR,4,K) ,. /* GET POTENTIAL DATE AREA.          */
          IXL=0 ,.                 /* INITIAL CHAR PTR OFFSET BY -1.    */
        FSRCH .. IXL=IXL+1 ,.      /* ADVANCE PTR TO NEXT CHARACTER.    */
          IF IXL LE K THEN DO ,.   /* WHILE WITHIN TDATE'S LENGTH.      */
          CE=TDATA(IXL) ,.                    /* LOAD ICE WITH DEC VAL. */
          IF ICE=64 THEN GOTO FSRCH ,.        /* SKIP IF LEADING BLANK. */
          IF ICE GE 240 THEN DO ,.            /* IF FOUND 1ST DIGIT.    */
          IXR=IXL ,.                          /* REMEMBER ITS POSITION. */
        ESRCH .. ITP=IXR+1 ,.      /* ADVANCES PTR TO NEXT CHAR.        */
        GSRCH ..
          IF ITP LE K THEN DO ,.   /* WHILE STILL IN RANGE OF TDATE.    */
          CE=TDATA(ITP) ,.         /* USE SUBSCRIPT RATHER THAN SUBSTR */
          IF ICE GE 240 THEN DO ,. /* IF DEC DIGIT THEN DO ,.           */
          IXR=ITP ,.   GOTO ESRCH ,.  END ,. /* REMEMBER ITS POSITION. */
          IF ICE=64 THEN DO ,.                /* IF ONLY A BLANK THEN   */
          ITP=ITP+1 ,.   GOTO GSRCH ,.        /* SEARCH FOR MORE DIGITS.*/
          END ,.   END ,.
          K=IXR-IXL+1 ,. /* GET LENGTH OF STRING FROM 1ST DIGIT TO LAST*/
          IF K GE 1 THEN DO ,.     /* IF STRING LENGTH IS GT 0 THEN.    */
          DATEMP=SUBSTR(TDATE,IXL,K) ,.       /* MOVE DATE INTO TEMP.   */
          IF K LE 2 THEN DAY=DATEMP ,.        /* IF LE 2 THEN DAY ONLY. */
          ELSE DO ,.   DAY=SUBSTR(DATEMP,K-1) ,.
          IF K LE 5 THEN MNTHDAY=DATEMP ,.
          ELSE DO ,.   YRMNDAY=DATEMP ,.   MNTHDAY=SUBSTR(DATEMP,6,2) ,.
          END ,.   END ,.   END ,.   END ,.   END ,.   END ,.
          IF VERIFY(SUBSTR(STR,3,1),SMOD)=0 THEN DO ,. /* IF A TYPE OF */
                                   /* SECTION SELECTED FOR OUTPUT.      */
          IF IOPT(2) OR (IOPT(5) AND ISERRI=0) THEN
          PUT FILE(SYSPRINT) SKIP(1) EDIT(SERRI(ISERRI),
                M,STR,'''') (A,F(4),2 A) ,.
          ISERRI=1 ,.              /* RESET ERROR INDICATOR.            */
          M=M+1 ,.                 /* COUNTS OUTPUT RECORDS ,.          */
          WRITE FILE(PURE) FROM(STR) ,.       /* PUT OUT STRUCTURED SEC */
          NOUT=NOUT+LENGTH(STR) ,. /* STATISTIC, TOTAL STRUCTURED CHARS*/
          END ,.   END ,.
          STR=' ' ,.               /* RESET OUTPUT BUFFER.              */
          IF CD='c' THEN DO ,.     /* IF SMALL C FOLLOWS ASTERISK.      */
          MODE=4 ,.   GOTO EXTRAN ,.  END ,. /* ENTER COMMENT MODE.     */
          IXL=0 ,.       /* TO INCLUDE ASTERISK WITH TEXT BEING ADDED  */
          END ,.         /* TO SECTION, 1ST PEICE OF SEC, OF COURSE.   */
          STR=STR CAT SUBSTR(B,LOS+IXL,LIT-LOS-IXL) ,. /* ADD PEICE TO */
                                   /* STRUCTURED SECTION.               */
          END ,.                   /* END OF CODING FOR MODE 0.         */
          ELSE IF MODE LT 0 THEN DO ,.        /* IF IN AN INDEX MODE    */
          IF IC=3 THEN DO ,.                  /* IF IT'S AN '='.        */
          C=C CAT SUBSTR(B,LOS,LIT-LOS+1) ,. /* CREATE INDEX ENTRY.     */
          INDXS=INDXS+1 ,.         /* COUNT INDEX ENTRIES PUT OUT.      */
          IF IOPT(13) THEN
          PUT FILE(SYSPRINT) LIST('INDX=''' CAT C CAT '''') ,.
          C='' ,.                             /* NULL C.                */
          GOTO MODECHNG ,.   END ,.           /* GET NEXT PIECE.        */
          ELSE DO ,.                /* NON-'=' DIM IN INDEX ENTRY, BAD. */
          C=C CAT SUBSTR(B,LOS,LIT-LOS) ,.    /* ADD PIECE TO INDEX.    */
          END ,.   END ,.
        ELSE EXTRAN .. DO ,.                  /* AN EXTRANEOUS TXT MODE */
          CALL PAGECHK(SUBSTR(B,LOS,LIT-LOS)) ,.   END ,.
        MODECHNG ..        /* TRANSFORMS MODE, DEPENDING ON OLD MODE &   */
                           /* CURRENT DLM, EVEN IF ILLEGAL IN CONTEXT.   */
          MODOLD=MODE ,.           /* SAVE CURRENT MODE (NOT YET USED)  */
```

```
      NEWFRK(MODE,IC)=NEWFRK(MODE,IC)+1 ,.
           /* ABOVE STMT IS FREQUENCY TABLE OF DELIMITERS FOUND    */
           /* IN VARIOUS MODES, ILLEGAL AND LEGAL.                 */
      IF ERRAY(MODE,IC) THEN DO ,.
      IF IOPT(12) THEN
      PUT FILE(SYSPRINT) SKIP(1) EDIT
      ('///// ILLEGAL DELIM ''',CC,''' LINE',J, 'MODE=',MODE)
      (4 A,X(14),A,F(2)) ,.
      IF IOPT(8) THEN PUT FILE(SYSPRINT)  EDIT
      (SUBSTR(B,LOS,LTT-LOS+1)) (X(2),A) ,.
      IERRCNT=IERRCNT+1 ,.      /* TOTAL COUNT OF ILLEGAL DELIMS.   */
      ISERRI=0 ,.     /* INDICATE DUMPING ON PRINTER OF THIS      */
      END ,.          /* SECTION IF IOPT(5) IS ALSO SPECIFIED.    */
      IF IEND=0 THEN DO ,.      /* IF REAL OR SIGNALLED ENDFILE(IN) */
      PUT FILE(SYSPRINT) SKIP(2) EDIT
      (' INPUT LINES=' ,J,' CLEAN INPUT CHARS=',NCHARS,
      ' CLEAN UPPERCASE=',NCAPS,' INDEX ENTRIES=',INDXS,
      ' STRUCTURED OUTPUT CHARS=',NOUT,' TOTAL ILLEGAL DELIMS=',
      IERRCNT,' OUTPUT SECS=',M,' ILLEGAL CHARS=',ILLEGAL,
      ' LOGICAL OUTPUT CHARS=',NOUT-15*M,' TOTAL INPUT CHARS=',
      80*J,' ACCOUNTABLE INPUT CHARS=',4*J/15+NCAPS+NCHARS)
      (A,COL(27),F(8),SKIP)
      ((DLM(IXL) DO IXL=1 TO 9)) (COL(8),9 (A,X(7)))
      (((NEWFRK(IXL,IXR) DO IXR=1 TO 9),
      (ERRAY(IXL,IXR) DO IXR=1 TO 9) DO IXL=-5 TO 4))
      (SKIP,9 F(8)) ,.
      IF IOPT(14) THEN
      PUT FILE(CLOUT) EDIT(TP,NPAGE,-1,J,LEN,A) (R(EFORM)) ,.
      RETURN ,.   END ,.        /* END OF PROGRAM EXECUTION.        */
      MODE=NEWMODE(MODE,IC) ,. /* ACTUAL MODE TRANSFORMATION.       */
      NC=ND ,.
      GOTO RESRCH ,.            /* LOOK FOR NEXT DLM.               */
PAGECHK .. PROC(TEXT) ,.        /* CHECKS A SEGMENT OF TEXT FOR     */
                                /* PAGE ENTRIES, ROMAN OR DECIMAL.  */
      DCL TEXT CHAR(3625) VAR ,.
      LP=0 ,.                   /* INITIALIZE PTR (OFFSET BY -1).   */
LOOKAGIN ..                     /* TOP OF DECIMAL PAGE NUMBER LOOP. */
      IF IP LE LENGTH(TEXT)-3 THEN DO ,. /* TST TO AVOID STRG ERR. */
      IP=INDEX(SUBSTR(TEXT,LP+1),' p') ,.          /* IF FOUND     */
                                                   /* BLNK-SMALL P */
      IF IP GT 0 THEN DO ,.
      LP=IP+LP+2 ,.   NUMP=0 ,.
      DO IP=LP TO MIN(LP+4,LENGTH(TEXT)) ,.
      CP=SUBSTR(TEXT,IP,1) ,.  /* GET POSSIBLE DECIMAL CHAR.       */
      IF ICP LT 240 THEN IF ICP=64 THEN IF NUMP GT 0 THEN DO ,.
         NEWPAGE=NUMP ,.        /* SAVE NEW PAGE NUMBER.           */
         PUT FILE(SYSPRINT) SKIP(1) EDIT('NEW PAGE',NEWPAGE)
         (A,F(5)) ,.
      GOTO LOOKAGIN ,.   END ,.          /* CHK IF MORE PAGE NO.'S */
      ELSE GOTO LOOKAGIN ,.   ELSE GOTO LOOKAGIN ,.
      IF ICP GT 249 THEN GOTO LOOKAGIN ,.
      NUMP=NUMP*10+ICP-240 ,.  /* ADD IN VALUE OF NEW DIGIT ETC.   */
      END ,.
      PUT FILE(SYSPRINT) SKIP(1) EDIT(
      '*** PAGE NUMBER TOO LONG IN OR BEFORE LINE',J) (A) ,.
      ISERRI=0 ,.               /* INDICATE ERR IF SECTION LISTED.  */
      GOTO LOOKAGIN ,.   END ,.   END ,.
      LP=0 ,.
NROMAN ..                       /* CHECK FOR ROMAN PAGE NUMBERS.    */
      IF IP LE LENGTH(TEXT)-3 THEN DO ,. /* TST TO AVOID STRG ERR. */
      IP=INDEX(SUBSTR(TEXT,LP+1),' i') ,.
      IF IP GT 0 THEN DO ,.
      LP=IP+LP+2 ,.   ROMAN='' ,.
      DO IP=LP TO MIN(LP+4,LENGTH(TEXT)) ,.
      CP=SUBSTR(TEXT,IP,1) ,.
      IF ICP=64 THEN IF LENGTH(ROMAN) GT 0 THEN DO ,.
```

```
            ROMAN=POMAN ,.              /* ACCEPT NEW ROMAN NUMERAL PAGE NO.*/
            PUT FILE(SYSPRINT) SKIP(1) EDIT
            ('***** NEW PAGE ',ROMAN) (A) ,.
            ISERRI=0 ,.
            GOTO NROMAN ,.   END ,.
            IF ICP=229 OR ICP=231 OR ICP=201 THEN POMAN=POMAN CAT CP ,.
            ELSE GOTO PAGERR ,.   END ,.
        PAGERR ..                       /* BAD ROMAN NUMERAL PAGE ENTRY.    */
            PUT FILE(SYSPRINT) SKIP(1) EDIT
            ('********** BAD PAGE IN OR BEFORE RECORD ',J) (A,F(5)) ,.
            GOTO NROMAN ,.   END ,.   END ,.
            END PAGECHK ,.
        LINELOG .. PROC ,.              /* LINE LISTING PROCEDURE.          */
            PUT FILE(SYSPRINT) SKIP(1) EDIT
            (A,'''',J,YRMNDAY,'/',MNTHDAY,'/',DAY,' M=',M)
            (2 A,COL(83),F(7),X(1),5 A,COL(101),A,F(5)) ,.
            END LINELOG ,.
        PAGELOG .. PROC ,.              /* PROC TO LOG PAGE CHANGES ETC.    */
               PUT FILE(SYSPRINT) SKIP(1) EDIT
               ('* NEWPAGE ',  TP,NPAGE,' ON LINE',J,YRMNDAY,'/',
               MNTHDAY,'/',DAY,'M=',M) (5 A,X(2)) ,.
               END PAGELOG ,.
            END SCAN ,.
    /*
    //STRUCTUR JOB 99999211,WCD,MSGLEVEL=(1,1)
    //S EXEC LSPUPGEN,DSSP=OLD,SSP=20,MEM=STRCT,
    // PARM.PL1L='C48,NT,A,X,FW,OPT=0,SIZE=90K'
    //BAKUP.SYSUT1  DD *,DCB=BLKSIZE=80
             /* STRUCTUR PROGRAM */
            /* NOTE THAT, SINCE THE ONE LINE MESSAGE FOR EACH SECTION IS */
            /* PUT OUT AFTER MOST PROCESSING OF THE SECTION, THE ERROR   */
            /* MESSAGES WILL GENERALLY PRECEDE THE SECTION MESSAGE LINE. */
            /* SEQL OUTPUT IS VERY SLOW.  CORE/TIME TRADEOFF COULD VERY  */
            /* POSSIBLY BE MADE.                                         */
        STRUCT .. PROC OPTIONS(MAIN) ,.
        G .. FORMAT(A,F(4)) ,.
        H .. FORMAT(A,F(5)) ,.
            DCL             /* DECLARE STATEMENT FOR CONSTANTS & SUCH.    */
            MONTHSMALLS CHAR(21) STATIC INIT('abcdefghijlmnoprstuvy'),
            MONTHCAPS CHAR(21) STATIC INIT('ABCDEFGHIJLMNOPRSTUVY'),
            /* ABOVE 2 CONSTANTS ARE FOR CONVERTING NAME OF MONTH IN     */
            /* LADDER REFS TO ALL UPPER CASE.                            */
            STYPE(26) CHAR(1) STATIC INITIAL('p','a','b','i','o','t','u',
            'p','p','p','p','p','p', 'd','e','m','s','p','p','p','p',
            'p','p','p','p','p'),    /* TITLED SECTIONS ARE THOSE FROM 1 */
                /* 13 IN THIS ARRAY, UNTITLED ARE FROM 14 TO 26.    L    */
                /* ALL P'S EXCEPT THE FIRST ARE JUST FOR FILL.           */
            FA CHAR(1) INITIAL('?'),          /* ITEM-START DELIMITER    */
            FC CHAR(1) INITIAL('?'),          /* GROUP-START DELIMITER   */
            ERRMSG ENTRY(CHAR(4), FIXED BIN(15,0)),
            STRU ENV(REGIONAL(3)) SEQL OUTPUT KEYED,
            COGO ENTRY(FIXED BIN(15,0),FIXED BIN(15,0),FIXED BIN(15,0),
              LABEL),
            GTCK ENTRY(FIXED BIN(15,0),CHAR(1)),        /* ROUTINE TO    */
                                    /* GET ITEM & DETERMINE IF IT IS A   */
                                    /* LADDER ENTRY, AND PROCESS IT IF   */
                                    /* IS A LEGITIMATE ONE.              */
            GITM ENTRY(FIXED BIN(15,0),CHAR(1)),
            SD CHAR(2) INITIAL(' '),          /* BLANK-SECTION DELIM     */
            SYSPRINT PRINT ENV(F(133,133)) ,.
            DCL
            IERRNO STATIC INIT(0),    /* ERROR MESSAGE COUNT.            */
            IS,                       /* PTR TO CURRENT SEASON-END SPECS. */
            ISNOW,                    /* SECTION NO. WITHIN PERFORMANCE.  */
            ISCNT(20),                /* SECTION NUMBER FOR SEASON END.   */
            ISDATE(20,3),             /* SEASON-END DATES                 */
```

```
THTR(20) CHAR(8),          /* THEATRE FOR SEASONEND CHECKS.     */
MONSTR(12) CHAR(9) VAR,    /* MONTH CHAR STRINGS ARRAY.         */
IPDATE(4),                 /* TEMPS FOR POTENTIAL NEW DATES.    */
IRDATE(3),                 /* ARRAY OF ACTUAL CURRENT DATE.     */
DSARBIT(8) BIT(1) DEFINED IDBITS POSITION(9),
IDATCHK(3,2) STATIC INITIAL(1,31,1,12,1659,1800),
CH4 CHAR(4),               /* FOR GTCK'S 'SEE ' CHECKING, CH3   */
                           /* IS DEFINED ON CH4.                */
CH3 CHAR(3) DEFINED CH4,   /* FOR GTCK'S 'AS ' CHECKING         */
CHPAGE CHAR(4),            /* CURRENT PAGE NO. AS CHAR STRING   */
ILD(3),                    /* REFERRED-TO DATE IN LADDER REF    */
C CHAR(120) VAR,           /* GITM & GTCK RETURN ITEM IN C      */
CSV CHAR(120) VAR,         /* TEMP SAVE AREA FOR C              */
C1 CHAR(1),                /* TEMP.                             */
C2 CHAR(1) DEF MNUM POS(2),        /* USED IN COGO ONLY         */
MNUM,                      /* C2 IS DEFINED ON MNUM.            */
C3 CHAR(1),                /* TEMP, OR LADDER REF TYPE CHAR -   */
                           /* C, L, OR S.                       */
CHOGO CHAR(20),            /* CHOGA IS DEFINED ON CHOGO.        */
CHOGA(20) CHAR(1) DEFINED CHOGO ,.
DCL
SKEY CHAR(12),             /* RECORDED KEY FOR STRU FILE HERE.  */
IDATE FIXED BIN(31,0),     /* BASE FOR CHIDATE'S 3 DATE CHARS.  */
CHIDATE CHAR(3) DEFINED IDATE POSITION(2),   /* SEE ABOVE.      */
IOPT(32) BIT(1),           /* OPTION BITS ARRAY                 */
MAXOUT INIT(0) STATIC,     /* STATISTIC, MAX OUTPUT BLKSIZE.    */
IPAG INITIAL(0),           /* KEEPER OF THE CURRENT PAGE        */
IPAGES(0 .. 36) CHAR(4),   /* AS ABOVE, BUT FOR MAP/LOG.        */
IDAYS (0 .. 36) FIXED BIN(31,0),   /* DAY FOR MAP/LOG.          */
MONTHS(0 .. 36) CHAR(2),           /* MONTH FOR MAP/LOG.        */
ERRORS(0 .. 36) CHAR(4) ,          /* 1 ERROR FOR MAP/LOG.      */
NOTES (0 .. 36) CHAR(3) VAR,       /* NEW SECTION AND PAGE      */
                           /* INDICATORS FOR MAP/LOG            */
TITLTM CHAR(8) VAR,        /* FOR TITLE TIME ENTRY.             */
FGBIT BIT(1),              /* FIRST GRP DONE BIT. '0'B=DONE.    */
TTMBIT BIT(1),             /* TITLE TIME ENTRY BIT.  '1'B='YES  */
                           /* THERE IS A TITLE TIME ENTRY.'     */
BIT1 BIT(1),               /* '1'B = 'DATE WAS UPDATED IN THIS  */
                           /* SECTION', '0'B MEANS OPPOSITE.    */
ERR BIT(1),                /* ERROR BIT='1'B IF ERRMSG CALLED   */
                           /* DURING PROCESSING OF THIS BLK.    */
BITSEC BIT(1), /* IF='1'B THEN CAST TIME ENTRIES NOT ALLOWED */
TITBIT BIT(1), /* '1'B INDICATES NON-PLAY TITLED SECTION.    */
IOYEAR FIXED BIN(31,0) INITIAL(0),
P CHAR(3625) VAR,          /* INPUT BUFR, P STANDS FOR 'PURE'.  */
S CHAR(3625) VAR,          /* OUTPUT BUFR.  'S' STANDS FOR      */
                           /* 'STRUCTURED'.  GOES INTO STRU.    */
SMOD CHAR(1),              /* CURRENT SECTYPE INDICATOR CHAR    */
CTMP CHAR(8),  /* THEATRE ABBREVIATIONS HELD HERE SKEY & KEY */
(NOUT,NLAD,NSEE,NCGLE) FIXED BIN(31,0) STATIC INIT(0),
/* ABOVE 4 VARIABLES FOR STATISTICS, NUMBER OF OUTPUT CHARS,
   NUMBER OF LADDER AS REFS, SEE REFS, CGLE'S RESPECTIVELY.  */
(IO INITIAL(0),            /* OUTPUT RECORD COUNTER.            */
J INITIAL(0),              /* NUMBER OF CURRENT INPUT BLK.      */
NOPT INITIAL(1),           /* DEBUG CHAIN LINK VARIABLE.        */
KSTRU INITIAL(0))          /* CURRENT REL TRK IN STRU FILE      */
   FIXED BIN(31,0),
NONEXIT LABEL,             /* GTCK & GITM EXIT FOR NO ITEM CASE*/
EXITSV LABEL,              /* TEMP SAVE FOR NONEXIT ABOVE.      */
CONGO LABEL(NOTAPE,THEATRE,TITLE) ,.
ON CONVERSION GOTO CONGO ,.        /* LABEL VARIABLE SWITCH     */
ON ENDFILE(PURE) GOTO ENDPROG ,.
ON KEY(STRU) BEGIN ,.
GOTO STRUWRITE ,.   END ,.
DO L=1 TO 12 ,.
GET FILE(SYSIN) EDIT(K,C,SMOD)
```

```
        (F(1),A(9),A(70)) ,.
        MONSTR(L)=SUBSTR(C,1,K) ,.  END ,. /* SHORTEN INTO MONSTR.   */
        DO IS=1 TO 20 ,.
        GET FILE(SYSIN) EDIT(ISDATE(IS,*),THTR(IS),ISCNT(IS),SMOD)
        (F(4),2 F(2),A(8),F(2),A(62))  ,.
        IF ISCNT(IS)=0 THEN GOTO SEASONEND ,.   END ,.
    SEASONEND ..IS=1 ,.          /* PTR TO SEASON END SPECS, INITIAL */
    INPUT .. READ FILE(PURE) INTO(P) ,.   /* TOP OF MAIN LOOP.      */
        J=J+1 ,.  I=LENGTH(P) ,.          /* J COUNTS INPUT RECORDS */
        ERR='0'B ,.               /* INDICATE NO ERRORS SO FAR IN BLK */
        IDAYS=IDAY ,.  MONTHS='' ,.  IPAGES='' ,.    /* FOR BLK MAP  */
        ERRORS='' ,.   NOTES='' ,.                   /* FOR BLK MAP  */
        PUT STRING(IPAGES(0)) EDIT(IPAG) (F(4)) ,.   /* FOR BLK MAP  */
        PUT STRING(MONTHS(0)) EDIT(IMONTH) (F(2)) ,. /* FOR BLK MAP  */
        IF J=NOPT THEN GET FILE(SYSIN) EDIT          /* FOR OPTION-  */
        (IOPT,NOPT,SMOD) (32 B(1),F(6),A(42)) ,.     /* CNTRL CHAIN. */
        IF IOPT(25) THEN GOTO INPUT ,.    /* THE SKIP FUNCTION.     */
        IF IOPT(5) THEN SD=' *' ,.        /* SCANNER TYPE SEC-DIM   */
        IF IOPT(1) THEN                   /* RAW INPUT DUMP OPTION  */
        PUT FILE(SYSPRINT) SKIP(2) EDIT(P) (A) ,.
        /* FOLLOWING STATEMENT IS REPLACED BY MTST CARDS FOR USE WITH*/
        /* MTST DATA.   */   GOTO NOTAPE ,.
    NEXTSEC ..                    /* TOP OF SECTION PROCESSING LOOP  */
        SKEY=CHIDATE CAT SMOD CAT CTMP ,. /* SET UP RECORDED KEY     */
        IO=IO+1 ,.                /* INCREMENT OUTPUT BLK COUNTER    */
        IF IOPT(2) OR ERR THEN    /* ONE LINE SECTION HEADER MESSAGE */
    (STRG) ..
        PUT FILE(SYSPRINT) SKIP(1) EDIT
       ('IN=',J,' OUT=',IO,' L=',I,SUBSTR(P,ISI+2,MIN(94,ISR-ISL-1)),
        '''',(IRDATE(L) DO L=3 TO 1 BY -1))
        (COL(3),3 (A,F(4)),X(1),2 A,COL(123),F(4),2 F(3)) ,.
        IF IOPT(04) OR ERR THEN
        PUT FILE(SYSPRINT) SKIP(1) LIST(S) ,.
        IF LENGTH(S) GT 0 THEN    /* AVOID WRITING ZERO LENGTH RECORD */
    WRITO .. DO ,.                /* GENERAL OUTPUT DOGROUP.         */
        NOUT=NOUT+LENGTH(S) ,.    /* TOTAL OUTPUT CHARS STATISTICS.  */
        MAXOUT=MAX(MAXOUT,LENGTH(S)) ,.   /* FOR MAX BLKSIZE STATS. */
        IF IOPT(7) THEN DO ,.     /* OUTPUT ONE SECTION AS UNKEYED   */
        S='  ' CAT SKEY CAT S ,. /* BLK WITH 2 BLANKS FOR MIN 18    */
                                  /* BYTE TAPE BLKS, WITH 'KEY' IN 3. */
        WRITE FILE(STRS) FROM(S) ,.       /*  ONLY WAY FOR TAPE.    */
        S=SUBSTR(S,15) ,.   END ,.        /* REMOVE SKEY FROM S.    */
        IF IOPT(8) THEN
    STRUWRITE ..      /* OUTPUT ONE SECTION AS A KEYED PHYSICAL BLK */
        WRITE FILE(STRU) FROM(S) KEYFROM(SKEY CAT KSTRU) ,.   END ,.
        IF ISNOW=ISCNT(IS) THEN           /* START CHECKING FOR      */
        IF CTMP=THTR(IS) THEN             /* SEASON CHANGE.          */
        IF IRDATE(1)=ISDATE(IS,3) THEN    /* CHECK DAY OF MONTH.     */
        IF IRDATE(2)=ISDATE(IS,2) THEN
        IF IRDATE(3)=ISDATE(IS,1) THEN DO ,.
        IS=IS+1 ,.
        S,SKEY='XXXXXXXXXXXX' ,. /* END OF SEASON DELIMITER.         */
        PUT SKIP(3) EDIT('NEW SEASON   **********************') (A) ,.
        GOTO WRITO ,.   END ,.
        IF ISR=I-2 THEN DO ,.             /* IF LAST SECTION DONE    */
        IF IOPT(3) OR ERR THEN
        PUT FILE(SYSPRINT) SKIP(1) EDIT   /* PRINT MAP OF INPUT BLK */
        ((ERRORS(L/100),L/100,SUBSTR(P,L,MIN(100,I-L+1)),
        IPAGES(L/100),
        NOTES(L/100), (K DO K=1 TO 9),
        MONTHS(L/100),IDAYS(L/100) DO L=1 TO I BY 100))
        (SKIP(1),A(4),F(3),X(3),A(100),X(2),A(4),X(1),A(3),
        SKIP(1),COL(20),9 (F(1),X(9)),
        X(6),A(2),F(3)) ,.
        GOTO INPUT ,.   END ,.    /* GET NEXT INPUT BLK AND PROCESS  */
    ISTSEC ..           /* ENTRY POINT FOR SECTION PROCESSING LOOP  */
```

```
          S='' ,.                    /* INITIALIZE THE OUTPUT BUFFER     */
          ISL=ISR ,.       /* SET END OF LAST SEC=START OF NEXT SECTION. */
                           /* IF THERE IS A NEXT SECTION THEN ISL IS NOW */
                           /* POINTING AT THE BLANK PRECEDING THE SECTION*/
                           /* DELIM, '*' OR PREFIX.                      */
          ISR=INDEX(SUBSTR(P,ISL+1),SD  ) ,. /* GET END OF CURRENT SEC */
          IF ISR=0 THEN ISR=I-2 ,. /* IF IT'S THE LAST SECTION THEN . .*/
          ELSE ISR=ISR+ISL ,.      /* ACTUAL PTR TO SEC'S RIGHT END    */
          SMOD=SUBSTR(P,ISL+2,1) ,.          /* GET SECTION TYPE CODE  */
          NOTES(ISL/100)=NOTES(ISL/100) CAT SMOD ,.    /* NOTE NEW SEC */
          BITSEC='0'B ,.             /* CLEAR 'TIME ENTRY ALLOWED' BIT   */
          TITBIT='0'B ,.             /* CLEAR 'NON-PLAY TITLED SEC' BIT  */
          ITBIT=1 ,.       /* INDICATES NO ROLES OR ACTORS PUT OUT YET   */
                           /* IN CURRENT SECTION, HELPS DEFINE CGLE'S.   */
          ITL=1 ,.         /* SET ONLY TO AVOID STRG ERR IN CASE OF      */
                           /* NOITEM INTERRUPT ON 1ST GITM CALL FOR BLK. */
          DO NS=1 TO 26 ,.
          IF STYPE(NS)=SMOD THEN GOTO SECFOUND ,.   END ,.
          CALL ERRMSG('ISEC',ISL+2) ,.       /* ILLEGAL SECTION TYPE   */
          IO=IO-1 ,.                  /* DECREMENT RECORD OUTPUT COUNTER */
          GOTO NEXTSEC ,.             /* SINCE THIS ILLEGAL SEC SKIPPED. */
     SECFOUND ..
          IF NS=1 THEN   /* IF PLAY SEC (STARTING NEW PERFORMANCE).     */
          ISNOW=0 ,.     /* RESET 'SEC NO. WITHIN PERFORMANCE' COUNTER  */
          ELSE IF NS LE 13 THEN TITBIT='1'B ,.
          /* TITBIT='1'B INDICATES A NON-PLAY TITLED SECTION.            */
          ISNOW=ISNOW+1 ,.           /* COUNT SEC NO.'S WITHIN PERF.     */
          ITR=ISL+3 ,.               /* SET ITR TO POINT AT START OF SEC */
                                     /* FOLLOWING THE SECTYPE LETTER.    */
          IGR=ISL+2 ,.               /* IGR NOW PTS AT SECTYPE LETTER.   */
          IF NS GT 13 THEN GOTO NEXTGRP ,.   /* IF UNTITLED SECTION.     */
                                     /* PROCESS DATE, THEATRE, AND TITLE */
          BITSEC='1'B ,.             /* SPECIFY THAT THIS KIND OF SECTION
                                     /* HAS NO TIME ENTRIES.             */
          S=S CAT FC CAT SMOD ,.     /* TITLE GRP INDICATOR.             */
          ISC=INDEX(SUBSTR(P,ISL+1,ISR-ISL+1),'.  ') ,.
          ISC2=INDEX(SUBSTR(P,ISL+1,ISR-ISL+1),'!  ') ,.
          IF ISC2 GT 0 AND (ISC2 LT ISC OR ISC=0) THEN ISC=ISC2 ,.
          IF ISC=0 THEN DO ,.        /*   IF NO END OF TITLE DELIMITER  */
          CALL ERRMSG('NTTL',ISL+2) ,.
          ISC=ISR-ISL ,. /* NO TITLE, SET UP WHOLE SECTION=HEADER       */
          END ,.
          NONEXIT=NOITEM ,.          /* GIVE ERROR MESSAGE IF FIND NO ITM*/
          ISC=ISC+ISL ,.             /* NOW ACTUAL PTR TO END OF TITLE   */
          IF TITBIT THEN GOTO TITLE ,.       /* IF NOT A PLAY SECTION  */
          CALL GITM(ISC,'A') ,.      /* TAKE CARE OF POSSIBLE PAGE ENTRY */
          ITR=ITL ,.                 /* AND RESTORE STATUS FOR COGO      */
          BIT1='0'B ,.
          IDBITS=0 ,.
          DO IX=1 TO 4 ,.            /* THIS LOOP GETS UP TO 3 DATE PARTS*/
          CALL COGO(ITR,ISC,IPDATE(IX),DATEDONE) ,.
          BIT1='1'B ,.   /* INDICATES THAT AT LEAST ONE DATE ENTRY HAS */
                                                 /* BEEN FOUND.         */
          IF IX=1 THEN IDPTR=NBL/100 ,.        /* PTS TO LINE FOR DATE   */
          END ,.                               /* ENTRY LOGGING.         */
          CALL ERRMSG('XDAT',IDPTR*100+1) ,. /* '4TH PART OF DATE'       */
          IX=4 ,.          /* HAS BEEN FOUND, SO SIMULATE ONLY 3 PARTS.  */
     DATEDONE ..           /* DATE UPDATING, CHECKING, AND SETTING UP    */
          M=IX ,.          /* OF DATE SIGNIFICANCE BITS OCCURS BELOW.    */
          DO IX=1 TO IX-1 ,.
          M=M-1 ,.
          IF IPDATE(M)=0 THEN                  /* IF NON-SIGNIFICANT DATE*/
          DSARBIT(9-IX)='1'B ,.                /* PART THEN SET BIT TO 1 */
          ELSE
          IF IPDATE(M ) LT IDATCHK(IX,1) OR
             IPDATE(M ) GT IDATCHK(IX,2) THEN
```

```
            CALL ERRMSG('DPRT',IDPTR*100+1) ,. /* PART OF DATE IS OUT OF */
                                               /* PERMISSIBLE RANGE.     */
            ELSE            /* IF THIS PART IS INSIDE PERMISSIBLE RANGE.  */
            IRDATE(IX)=IPDATE(M) ,.  /* ACCEPT THIS PART OF DATE.         */
            END ,.
            IDATE=((IRDATE(3)-1659)*372+(IRDATE(2)-1)*31 +
                  IRDATE(1)-1)*8+IDBITS ,.     /* CONVERT NEW DATE TO     */
                           /* CODED DATE FOR DA KEY OR EQUIV.               */
            IF IDATE  LT IOYEAR OR (IDATE -IOYEAR GT 096 AND IOYEAR NE 0)
            THEN                               /* RELATIVE DATE CHECK     */
            CALL ERRMSG('DCHG',ITL) ,.         /* DATE ADVANCED TOO MUCH */
                                               /* OR WENT BACKWARDS.      */
            IOYEAR=IDATE ,.
            CALL GITM(ISC,' ') ,.    /* THEATRE AFTER A FULL DATE ENTRY. */
        THEATRE ..
            CTMP=C ,.       /* SAVE 8-BYTE THEATRE FOR RECORDED KEY STRNG */
        TITLE ..
            IF BIT1 THEN DO ,.         /* IF DATE WAS UPDATED IN THIS SEC  */
                                       /* THEN DO UPDATE FOR MAP/LOG.      */
            PUT STRING(MONTHS(IDPTR)) EDIT(IRDATE(2)) (F(2)) ,.
            DO K=IDPTR TO 36 ,.   IDAYS(K)=IRDATE(1) ,.   END ,.   END ,.
            CALL GITM(ISC,'.') ,.      /* GET TITLE OF SECTION             */
            L=INDEX(C,';') ,.          /* SEARCH FOR SEMICOLON, IF ANY,    */
                                       /* WHICH DEFINES START OF SUBTITLE, */
            IF L GT 1 THEN             /* AND IF FIND A GOOD ONE THEN      */
            C=SUBSTR(C,1,L-1) ,.       /* DELETE THE SUBTITLE.             */
            FGBIT,TTMBIT='0'B ,.       /* CTRL BITS FOR TITLE TIME ENTRIES */
            M=INDEX(C,': ') ,.         /* SRCH FOR END OF TITLE TIME ENTRY */
            IF M GT 1 THEN DO ,.       /* IF FOUND POSSIBLE TITLE TIME     */
            TITLTM=SUBSTR(C,1,M-1) ,.          /* GET TITLE TIME ENTRY.    */
            C=SUBSTR(C,M+2) ,.         /* THEN DELETE, LEAVING ONLY TITLE. */
            BITSEC,TTMBIT,FGBIT='1'B ,.   END ,.
            S=S CAT FA CAT SMOD CAT C ,.           /* PUT OUT TITLE ITEM   */
            IGR=ITR ,.       /* INITIALIZE OLD RIGHT GROUP POINTER         */
                             /* END OF DATE, THEATRE, & TITLE PROCESSING.  */
        NEXTGRP ..           /* TOP OF LOOP PROCESSING CAST LISTS.         */
            IGL=IGR ,.       /* OLD RIGHT GRP PTR = NEW LEFT GRP PTR.      */
                             /* AT THE START OF A SECTION IGL PTS AT THE   */
                             /* CHAR FOLLOWING THE SECTION TYPE LETTER.    */
            (STRG) ..        /* TEMP ERR CHECK, REMOVE IF YOU SEE THIS CARD*/
            IGR=INDEX(SUBSTR(P,IGL+1,ISR-IGL),';') ,.    /* SEMICOLON    */
            IF IGR NE 0 THEN GOTO VFRSTGRP ,.  /* IF SEMICOLON WAS FOUND   */
            IGR=ISR ,.                         /* SET GROUP-END = SECTION-END */
            IF VERIFY(SUBSTR(P,IGL+1,ISR-IGL),', ;:!?.+&()') NE 0
            THEN GOTO ISTGRP ,.      /*  GENERALLY, IF YOU FIND A LETTER */
                                     /* OR A NUMBER THEN GOTO ISTGRP      */
            /* AT THIS POINT THERE ARE APPARENTLY NO MORE GRPS IN THE     */
            /* SEC, SO IF NO GRPS AT ALL WERE IN THE SECTION, AND IF      */
            /* THIS SECTION HAS A TITLE TIME ENTRY THEN PUT IT OUT.       */
            IF FGBIT THEN S=S CAT FC CAT 'G' CAT FA CAT 'T' CAT TITLTM ,.
            GOTO NEXTSEC ,.
        VFRSTGRP .. IGR=IGR+IGL ,.   /* ACTUAL PTR TO END OF GRP DELIM     */
        ISTGRP ..
            S=S CAT FC CAT 'G' ,.    /* ANOTHER GROUP EXISTS, SO PROCESS   */
            IF BITSEC THEN GOTO PREROLE ,.     /* SKIP TIME ITEM IF AN
                                     /* AFTERPEICE OR PERFMANCE SECTION    */
            NONEXIT=NOITEM ,.        /* EVEN IF NO TIME ENTRY THERE MUST   */
                                     /* BE AN ITEM SINCE THERE IS A GRP    */
            ISC=ITR ,.       /* SAVE ITR FOR REUSE IF NO TIME ENTRY        */
            CALL GITM(IGR,' ') ,.    /* GET POTENTIAL TIME ENTRY (BLANK)   */
            L=LENGTH(C) ,.
            IF SUBSTR(C,L,1) NE ':' THEN DO ,. /* IF HAS NO COLON THEN    */
                  ITR=ISC ,. GOTO PREROLE ,.   /* IT'S NO TIME ENTRY, SO  */
                  END ,.                       /* SEE IF IT'S A ROLE      */
            K=VERIFY(C,'0123456789IVXabde') ,. /* LEGAL TIME ENTRY CHARS  */
            IF K NE L THEN CALL ERRMSG('UNVT',ITR) ,.    /* IF THE 1ST    */
```

```
                        /* 'ILLEGAL' CHAR IS NOT THE COLON THEN ERROR */
      S=S CAT '?T' CAT SUBSTR(C,1,L-1) ,.          /* ADD TIME      */
                                                   /* ENTRY TO BUFFER.  */
   PREROLE .. NONEXIT=SECNDGRP ,.
      IL=LENGTH(S) ,.              /* PATCH IN CASE OF NXTIT ACTION.   */
      IF TTMBIT THEN S=S CAT FA CAT 'T' CAT TITLTM ,.
      FGBIT='0'B ,.  /* INDICATES THAT THERE IS NO NEED TO PUT OUT */
                     /* A 'TITLE TIME ENTRY ITEM GRP' IN THE EVENT */
                     /* OF THE END OF THE SECTION BEING FOUND.     */
      ISC=INDEX(SUBSTR(P,IGL+1,IGR-IGL), '-') ,.   /* SEARCH MINUS */
      IF ISC=0 THEN GOTO SECNDGRP ,.     /* IF NO DASH THEN . . .    */
      ISC=ISC+IGL ,.               /* ACTUAL PTR TO END OF ROLE SUBGRP */
   ROLES ..
      CALL GTCK(ISC,',') ,.        /* GET A ROLE IF THERE IS ONE       */
      S=S CAT '?R' CAT C ,.        /* PUT OUT A SYNTACTIC ROLE ITEM.   */
      ITBIT=0 ,.       /* INDICATES AN ITEM HAS BEEN PUT OUT IN SEC    */
      GOTO ROLES ,.                            /* GET NEXT ROLE, IF ANY. */
   SECNDGRP ..                     /* ENTRY TO ACTOR PROCESSING LOOP.  */
      IL=LENGTH(S) ,.              /* SAVE CURRENT LENGTH BEFORE START */
                     /* OF ACTOR SUBGRP IN CASE OF A TO R CONVERT    */
                     /* DUE TO CAST GRP LADDER ENTRY (CGLE).         */
      NONEXIT=NEXTGRP ,.           /* WHERE TO GO WHEN NO MORE ACTORS. */
   ACTORS .. CALL GTCK(IGR,',') ,.          /* TRY TO GET AN ACTOR.    */
      S=S CAT '?A' CAT C ,.        /* PUT OUT A SYNTACTIC ACTOR ITEM.  */
      ITBIT=0 ,.       /* INDICATES AN ITEM HAS BEEN PUT OUT IN SEC    */
      GOTO ACTORS ,.                           /* GET NEXT ACTOR, IF ANY */
   NOTAPE ..           /* INTERRUPT SERVICE FOR NO MTST TAPE ENTRY.    */
         /* (ALWAYS OCCURS FOR SCANNER STUFF.)                         */
      ISR=INDEX(P,SD  ) ,.         /* SET UP PTR TO 1ST SECTION IN BLK */
      IF ISR=0 THEN CALL ERRMSG('NSEC',48) ,.      /* NO SECTION.      */
      GOTO ISTSEC ,.               /* CONTINUE PROCESSING BLK NORMALLY */
   NOITEM ..           /* INTERRUPT SERVICE FOR MISSING ITEM.          */
      CALL ERRMSG('NIT1',ITL) ,.
      CALL ERRMSG('NIT2',ITR) ,.
      GOTO NEXTSEC ,.
   GTCK .. PROC(LAT,DEL) ,.        /* GET AN ITEM, AND CHECK TO SEE IF */
      DCL DEL CHAR(1) ,.           /* IT'S A LADDER ENTRY OR NOT.      */
   TRY2 .. CALL GITM(LAT,DEL) ,.                /* REENTRY AFTER LADDER */
                                   /* ENTRY SUCCESSFULLY PROCESSED.    */
      L=LENGTH(C) ,.               /* SET UP FOR LENGTH TEST           */
      IF L LT 7 OR L GT 17 THEN RETURN ,.          /* TOO SHORT OR     */
                                   /* TOO LONG TO BE A LADDER ENTRY.   */
      C3='S' ,.
      IF CH4='See ' OR CH4='see ' THEN GOTO DATEREF ,.
      C3='L' ,.
      IF CH3 NE 'As ' THEN IF CH3 NE 'as ' THEN RETURN ,.
   DATEREF .. K=ITL+3 ,.
      CALL COGO(K,LAT,ILD(1),ENDGTCK) ,. /* TRY TO GET AND CONVERT */
                                                 /* THE DAY OF THE MONTH. */
      IF ILD(1) GT 31 THEN DO ,.
         CALL ERRMSG('DRNG',L) ,.   RETURN ,.   END ,.
      ITRSV=ITR ,.                 /* HERE DAY   IS KNOWN TO BE OK, SO */
      CSV=C ,.                     /* SAVE GITM'S ENVIRONMENT, AND     */
      EXITSV=NONEXIT ,.            /* THEN CALL IT - AFTER SETTING UP  */
      NONEXIT=NOMN ,.              /* FOR MISSING MONTH CASE.          */
      ITR=K ,.                     /* SET ITR FORWARD DUE TO COGO CALL */
      IPM=IPL+1 ,.                 /* CHECK ONLY WITHIN CURRENT ITEM.  */
      CALL GITM(IPM,' ') ,.        /* TRY FOR MONTH-WORD WITH DIM=BLNK */
      IF IOPT(9) THEN C=TRANSLATE(C,MONTHCAPS,MONTHSMALLS) ,.
      /* ABOVE STMTS ARE FOR   ANYCASE MONTH OPTION.                   */
      DO K=1 TO 12 ,.              /* LOOP FOR MATCHING MONTH NAME TO  */
                                   /* THE NUMBER OF THE MONTH.  JAN=1. */
      IF C=MONSTR(K) THEN GOTO CHEKYR ,.
      END ,.
      CALL ERRMSG('MNTH',ITL) ,.
      PUT FILE(SYSPRINT) EDIT(' ''',C,'''') (A) ,.
```

```
        GOTO GITOUT ,.
    CHEKYR .. ILD(2)=K ,.
        CALL COGO(ITR,IPM,ILD(3),NOYEAR) ,.
        IF ILD(3) GT 1800 OR ILD(3) LT 1659 THEN DO ,.
           CALL ERRMSG('BDYR',ITR) ,.   GOTO GITOUT ,.   END ,.
    PUTAS ..
        C='        ' ,.                          /* SET C TO LENGTH OF 8.  */
        PUT STRING(C) EDIT(ILD) (2 F(2),F(4)) ,.
        IF ITBIT=0 THEN DO ,.                    /* INDICATES CGLE.        */
        C3='C' ,.   NCGLE=NCGLE+1 ,. END,.       /* AND COUNTS CGLE'S.     */
        IF C3='L' THEN NLAD=NLAD+1 ,.            /* COUNTS 'AS' REFS.      */
        ELSE IF C3='S' THEN NSEE=NSEE+1 ,.       /* COUNTS 'SEE' REFS.     */
        S=S CAT FA CAT C3 CAT C ,.               /* PUT OUT LADDER ENTRY.  */
        ITR=ITRSV ,.   NONEXIT=EXITSV ,.         /* RESTORE GITM'S ENV,    */
                       /* EXCEPT FOR C, THAT EXISTED AFTER THE FIRST */
                       /* CALL TO GITM AFTER ENTRY TO GTCK (AFTER).  */
                       /* THIS SHOULD SET UP THE CALL FOR THE ITEM   */
                       /* FOLLOWING THE ENTIRE LADDER REFERENCE.     */
        IF ITBIT=0 THEN DO ,.    /* IF THIS IS A CAST GRP ENTRY THEN */
    NXTIT ..           /* SRCH FOR ANY SYNTACTIC ROLES PREVIOUSLY    */
                       /* THOUGHT TO HAVE BEEN SYNTACTIC ACTORS.     */
        IP=INDEX(SUBSTR(S,IL),'?A') ,.
        IF IP GT 0 THEN DO ,.    /* IF YOU FIND ONE THEN . . . . . */
           SUBSTR(S,IL+IP,1)='R' ,.              /* CORRECT THE TAG.       */
           GOTO NXTIT ,.   END ,.
        GOTO SECNDGRP ,.         /* ONLY ACTORS REMAIN, ROLES DONE.  */
        END ,.
        GOTO TRY2 ,.
    NOYEAR .. ILD(3)=IRDATE(3) ,.   GOTO PUTAS ,.
    NOMN ..
        CALL ERRMSG ('NOMN',ITL) ,.
    GITOUT ..
        C=CSV ,.   ITR=ITRSV ,.   NONEXIT=EXITSV ,.   RETURN ,.
    ENDGTCK .. END GTCK ,.
    GITM .. PROC(LST,DLM) ,.
        /* ON NORMAL EXIT, ITL POINTS TO LEFT CHAR OF ITEM, ITR PTS  */
        /* TO THE NEXT CANDIDATE FOR A GROUP OF ITEM TO BE SEARCHED. */
        DCL LST FIXED BIN(15,0), /* LST PTS TO AN OUT OF RANGE DELIM */
        DLM CHAR(1) ,.           /* DELIMITS ITEM SOUGHT, UNLESS THE */
                                 /* ITEM IS THE LAST ONE IN RANGE.   */
        C='ERRONEOUS ITEM' ,.    /* SHOULD SHOW IN CERTAIN ERRORS.   */
    RENT ..            /* RE-ENTRY POINT AFTER PAGE ENTRY PROCESSING */
                       /* ITR POINTS TO THE CHARACTER WHICH IS THE   */
                       /* 1ST CANDIDATE FOR START OF ITEM.           */
        IF IOPT(20) THEN PUT FILE(SYSPRINT) EDIT(LST,' ''',DLM,'''')
        (F(4), 3 A) SKIP ,.
        IF ITR+2 GE LST THEN GOTO NONEXIT ,.          /*ITEM TOO SMALL*/
        L=VERIFY(SUBSTR(P,ITR  ,LST-ITR  ),', ;.!?-:') ,.
                       /* ABOVE STATEMENT SKIPS PAST PUNCTUATION     */
                       /* FOLLOWING LAST ITEM PROCESSED.  PURPOSE IS */
                       /* TO SEARCH FOR BEST CANDIDATE FOR LEFTMOST  */
                       /* CHARACTER OF CURRENTLY SOUGHT ITEM.        */
                       /* PUNCTUATION INCLUDES COMMA, BLANK, PERIOD, */
                       /* EXCLAMATION POINT, MINUS SIGN, QUESTION    */
                       /* MARK, AND COLON.                           */
        IF L=0 THEN GOTO NONEXIT ,.      /* IF NO PROSPECTIVE ITEMS*/
        ITL=ITR+L-1 ,. /* POINTS TO LEFTMOST CHARACTER OF ITEM       */
        ITR=INDEX(SUBSTR(P,ITL+1,LST-ITL-1),DLM) ,.
        IF ITR=0 THEN ITR=LST-1 ,.               /* POINTS TO LAST CHAR    */
        ELSE ITR=ITR+ITL ,.      /* POINTS TO DELIM FOLLOWING ITEM   */
        IPL=ITR ,.               /* POINTS TO DELIM OR LAST CHARACTER*/
            /* IF NO EXPLICIT DELIMITER IS FOUND IN RANGE THEN ITR  */
            /* PTS TO THE LAST CHAR OF THE ITEM, ELSE ITR PTS TO    */
            /* THE DELIMITER ITSELF AT THIS TIME.                   */
    ITREACK .. C1=SUBSTR(P,IPL,1) ,.
        IF C1=' ' OR C1='.' OR C1=',' OR C1=DLM THEN
```

```
            DO ,.          /* CHOP TRAILING BLANKS, PERIODS, & COMMAS    */
          IPL=IPL-1 ,.     /* BY MOVING BACK PTR, AND THEN GOING BACK TO */
          GOTO ITRBACK ,.    END ,. /* SEE IF ANY MORE TO CHOP.           */
          IF IPL-ITL LT 0 THEN DO ,.          /* IF CHOPPED TOO MANY.    */
             CALL ERRMSG('NLI1',ITL) ,.       /* 'NEGAITVE LENGTH ITEM'  */
              CALL ERRMSG('NLI2',IPL-ITL+1) ,.   GOTO ENDGITM ,.   END ,.
          IF SUBSTR(P,IPL-3,4)=' but' THEN    /* TAKE OUT TRAILING BUTS  */
          DO ,.    IPL=IPL-4 ,.    GOTO ITRBACK ,.    END ,.
          IF SUBSTR(P,ITL-1,2) = ' p' THEN
             DO ,.                            /* CHECK FOR PAGE CHANGE   */
          L=ITL/100 ,.     /* COMPUTE AND SAVE LINE NUMBER FOR MAP       */
          K=ITL+1 ,.       /* SET UP NON-DUMMY PARM FOR PAGE NUMBER
                           /* CONVERSION ATTEMPT, TO SAVE RETURN VALUE   */
          CALL COGO(K   ,  IPL+1,N   ,NOPAGE) ,.
          IF N LT IPAG OR IPAG GT N+1 THEN CALL ERRMSG('SUSP',ITL) ,.
          IPAG=N ,.        /* NEW PAGE FOUND AND CONVERTED SUCCESSFULLY  */
          ITR=K ,.         /* SET UP ITR TO GET ITEM AFTER PAGE ENTRY    */
          PUT STRING(CHPAGE) EDIT(IPAG) (F(4)) ,.
          IPAGES(L)=CHPAGE ,.
          S=S CAT FA CAT 'B' CAT CHPAGE ,.   /* ENTRY TO STRUCT FILE     */
          NOTES(L)=NOTES(L) CAT '*' ,.
          IF IOPT(6) THEN
          PUT FILE(SYSPRINT) SKIP(2) EDIT
          ('*** NEW PAGE ',IPAG,' ') (COL(21),A,F(4),SKIP(1),A) ,.
          GOTO RENT ,.                       /*  TRY FOR ITEM AGAIN     */
       NOPAGE .. END ,.
          IF IPL-ITL GT 3 THEN DO ,.         /* CHOP LEADING BUTS       */
          CH4=SUBSTR(P,ITL,4) ,.
          IF CH4='But ' OR CH4='but ' THEN DO ,. /* CAPITAL & SMALL     */
          ITR=ITL+4 ,.
          GOTO RENT ,.
          END ,.   END ,.
          C=SUBSTR(P,ITL,IPL-ITL+1) ,.       /* DELIVER ALL ITEMS IN C */
          IF DLM NE ' ' THEN ITR=ITR+1 ,.    /* MOVE PTR PAST DELIMITER*/
          IF IOPT(21) THEN PUT FILE(SYSPRINT) EDIT
          (' ''',C,'''') (A) ,.
       ENDGITM .. END GITM ,.
       COGO .. PROC(IPTR,IRPT,IVAL,WGO) ,.   /* SEARCH FOR AND ATTEMPT */
                           /* TO CONVERT A NUMBER TO INTERNAL BINARY,.   */
                           /* IN CALLING SEQUENCE IPTR PTS TO FIRST      */
                           /* POSSIBLE CHAR OF NUMBER.  IRPT PTS TO ONE  */
                           /* CHAR PAST THE POSSIBLE END OF THE NUMBER.  */
                           /* EXIT VIA WGO IF NO FIND OR BAD CONVERT.    */
          DCL WGO LABEL ,.
          IRNG=MIN(IRPT-IPTR  ,20) ,.
          IF IRNG LT 1 THEN GOTO WGO ,.      /* STRING TOO SMALL.       */
          CHOGO=SUBSTR(P,IPTR  ,IRNG) ,.
          DO NB=1 TO IRNG ,.          /* SRCH FORWARD FOR 1ST NONBLANK.  */
          IF CHOGA(NB) NE ' ' THEN GOTO STST ,.
          END ,.
          GOTO WGO ,.    /* SINCE NOTHING BUT BLANKS IN RANGE.           */
       STST .. NUM,MNUM=0 ,.
          NBL=NB ,.
          DO NN=NB TO IRNG ,.
          C2=CHOGA(NN) ,.
          IF C2 LT '0' OR C2 GT '9' THEN GOTO CHEKOUT ,.
          MNUM=MNUM-240 ,.
          NUM=NUM*10+MNUM ,.
          END ,.
       CHEKOUT .. IF NN=NB THEN GOTO WGO ,.  /* IF 1ST NONBLNK CHAR IS */
                                             /* NOT A DIGIT.           */
          IVAL=NUM ,.    /* PUT CONVERTED VALUE INTO PARM FOR RETURN.   */
          IPTR=IPTR+NN-1 ,.          /* NOW PTS TO 1ST CHAR AFTER DIGITS */
          END COGO ,.
    ERRMSG .. PROCEDURE(ABBRV,LOC) ,.          /* ERROR MESSAGE ROUTINE  */
                                    /* SEE NOTE AT TOP OF LISTING.      */
```

```
            DCL ABBRV CHAR(4),
            LOC FIXED BIN(15,0) ,.
            ERRORS(LOC/100)=ABBRV ,.
            IERR=MAX(1,LOC-49) ,.     /* AVOIDING STRG PROBLEM.          */
            (STRG) .. PUT FILE(SYSPRINT) SKIP(2) EDIT
            ('ERROR NOTE ***** ',ABBRV,' AT',LOC,' ''',
            SUBSTR(P,IERR,MIN(100,I-IERR+1)),'''')
            (3 A,F(5),3 A) ,.
            IF IOPT(27) THEN ERR='1'B ,.      /* FOR OTHER ERR MSG'S.    */
            IERRNO=IERRNO+1 ,.        /* FOR ERROR STATISTICS.           */
            END ERRMSG ,.
        ENDPROG ..
            PUT FILE(SYSPRINT) SKIP(2) EDIT
            ('MAX BLKSIZE=',MAXOUT,' ERRMSG ERRORS=',IERRNO,' TRKS=',
            KSTRU,' OUTPUT CHARS=',NOUT,' ''AS'' REFS=',NLAD,
            ' CGLE''S=',NCGLE, ' SEE REFS=',NSEE)
            (8 A,SKIP) ,.
            END ,.
    /*
    //LADDER   JOB 99999211,WCD,MSGLEVEL=1
    //L EXEC LSPUPGEN,MEM=LADDR,
    // PARM.PL1L='C48,NT,A,X,FW,OPT=0,SIZE=88K,E,LC=62',
    // PARM.LKED='MAP,LIST'
    //BAKUP.SYSUT1  DD *,DCB=BLKSIZE=80
            /* LADDER PROGRAM */
            /* TLE = 'TITLE LADDER ENTRY'.                               */
            /* SLE ='SEE TYPE LADDER ENTRY'.                             */
            /* CGLE = 'CAST GRP LADDER ENTRY.'                           */
        LADDER .. PROC OPTIONS(MAIN) ,.
            DCL
               /* THE NEXT 8 ARRAYS FORM THE GROUP CONTROL BLOCKS (GCB'S)*/
               GBIT(64) BIT(1),       /* '1'B INDICATES GRP DELETE FCN   */
               GTM(064) FIXED BIN(15,0),      /* PTS TO TIME ENTRY ICB,  */
                                      /* OR IS 0 IF NO TIME ENTRY IN GRP. */
               GR (064) FIXED BIN(15,0),      /* PTS TO ROLE ICE CHAIN   */
               GA (064) FIXED BIN(15,0),      /* PTS TO ACTOR ICE CHAIN  */
               GRN(064) FIXED BIN(15,0),      /* COUNT OF ROLES IN CHAIN*/
               GAN(064) FIXED BIN(15,0),      /* ACTOR COUNT IN CHAIN    */
               GCG(064) FIXED BIN(15,0),      /* CAST GRP ENTRY OR ZERO  */
               GF (064) FIXED BIN(15,0),      /* FORWARD CHAIN PTR       */
            IGPSIZ INITIAL(064) STATIC,       /* SIZE OF GCB ARRAYS.     */
               /* THE NEXT 5 ARRAYS FORM THE ITEM CONTROL BLOCKS (ICB'S) */
            ITMSIZ INITIAL(128) STATIC,       /* SIZE OF ICB ARRAYS.     */
               IPBIT(128) BIT(1),     /* ITEM ADD INDICATOR              */
               IMBIT(128) BIT(1),     /* ITEM DELETE INDICATOR           */
               IP   (128),            /* ACTUAL PTRS TO ACTUAL ITEMS     */
               IL   (128),            /* ACTUAL LENGTHS OF ACTUAL ITEMS  */
               ILK  (128) ,.          /* ITEM CONTROL BLK CHAIN LINK WRDS */
            DCL              /* DCL STMT FOR FILE STORAGE & SEARCHES      */
            ERR BIT(1),              /* ERROR INDICATOR BIT FOR SECTION.  */
            IOLDATE FIXED BIN(31,0), /* OLDKEY BIT(16) BELOW, IS          */
                                     /* DEFINED ON ICLDATE POS(3).        */
            OLDKEY BIT(16) DEF IOLDATE POS(3),
            IDTRK(000 .. 371),       /* DAY OF YEAR TO TRK MAP.           */
                                     /* MAP.  GIVES 1ST TRACK TO SEARCH   */
                                     /* FOR ANY SECTION WITH A CERTAIN    */
                                     /* DATE.  ELEMENT=-1 IF NO SUCH      */
                                     /* DATE IS IN THE LADA FILE.         */
            NEWDATE FIXED BIN(31,0) INITIAL(0) STATIC,
            CRNTKEY BIT(16) DEF NEWDATE POS(3),
            LADA ENV(REGIONAL(3)) KEYED DIRECT,
            SKEY CHAR(12), /* RECORDED PART OF SEARCH KEY FOR LADA BLKS   */
            REG8 CHAR(8),  /* TRACK NUMBER IN CHAR STRING FORM FOR SRCH   */
                           /* KEY IN LADA FILE SEARCH.                    */
            RLAD,                    /* CURRENT TRACK FOR SEQUENTIAL      */
                                     /* LADA OUTPUT.                      */
```

```
NOUT FIXED BIN(31,0) STATIC INIT(0),          /* TOTAL OUTPUT */
                                              /* CHARACTERS.  */
RL FIXED BIN(15,0) STATIC,
SP INITIAL(3625) STATIC  /* AVAILABLE SPACE ON CURRENT LADA  */
   FIXED BIN(15,0) ,.    /* SEQUENTIAL OUTPUT TRACK.         */
DCL (          /* DCL STMT FOR CONSTANTS ETC.               */
FA CHAR(1) INITIAL('?'),          /* ITEM-START DELIMITER    */
FC CHAR(1) INITIAL('?')) STATIC,  /* GROUP-START DELIMITER   */
STYPE(26) CHAR(1) STATIC INITIAL('p','a','b','i','o','t','u',
'p','p','p','p','p','p', 'd','e','m','s','p','p','p','p',
'p','p','p','p','p'),    /* SECTION TYPE LETTERS.          */
SYSPRINT PRINT ENV(F(133,133)),
STRU ENV(REGIONAL(3)) INPUT KEYED, /* STRUCTURED INPUT FILE  */
KEYS ENTRY(FIXED BIN(15,0), FIXED BIN(15,0)),
ERRMSG ENTRY(CHAR(4),FIXED BIN(15,0)) ,.
DCL            /* DCL STMT FOR WRK & CTRL VARIABLES         */
C1 CHAR(1),                   /* TEMP ONLY                  */
RBITS(3) BIT(1),              /* RETURN    BITS FOR KEY SEARCHES. */
SBITS(3) BIT(1),              /* SELECTION BITS FOR KEY SEARCHES  */
INTRCHN INIT(1) STATIC,       /* POST-CGLE RESTORE & RELINK SWITCH
                              /* INITIALLY SET TO 'NO RESTORE'. */
SBITE BIT(3) DEF SBITS,       /* EFFICIENT ASSIGNMENT OVERLAY   */
EMSG CHAR(4),                 /* AVOIDS PASSING PARM IN SUBR CALL */
IOPT(32) BIT(1),              /* OPTION BITS, STATIC AFTER START  */
                              /* OF PROGRAM.                */
GTYP CHAR(2),                 /* CURRENT GRP DIM CONTROL CHAR.  */
GTYP2 CHAR(1) DEF GTYP POS(2),     /* GRP-TYPE CHAR.          */
ITYP CHAR(2),                 /* CURRENT ITEM START CTRL CHARS. */
ITYP1 CHAR(1) DEF ITYP,            /* FOR PGM LOGIC ERR CHK. */
ITYP2 CHAR(1) DEF ITYP POS(2),     /* CURRENT ITEM TYPE CHAR */
S CHAR(1800) VAR,             /* PRIMARY INPUT BUF, & WRK BUFFER  */
Z CHAR(1800) VAR,             /* LADA SEARCH INPUT BUF, AND MAIN  */
                              /* OUTPUT BUFFER, TEMP INPUT BUFR.  */
CRTITL CHAR(83 ) VAR,         /* CURRENT OPERA, PLAY, OR AFTER-   */
                              /* PEICE TITLE.                     */
CTMP CHAR(83 ) VAR,           /* TITLE COMPARISON TEMP,           */
                              /* FOR LADDER SEARCHES.             */
COMPTMP CHAR(40 ) VAR,        /* ROLE, ACTOR, AND TIME ENTRY      */
                              /* COMPARISON TEMP FOR KEYS CALLS.  */
GCHK BIT(1),   /* TEMP USED ONLY IN RASH IN KEYS                  */
LBIT BIT(1),   /* LOCKOUT BIT FOR MORE THAN ONE LADDER ENTRY      */
/* KNTIME,     /* COUNTS TIME ENTRIES WITHIN EACH GRP, TO         */
               /* DETECT & PREVENT MULTIPLE TIME ENTRIES          */
STBT(26) BIT(1),              /* SECTION TRACK BITS FOR REPEATED  */
                              /* NON-PLAY TITLED SECTIONS ON TRK  */
TITBIT BIT(1),                /* '1'B = 'NONPLAY TITLED SECTION'. */
LASTRK STATIC INIT(0),        /* FOR DIRECT ACCESS SPACE STATS.   */
IERRNO STATIC INIT(0),        /* FOR ERROR COUNT STATISTICS.      */
IGPMAX STATIC INIT(0),        /* MAX GCB'S USED STATISTIC.        */
ITPMAX STATIC INIT(0),        /* MAX ICB'S USED STATISTIC.        */
MAXOUT INIT(0) STATIC,        /* FOR MAX OUTPUT BLKSIZE STATISTICS*/
(NOPT INITIAL(1),             /* OPTION NUMBER CHECKER, NEW-OPT.  */
J INITIAL(0)) STATIC ,.       /* INPUT RECORD COUNTER             */
DCL            /* DCL STMT FOR KEY & DATE MANIPULATION ETC.  */
IDATE FIXED BIN(31,0),        /* FOR CODED DATE CONVERSION OVERLAY*/
CHIDATE CHAR(3) DEF IDATE POS(2),
STRUKEY CHAR(12),             /* INPUT BUF FOR KEY IN STRU READS  */
FREFCHK CHAR(3) DEF STRUKEY,       /* FORWARD REFERENCE CHK. */
SMOD CHAR(1) DEFINED STRUKEY POS(4),          /* SECTION TYPE */
R9OFKEY CHAR(9) DEFINED STRUKEY POS(4),
STRUDEF BIT(24) DEF STRUKEY,
TRANSBITS BIT(24) INITIAL('111111111111111111111111'B) STATIC,
KEYBITS BIT(16) DEF TRANSBITS POS(6) ,.
ON ERROR SNAP BEGIN ,.
PUT FILE(SYSPRINT) SKIP(1) EDIT('ERROR ONCODE=',ONCODE)
(A,F(4)) ,.   GOTO PROGEND ,.   END ,.
```

```
          ON STRG SNAP PUT EDIT(IGL,IGP,ITP,GCG,S,IXX,Z,I) (A) ,.
          ON KEY(LADA) BEGIN ,.
          IF ONCODE=51 THEN DO ,.
             CALL ERRMSG('KRNF',M-L+1) ,.    /* KEYED RECORD NOT FOUND */
             GOTO NOMATCH ,.   END ,.        /* ERR - SEC NOT FOUND.   */
          PUT SKIP(3) EDIT('KEY ONCODE=',ONCODE) (A) ,./* BAD ERR.     */
          END ,.
          ON ENDFILE(STRS) GOTO PROGEND ,.
          GOTO ISTSEASON ,.          /* AVOID SEASON CHANGE MESSAGE.    */
       SEASON ..             /* TOP OF SEASON LOOP.                     */
          PUT SKIP(6) EDIT('SEASON CHANGE $$$$$$$$$ TRK=',KLAD) (A) ,.
          IF IOPT(2) THEN ISTSEASON .. DO ,.
          OPEN FILE(LADA) KEYED DIRECT OUTPUT ,.       /* FORMAT LADA  */
          CLOSE FILE(LADA) ,.
          OPEN FILE(LADA) KEYED DIRECT UPDATE ,.
          KLAD=0 ,.   END ,.         /* RESET OUTPUT TRK PTR TO START.  */
          IDTRK=-1 ,.     /* CLEAR DATE/TRACK MAP FOR PREVIOUS SEASON.  */
          TRANSBITS='11111111111111111111111'B ,.     /* ASSURES NEW   */
               /* IDTRK ENTRY DUE TO APPARENT DATE CHANGE AT START OF   */
               /* EVERY SEASON, INCLUDING THE FIRST.                    */
       INPUT ..              /* TOP OF SECTION LOOP.                    */
          ERR='0'B ,.     /* RESET BLK ERR INDICATOR TO 'NO ERRORS YET' */
          DO IGP=1 TO IGPSIZ ,.   GF(IGP)=IGP+1 ,.     /* CLEAR PARTS  */
             GCG(IGP)=0 ,.   GBIT(IGP)='0'B ,.  END ,. /* OF GCB'S.    */
          DO ITP=1 TO ITMSIZ ,.   ILK(ITP)=ITP+1 ,.   END ,.
          OLDKEY=KEYBITS ,.          /* EXCEPT FOR 1ST READ, IOLDATE HAS */
                                     /* 'PREVIOUS' KEY IN LOWER 16 BITS. */
          IF J=NOPT-1 THEN GET FILE(SYSIN) EDIT(IOPT,NOPT,C1)
          (32 B(1),F(6),X(41),A(1)) ,.       /* OPTION CARD CHAIN.     */
          IF IOPT(6) THEN DO ,.     /* UNKEYED SEQL FORMAT INPUT OPTION */
          READ FILE(STRS) INTO(Z) ,.
          STRUKEY=SUBSTR(Z,3,12) ,.          /* EXTRACT 'KEY'.         */
          S=SUBSTR(Z,15) ,.   END ,.         /* REMOVE 'KEY' & BLANKS. */
          ELSE                       /* DA KEYED SEQL INPUT.           */
          READ FILE(STRU) INTO(S) KEYTO(STRUKEY) ,.
          J=J+1 ,.                   /* STRU INPUT BLK CTR             */
          IF IOPT(3) THEN GOTO INPUT ,.      /* SKIP INPUT BLKS OPT.   */
          IF STRUKEY='XXXXXXXXXXXX' THEN DO ,.
          J=J-1 ,.                   /* DON'T COUNT IT AS A SECTION.   */
          CLOSE FILE(LADA) ,.   GOTO SEASON ,.   END ,.
          I,IF=LENGTH(S) ,.          /* 'CURRENT ACCESSIBLE LENGTH' =  */
                                     /* CURRENT ACTUAL LENGTH.         */
          ISEE=I ,.                  /* BOUNDARY FOR FIRM GRPS.        */
          TITBIT='0'B ,.             /* 'NOT NON-PLAY TITLED SECTION'. */
          DO NS=1 TO 26 ,.           /* INVESTIGATE SECTYPE & PROPERTIES */
          IF STYPE(NS)=SMOD THEN GOTO SECFOUND ,.   END ,.
          CALL ERRMSG('ISEC',J) ,.   GOTO INPUT ,.     /* ILLEGAL SEC. */
       SECFOUND ..IF NS=1 THEN STBT='0'B ,.   /* ALL BITS = 0 IF PLAY   */
          ELSE DO ,.   IF NS LE 13 THEN TITBIT='1'B ,.   END ,.
          /* TITBIT='1'B INDICATES A NON-PLAY TITLED SECTION.           */
          TRANSBITS=STRUDEF ,.       /* KEYBITS NOW CONTAINS THE CODED */
                                     /* DATE FROM STRUKEY (SEE DECLARE */
                                     /* STATEMENTS FOR SIY DEFINING).  */
          CRNTKEY=KEYBITS ,.         /* NEWDATE NOW HAS NEW CODED DATE */
          IPBIT,IMBIT='0'B ,.
          LBIT='0'B ,.               /* REPEATED TITLE LADDER LOCKOUT  */
          CRTITL='' ,.               /* INDICATE NO TITLES DONE YET IN */
                                     /* THIS SECTION.  (PLAY, OPERA, OR */
                                     /* AFTERPEICE TITLES, THAT IS).   */
          IS,              /* SECOND LENGTH, LENGTH AFTER LADDER READ IN */
          IGP2,IGP3,IGP4=0 ,.        /* INITIAL GCB CHAINS' HEAD & TAILS */
          IXX=0 ,.                   /* INDEX TO CHK GCG'S TO SEE IF   */
                                     /* THEY REFER TO A CGLE THAT HAS  */
                                     /* BEEN SUCCESSFULLY READ INTO CORE.*/
          IGR,             /* INITIAL RIGHT GRP PTR                     */
          IGP,             /* INITIALIZE GCB PTR                        */
```

```
        ITP=1 ,.        /* INITIALIZE ICB PTR                          */
        IF I LE 5 THEN GOTO FORMOUT ,.      /* SINCE REQUIRES NO       */
                                            /* FURTHER PROCESSING.     */
        S=S CAT FC ,.                 /* I NOW=LENGTH(S)-1.            */
        GOTO ISTGRP ,.                /* ENTRY TO GRP LOOP.            */
    GROUPS ..                         /* TOP OF GRP LOOP.              */
        IF IGR=I+1 THEN DO ,.         /* IF END OF SOME BLK OR BLK PART*/
        IF I=IS THEN DO ,.            /* IF END OF TLE THEN DO ,.      */
                IF IGP GT IGP2 THEN DO ,.  /* IF AT LEAST 1 TLE GRP. */
                   IGP3=IGP2+1 ,.   IGP4=IGP ,.   END ,.
                END ,.                /* NOW GO & CHK FOR CGLE'S.      */
                ELSE IF IXX=0 THEN DO ,.   /* END OF REFERRING GRP.  */
                IGP2=IGP ,.
                IF IS GT I THEN DO ,.
                   I=IS ,.   GOTO CONTGRPS ,.   END ,.   END ,.
        DO IXX=IXX+1 TO IGPSIZ ,.
        IF GCG(IXX) NE 0 THEN DO ,.         /* IF THIS GRP HAS READ A */
                          /* CGLE SUCCESSFULLY INTO CORE FCB SETUP.    */
        I=GCG(IXX) ,.   GCG(IXX)=IGP+1 ,.
        GOTO CONTGRPS ,.   END ,.   END ,.
        IGPMAX=MAX(IGPMAX,IGP) ,.           /* MAX GCB'S USED IN RUN. */
        ITPMAX=MAX(ITPMAX,ITP) ,.           /* MAX ICB'S USED IN RUN. */
        GOTO OUTPUT ,. /* ALL (ALL) INPUT/SETUP DONE, NOW PROCESS.     */
        END ,.
    CONTGRPS ..
        IF IGP GE IGPSIZ THEN DO ,.         /* CHECK OVERFLOW USAGE OF*/
                                            /* GROUP CONTROL BLOCKS.  */
        CALL ERRMSG('NGCB',IGR) ,.   IGR=I+1 ,.   GOTO GROUPS ,.
        END ,.
        IGP=IGP+1 ,.   /* SET PTR TO NEXT FREE GCB.                    */
    ISTGRP ..          /* ENTRY TO GRP LOOP FROM SECTION LOOP.         */
        KNTIME=0 ,.    /* CLEAR MULTIPLE TIME ENTRY DETECTOR/COUNTER */
        IGL=IGR ,.     /* LAST BYTE+1 OF PREVIOUS GRP = LEFTMOST       */
                       /* BYTE OF CURRENT/NEW GROUP, IF ANY.           */
        GR(IGP),GTM(IGP),GA(IGP),GRN(IGP),GAN(IGP)=0 ,.
        IGR=INDEX(SUBSTR(S,IGL+1),FC)+IGL ,.          /* FIND GRP-END */
        IF IGR-IGL LE 2 THEN DO ,.          /* GROUP TOO SHORT.       */
                          CALL ERRMSG('0GRP',IGL) ,.   IGP=IGP-1 ,.
                          GOTO GROUPS ,.   END ,.
        IF SUBSTR(S,IGR-1,1)='_' THEN       GRIT(IGP)='1'B ,.
        GTYP=SUBSTR(S,IGL,2) ,.  /* GET GRP DELIM CTRL CHARS.          */
        ITR=IGL+2 ,.             /* RESET ITEM PTR.                    */
    ITEMS ..
        IF ITR=IGR THEN DO ,.    /* IF LAST ITEM OF GRP IS DONE.       */
        IF GAN(IGP)+GRN(IGP)+GTM(IGP)=0 THEN IGP=IGP-1 ,.
        GOTO GROUPS ,. /* THE ABOVE TEST IS FOR GRPS THAT HAVE NO      */
        END ,.         /* SIGNIFICANT ITEMS.  TITLE GRPS USUALLY.      */
        ITL=ITR ,.
        ITYP=SUBSTR(S,ITL,2) ,.
        IF ITYP1 NE FA THEN DO ,.
        CALL ERRMSG('NEFA',ITL) ,.
        PUT EDIT(S  ,'''',Z,IGP,ITP) (A) ,.   END ,.
        ITR=INDEX(SUBSTR(S,ITL+1,IGR-ITL  ),FA) ,.
        IF ITR=0 THEN ITR=IGR ,.   ELSE ITR=ITR+ITL ,.
        IP(ITP)=ITL+2 ,.         /* SAVE PTR TO ITEM                   */
        IL(ITP)=ITR-ITL-2 ,.     /* SAVE LENGTH OF ITEM                */
        C1=SUBSTR(S,ITL+2,1) ,.  /* GET 1ST CHAR OF ITEM.              */
        IMBIT(ITP)=(C1='_') ,.   IPBIT(ITP)=(C1='+') ,.
        IF IMBIT(ITP) OR IPBIT(ITP) THEN DO ,.      /* IF SIGNED      */
           IP(ITP)=IP(ITP)+1 ,.  /* INCREMENT PTR PAST SIGN            */
           IL(ITP)=IL(ITP)-1 ,.  /* AND DECREASE LENGTH BY ONE         */
           END ,.
        IF SUBSTR(S,IP(ITP)+IL(ITP)-1,1)='_' THEN IL(ITP)=IL(ITP)-1 ,.
                          /* ABOVE STMT TAKES CARE OF TRAILING DASH,    */
                          /* PRESUMABLY DUE TO GRP DELETE FUNCTION.     */
        IF ITYP2='A' THEN DO ,.             /*$$$$$$$$$ ACTOR $$$$$$$$*/
```

```
            GAN(IGP)=GAN(IGP)+1 ,.                    /* ADD 1 TO GRP ACTOR CNT */
            IF GAN(IGP)=1 THEN GA(IGP)=ITP ,.  /* PTR TO PTR TO 1ST      */
                                       /* ACTOR IN A GRP IS NOW SET UP.    */
            GOTO ITEMDONE ,.   END ,.
            IF ITYP2='R' THEN DO ,.                   /*$$$$$$$$$ ROLE $$$$$$$$$*/
            GRN(IGP)=GRN(IGP)+1 ,.                    /* ADD 1 TO GRP ROLE CNT  */
            IF GRN(IGP)=1 THEN GR(IGP)=ITP ,.  /* IF 1ST ROLE IN THE GRP */
            GOTO ITEMDONE ,.   END ,.
            DO M=1 TO 13 ,.                /* FOR ALL TITLED SECTION LETTERS.  */
            IF ITYP2=STYPE(M) THEN DO ,.              /*$$$$$$$$ TITLE $$$$$$$$$*/
            IF CRTITL='' THEN                         /* IF NO PREVIOUS TITLE   */
               CRTITL=SUBSTR(S,IP(ITP),IL(ITP)) ,.   GOTO ITEMS ,.   END ,.
            END ,.                         /* END OF TITLE TITLE PROCESSING.   */
            IF ITYP2='L' OR                           /*$$$$$$$$ 'AS' GRP $$$$$$$*/
               ITYP2='C' OR                           /*$$$$$$$$ 'CGLE' GRP $$$$$*/
               ITYP2='S' THEN DO ,.                   /*$$$$$$$$ 'SEE' GRP $$$$$$*/
            IF LBIT THEN IF ITYP2 NE 'C' THEN DO ,.
               CALL ERRMSG('XLDR',ITL) ,.   GOTO ITEMS ,.   END ,.
            IF ITYP2 NE 'C' THEN     /* EXCEPT FOR CGLE'S.                    */
            LBIT='1'B ,.   /* LOCKOUT MORE THAN ONE TITLE LADDER REF.         */
            Z=SUBSTR(S,IP(ITP),IL(ITP)) ,.            /* TEMP FOR 3 USES SOON.  */
            GET STRING(Z) EDIT
            (IDAY,IMONTH,IYEAR) (2 F(2),F(4)) ,.              /* CONVERTS     */
                                       /* FULL LADDER REF DATE TO BINARY.    */
            IMONTH=IMONTH-1 ,.   IDAY=IDAY-1 ,.               /* SAVES CALCS. */
            IDATE=((IYEAR-1659)*372+IMONTH*31+IDAY)*8 ,. /* SET UP          */
                                       /* ENCODED DATE FOR SRCH KEY.         */
            SKEY=CHIDATE CAT R9OFKEY ,.               /* SET UP RECORDED KEY.   */
            M=IMONTH*31+IDAY ,.            /* GET ADDRESS OF IDTRK ENTRY FOR   */
                                       /* DATE OF LADDER REF SOUGHT.         */
            L=IDTRK(M) ,.  /* NOW L = 1ST TRACK TO SEARCH FOR THIS BLK       */
            IF L=-1 THEN DO ,.             /* IF NO SECTION AT ALL ON THAT     */
                                       /* DATE IS ON THE DISK THEN . . . . */
            CALL ERRMSG('NTRK',IDAY+1) ,.
            PUT EDIT(Z,SMOD,CRTITL) (3 (X(2),A)) ,.
            GOTO ITEMS ,.   END ,.
            ITER=0 ,.      /* ITERATION FACTOR FOR EXTRA READ ATTEMPTS        */
               IF CHIDATE GE FREFCHK THEN DO ,.
                  CALL ERRMSG('FREF',ITL) ,.   /* FORWARD LADDER REFERNC */
                  PUT FILE(SYSPRINT) SKIP(1) EDIT
                  (Z,SMOD,CRTITL) (3 (X(2),A)) ,.
                  GOTO ITEMS ,.   END ,.
            IF TITBIT THEN DO ,.           /* FOR NON-PLAY TITLED SECTIONS.   */
         GETITER ..            /* COMPUTE NUMBER OF EXTRA READ ATTEMPTS       */
               IF M=371 THEN M=0 ,.   ELSE M=M+1 ,.
               IF IDTRK(M)=-1 THEN GOTO GETITER ,.
               ITER=IDTRK(M)-L ,.   END ,.
               DO M=L TO L+ITER ,.   /* TRACK SEARCH LOOP                     */
            PUT STRING(REG8) EDIT(M) (F(8)) ,.
            READ FILE(LADA) INTO(Z) KEY(SKEY CAT REG8) ,.
            IF IOPT(4) THEN PUT EDIT(Z) (A) ,.
            /* AT LEAST A 0 LENGTH TITLE IS REQUIRED HERE                     */
            IF LENGTH(Z) GT 4 THEN DO ,.
               ITER=INDEX(SUBSTR(Z,5),FC) ,.   /* SRCH FOR END OF TITLE.  */
               /* EVEN UNTITLED SECTIONS HAVE AT LEAST A NULL TITLE.          */
               IF ITER=0 THEN DO ,.  /* IF REFERRED-TO GRP HAS NO ROLES       */
                  CALL ERRMSG('LERA',J) ,.      /* OR ACTORS THEN WARN.     */
                  ITER=LENGTH(Z)-3 ,.  END ,.   /* BUT GO AHEAD NORMALLY.   */
                                                /* ANYWAY, BUT AVOID STRG   */
               CTMP=SUBSTR(Z,5,ITER-1) ,.       /* MOVE TITLE TO CTMP.      */
               END ,.                           /* TITLE MAY BE NULL.       */
            IF IOPT(17) THEN
            PUT FILE(SYSPRINT) SKIP(1) EDIT
            ('CRTITL=''',CRTITL,'''','OLDTITL=''',CTMP,'''')
            (3 A,COL(40),3 A) ,.
            IF LENGTH(CRTITL)=0 THEN CRTITL=CTMP ,.
```

```
      IF CRTITL=CTMP THEN GOTO TITLESMATCH ,.
      END ,.                        /* BOTTOM OF LADDER SEARCH LOOP.    */
      CALL ERRMSG('TTLC',ITL) ,.
      PUT FILE(SYSPRINT) EDIT('  OLD=''',CTMP,'''') (A) ,.
      GOTO NOMATCH ,.
TITLESMATCH ..
      IF ITYP2='C' THEN DO ,.  /* IF CAST GRP LADDER ENTRY THEN DO */
      ITER=INDEX(Z,'?R' CAT SUBSTR(S,IP(GR(IGP)),IL(GR(IGP)))) ,.
      IF ITER=0 THEN DO ,.   CALL ERRMSG('NCGM',IP(GR(IGP))) ,.
         GOTO ITEMS ,.   END ,.          /* NO MATCH ON 1ST RCLE.    */
      DO K=ITER TO 1 BY -1 ,.            /* 1ST ROLE MATCHED.        */
      IF SUBSTR(Z,K,1)=FC THEN GOTO FCFOUND ,.   END ,.
      CALL ERRMSG('RNMP',ITER) ,.        /* PGM LOGIC ERROR.         */
FCFOUND ..          /* NORMAL EXIT FROM LEFT-HAND FC SEARCH LOOP. */
      ITER=INDEX(SUBSTR(Z,K+1),FC) ,.    /* SEARCH FOR END OF        */
      IF ITER=0 THEN ITER=LENGTH(Z)+1 ,. /* MATCHING ROLE'S GRP.     */
         ELSE ITER=ITER+K ,.
      S=S CAT SUBSTR(Z,K+1,ITER-K-1) ,.
      GCG(IGP)=LENGTH(S) ,.              /* SAVE FOR USE AS I ,.     */
      S=S CAT FC ,.
      GOTO ITEMS ,.   END ,.
      S=SUBSTR(S,1,I) CAT Z CAT FC ,.    /* APPEND SLE OR TLE.       */
      IS=LENGTH(S)-1 ,.                  /* SAVE FOR USE AS I.       */
      IF ITYP2='L' THEN ISEE=IS ,.
           /* ABOVE DEFINES BORDER BETWEEN SEEN GRPS & FIRM ONES.   */
           /* IF NO UNFIRM GRPS THEN ISEE IS STILL SET APPROPRIATE  */
           /* ISEE LESS THAN IS DEFINES SEEN GRPS EXISTENCE         */
      GOTO ITEMS ,.
NOMATCH ..
      PUT FILE(SYSPRINT) EDIT('  CRTITL=''',CRTITL,'''') (A) ,.
      GOTO ITEMS ,.   END ,.
      IF ITYP2='T' THEN DO ,.            /* $$$$$$$ TIME $$$$$$$     */
      IF KNTIME GT 0 THEN DO ,.   CALL ERRMSG('2TIM',ITL) ,.
                     GOTO ITEMS ,.   END ,.
      KNTIME=KNTIME+1 ,.          /* COUNT TIME ENTRIES WITHIN GRP.   */
      GTM(IGP)=ITP ,.             /* MAKE GCB POINT TO TIME'S ICB.    */
      GOTO ITEMDONE ,.   END ,.
      IF ITYP2='B' THEN GOTO ITEMS ,.    /* PAGES ARE IGNORED, BUT   */
                        /* THIS SAVES SPACE AND AVOIDS ITEM-CHAIN-    */
                        /* COUNT PROBLEMS (PAGE NUMBER ENTRY          */
                        /* REPLACING AN ITEM).                        */
ITEMDONE ..
      ITP=ITP+1 ,.   /* RESET TO PT TO NEXT FREE ICB, IF ANY.         */
      IF ITP GE ITMSIZ THEN DO ,.        /* IF NO ICB'S LEFT.        */
         CALL ERRMSG('NICB',ITR) ,.   IGR=I+1 ,.   GOTO GROUPS ,.
         END ,.
      GOTO ITEMS ,.                      /* GOTO TOP OF ITEMS LOOP.  */
OUTPUT .. IGP=0 ,. /* YOU COME HERE AFTER ALL (ALL) DATA HAS         */
                   /* BEEN READ IN AND GCB/ICB PTRS SET UP. 'FIG-*/
                   /* URING OUT', THAT IS ADDITION, DELETION ETC */
                   /* ARE NOW TO BE DONE.  ALSO CALLED PROCESSIN */
MORUPDAT ..        /* TOP OF INTERPRETATION LOOP.                    */
      IF INTRCHN=0 THEN DO ,.  /* IF CGLE JUST PROCESSED THEN          */
      INTRCHN=1 ,.             /* RESET THAT INDICATOR.                */
      IF IGP3SV=0 THEN         /* IF NO GRPS IN OUTPUT CHAIN YET       */
      IGP3,IGP4=GCG(IGP) ,.    /* THEN MAKE CGLE THE OUTPUT CHAIN.     */
      ELSE DO ,.               /* OTHERWISE MERELY                     */
      GF(GCG(IGP))=IGP3SV ,.   /* ADD CGLE TO CURRENT OUTPUT CHAIN,    */
      IGP3=GCG(IGP) ,.         /* RESTORE OUTPUT CHAIN HEAD PTR,       */
      IGP4=IGP4SV ,.   END ,.  /* AND OUTPUT CHAIN TAIL PTR.           */
      END ,.
      IGP=IGP+1 ,.             /* INCREMENT TO NEXT REGULAR GRP.       */
      IF IGP GT IGP2 THEN GOTO FORMOUT ,.          /* IF ALL          */
           /* REFERRING GRPS, IF ANY, HAVE BEEN PROCESSED THEN . . */
      IF GTM(IGP)+GRN(IGP)+GAN(IGP)=0 THEN DO ,.
         CALL ERRMSG('0GCB',IGP) ,.   GOTO MORUPDAT ,.   END ,.
```

```
            IF GRN(IGP)+GAN(IGP)=0 THEN CALL SUBGRADD ,. /* TIME RECHAIN */
            IF GCG(IGP) NE 0 THEN DO ,.          /* IF CAST GRP LADDER GRP */
               IGP3SV=IGP3 ,.   IGP4SV=IGP4 ,.
               IGP3,IGP4=GCG(IGP) ,. /* TEMPORARILY SET UP THE REFERRED- */
                 /* TO GRP AS THE OUTPUT CHAIN (ALSO THE SEARCH CHAIN). */
               INTRCHN=0 ,.          /* INDICATE CGLE 'JUST PROCESSED'. */
               SBITE='010'B ,.       /* SET FOR KEYING ON ROLES ONLY.   */
            DO ITL=1 TO GRN(IGP) ,.  /* FOR ALL ROLES IN REFERRING GRP. */
               CALL KEYS(IGP,ITL) ,. /* SRCH REFERRED-TO GRP FOR MATCH. */
            IF IRG =0 THEN DO ,.     /* IF NO MATCH THEN . . . . . . . */
               CALL ERRMSG('INKY',J),.  GOTO MORUPDAT,.  END,.  END,.
            DO ITL=GA(IGP) TO GAN(IGP)+GA(IGP)-1 ,.      /* TO AVOID    */
              /* REPLACEMENT PROCESSING, SET ALL ACTORS IN REFERRING  */
              /* GRP TO HAVE '+'S FOR PAIRED ACTOR ADDITION PROCESSIN */
            IPBIT(ITL)= NOT IMBIT(ITL) ,.   END ,.       /* UNLESS MINUS */
               END ,.
              /* IN 'SEE' LADDER CASE, BELOW, DELETE FIRST DUP ROLE,  */
              /* IF ANY, ENCOUNTERED FROM ITS UNFIRM GRP.  AND THEN   */
              /* PUT OUT BOTH FIRM AND UNFIRM GRP(S).                 */
            IF ISEE=IS THEN DO ,.                /* IF THIS SECTION     */
                           /* HAS A 'SEE' LADDER REF IN IT THEN DO ,.   */
               DO ITL=1 TO GRN(IGP) ,.
               SBITE='010'B ,.   CALL KEYS(IGP,ITL) ,.   /* KEY ON POS- */
               IF IRG NE 0 THEN DO ,.            /* SIBLY DUPLICATE ROLE. */
                  IF GRN(IRG)=1 THEN CALL GRPDLT ,.
                  ELSE DO ,.             /* DELETE ROLE FROM UNFIRM GRP.  */
                     IF IIL=0 THEN GR(IRG)=ILK(GR(IRG)) ,.
                     ELSE ILK(IIL)=ILK(IROF) ,.
                     GRN(IRG)=GRN(IRG)-1 ,.   END ,.   END ,.   END ,.
               END ,.
            IF GBIT(IGP) THEN DO ,.  /* IF GROUP DELETE THEN DO          */
               SBITE='111'B ,.   CALL KEYS(IGP,0) ,.
               IF IRG=0 THEN DO ,.   CALL ERRMSG('GDLT',IGP) ,.
                                     GOTO MORUPDAT ,.   END ,.
               CALL GRPDLT ,.
         GRPDLT .. PROC ,.                   /* THE GRP DELETE FUNCTION.  */
               IF INTRCHN=0 THEN GTM(IGP3),GRN(IGP3),GAN(IGP3)=0 ,.
                  ELSE
               IF IGP4=IGP3 THEN IGP3=0 ,.
                  ELSE DO ,.   IF IGL=0 THEN IGP3=GF(IRG) ,.
                  ELSE DO ,.   IF IRG =IGP4 THEN IGP4=IGL ,.
                  ELSE         GF(IGL)=GF(IRG) ,.   END ,.   END ,.
            END GRPDLT ,.
               GOTO MORUPDAT ,.   END ,.        /* END OF GRP DELET CODE  */
               /* $$$$$$$$$ UNPAIRED ROLE SUBGRP RECHAIN FOLLOWS BELOW  */
            IF GAN(IGP)=0 THEN IF NOT IMBIT(GR(IGP)) THEN CALL SUBGRADD ,.
               ELSE DO ,.  /* TRY GRP DELETE FOR UNPAIRED ROLE GRP.      */
               SBITE='010'B ,.
               CALL KEYS(IGP,1) ,.   IF IRG NE 0 THEN
               DO ,.   GR(IGP)=ILK(GR(IGP)) ,.   GRN(IGP)=GRN(IGP)-1 ,.
               GR(IRG)=ILK(GR(IRG)) ,.   GRN(IRG)=GRN(IRG)-1 ,.   END ,.
               ELSE          CALL ERRMSG('URDT',IGP) ,.
               GOTO MORUPDAT ,.   END ,.
               /*$$$$$$$$$$ UNPAIRED ACTOR SUBGRP RECHAIN FOLLOWS BELOW  */
            IF GRN(IGP)=0 THEN IF NOT IMBIT(GA(IGP)) THEN CALL SUBGRADD ,.
               ELSE DO ,.  /* TRY GRP DELETE FOR UNPAIRED ACTOR GRP.     */
               SBITE='001'B ,.
               CALL KEYS(IGP,1) ,.   IF IRG NE 0 THEN
               DO ,.   GA(IGP)=ILK(GA(IGP)) ,.   GAN(IGP)=GAN(IGP)-1 ,.
               GA(IRG)=ILK(GA(IRG)) ,.   GAN(IRG)=GAN(IRG)-1 ,.   END ,.
               ELSE        /* NO MATCH IN ATTEMPTED UNPAIRED ACTOR GRP   */
                                                    /* DELETION.        */
               CALL ERRMSG('UADT',IGP) ,.   GOTO MORUPDAT ,.   END ,.
            IF IPBIT(GR(IGP)) OR IMBIT(GR(IGP))          /* IF ROLE IS   */
               THEN GOTO ACTORKEY ,.                     /* SIGNED THEN KEY ON THE */
                                                         /* ACTOR PART OF GROUP.   */
```

```
            SBITE='010'B ,.                    /* ROLE IS NOT SIGNED, SO */
            EMSG='RKEY' ,.
            CALL PAIRS(GA,GAN) ,.
         ACTORKEY ..                           /* ROLE IS SIGNED, SO     */
            SBITE='001'B ,.                    /* SET UP FOR ACTOR SRCH, */
            EMSG='AKEY' ,.
            CALL PAIRS(GR,GRN) ,.
         PAIRS .. PROC (GA,GAN) ,.    /* THIS PROC, PROPERLY MASSAGED,     */
            /* DOES MOST OF THE WORK FOR ORDINARY PAIRED GRPS, ITEM */
            /* ADDITION, REPLACEMENT, & DELETION, & GRP RELINK IF   */
            /* NO MATCH.  RELINKING IS TO OUTPUT CHAIN, AND IS THE  */
            /* MOST FREQUENTLY PERFORMED OPERATION.                 */
            DCL (GA(*),GAN(*)) FIXED BIN(15,0) ,.
            CALL KEYS(IGP,1) ,.                /* KEY ON IT.        */
            IF IRG=0 THEN IF IPBIT(GA(IGP)) OR IMBIT(GA(IGP)) THEN DO ,.
            /* NO MATCH FOR A ROLE IN A PAIRED GRP WITH A SIGNED ACTOR */
               CALL ERRMSG(EMSG,IGP) ,.   GOTO MORUPDAT ,.   END ,.
               ELSE DO ,.  /* ADD GRP IF BOTH UNSIGNED & KEY SEARCH FAIL */
               CALL SUBGRADD ,.   GOTO MORUPDAT ,.   END ,.
               ELSE DO ,.  /* IRG NE 0 SO KEY SEARCH SUCCEEDED, SO DO  */
                           /* THE '+', '-', AND REPLACE FCNS.          */
         RPLUSMINUS ..
               IF IPBIT(GA(IGP)) THEN DO ,.    /* SINGLE ACTOR ADDITION */
               L=GA(IGP) ,.   GA(IGP)=ILK(L) ,.   GAN(IGP)=GAN(IGP)-1 ,.
               ILK(L)=GA(IRG) ,.   GA(IRG)=L ,.   GAN(IRG)=GAN(IRG)+1 ,.
               GOTO RPLUSMINUS ,.   END ,.
               IF IMBIT(GA(IGP)) THEN DO ,.    /* IF ACTOR DELETE THEN  */
                                               /* SRCH IRG GRP FOR      */
                                               /* ITEM EQUALLING GA(IGP) */
                  L=GA(IGP) ,.                 /* SAVES SUBSCRIPTING.   */
                  K=GA(IRG) ,.        /* SAVES SUBSCRIPTING.            */
                  IIN=0 ,.
            MORESRCH ..       /* THIS LOOP SEARCHES FOR THE ITEM TO BE    */
                              /* DELETED IN THE REFERRED-TO GRP.          */
                  IF IIN GE GAN(IRG) THEN DO ,.
                                    CALL ERRMSG('ADIT',IP(L)) ,.
                                    GOTO MORUPDAT ,.   END ,.
                  IF SUBSTR(S,IP(K),IL(K))=SUBSTR(S,IP(L),IL(L))
                     THEN DO ,.
                     IF K=GA(IRG) THEN GA(IRG)=ILK(K) ,.
                     ELSE ILK(KOLD)=ILK(K) ,.
                     GAN(IRG)=GAN(IRG)-1 ,.   GAN(IGP)=GAN(IGP)-1 ,.
                     GA(IGP)=ILK(L) ,.   GOTO RPLUSMINUS ,.   END ,.
                  KOLD=K ,.   K=ILK(K) ,.
                  GOTO MORESRCH ,.   END ,.
                 /* THE FAMOUS REPLACE FUNCTION FOLLOWS.                */
                  GAN(IRG)=GAN(IGP) ,.   GA(IRG)=GA(IGP) ,.
                  GOTO MORUPDAT ,.   END ,.   END ,.
         SUBGRADD .. PROC ,.              /* SUBGRP RECHAIN FOR UNPAIRED GRP */
            IF IGP4 NE 0 THEN GF(IGP4)=IGP ,.
            IGP4=IGP ,.   IF IGP3=0 THEN IGP3=IGP4 ,.
            GOTO MORUPDAT ,.   END ,.
         FORMOUT ..
            Z=FC CAT SMOD CAT FA CAT SMOD CAT CRTITL ,.  /* THIS STATE-
            /* MENT BEGINS THE FORMATTING OF FINAL OUTPUT IN THE Z BUFR. */
            IF IGP3=0 THEN GOTO ENDOUT ,.      /* IF EMPTY OUTPUT CHAIN. */
            IXG=IGP3 ,.    /* INITIALIZE PTR TO 1ST GRP IN OUTPUT CHAIN. */
            LBIT='1'B ,.   /* INDICATE LAST GRP NOT FOUND YET.           */
            DO WHILE(LBIT) ,.          /* LBIT='0'B=LAST GRP NOT ENCOUNTRD */
            IF IXG=IGP4 THEN LBIT='0'B ,.      /* IF LAST GRP ENCOUNTERD */
            IF GTM(IXG)+GRN(IXG)+GAN(IXG) GT 0 THEN      /* THIS STMT    */
               Z=Z CAT '?G' ,.         /* ELIMINATES NULL OUTPUT GRPS.    */
               ELSE GOTO ENDIN ,.
            IF GTM(IXG) NE 0 THEN     /* IF THIS GRP HAS A TIME ENTRY.    */
            Z=Z CAT '?T'         CAT SUBSTR(S,IP(GTM(IXG)) ,IL(GTM(IXG)))  ,.
            IXI=GR(IXG) ,.
```

```
        DO IXX=IXI TO IXI+GRN(IXG)-1 ,.
        Z=Z CAT '?R'       CAT SUBSTR(S,IP(IXI),II(IXI)) ,.
        IXI=ILK(IXI) ,.   END ,.
        IXI=GA(IXG) ,.
        DO IXX=IXI TO IXI+GAN(IXG)-1 ,.
        Z=Z CAT '?A'       CAT SUBSTR(S,IP(IXI),II(IXI)) ,.
        IF IP(IXI) GT ISEE THEN Z=Z CAT '?' ,.
        IXI=ILK(IXI) ,.   END ,.
    ENDIN ..
        IXG=GF(IXG) ,.  IF IXG=0 THEN CALL ERRMSG('IXG0',IXI) ,.
        END ,.
    ENDOUT ..           /* AT THIS POINT THE OUTPUT BLK, IN Z, IS     */
                        /* READY, AND ONLY HOUSKEEEING REMAINS BEFORE */
                        /* THE ACTUAL WRITE STATEMENT.                */
        IF IOPT(24) OR ERR THEN
        PUT FILE(SYSPRINT) EDIT
        (J,' OUTPUT=''',Z,'''') (X(3),4 A) ,.
        RL=LENGTH(Z)+32 ,.
    RESIZE .. SP=SP-RL ,.
        IF SP LT 0 OR (TITBIT AND STBT(NS)) THEN DO ,.
        IF RL GT 3588 THEN CALL ERRMSG('OTSZ',LENGTH(Z)) ,.
        SP=3625 ,.   KLAD=KLAD+1 ,.   STBT='0'B ,.
        GOTO RESIZE ,.   END ,.
        STBT(NS)='1'B ,.
        PUT STRING(REG8) EDIT(KLAD) (F(8)) ,.
        WRITE FILE(LADA) FROM(Z) KEYFROM(STRUKEY CAT REG8) ,.
        SP=SP-(537*(RL-20)/512+61) ,.
        IF IOPT(1) OR ERR THEN PUT FILE(SYSPRINT) EDIT
              ((L,SUBSTR(S,L*100+1,MIN(100,I-L*100))
              DO L=0 TO (I-1)/100 )) (SKIP,F(2),COL(10),A(100)) ,.
        IF IOPT(5) OR ERR THEN
        PUT FILE(SYSPRINT) SKIP(1) EDIT
        ( 'B=',J,' LNGTH=',IF,' KEY=''',R90FKEY,''' ''',
        SUBSTR(S,1,MIN(83,I)),''''    ,NEWDATE/372+1659,
        TRUNC(MOD(NEWDATE,372)/31)+1,MOD(NEWDATE,31)+1)
        (A,F(5),A,F(4),5 A,COL(123),F(4),2 F(3)) ,.
        MAXOUT=MAX(MAXOUT,LENGTH(Z)) ,.    /* FOR MAX BLKSIZE STATS. */
        NOUT=NOUT+LENGTH(Z) ,.   /* TOTAL OUTPUT CHAR STATS.          */
        LASTRK=MAX(LASTRK,KLAD) ,.
        IF IOPT(7) THEN DO ,.
        S='  ' CAT STRUKEY CAT Z ,.          /* KEY, ALSO AT LEAST 18  */
                                             /* BYTES FOR TAPE BLOCK.  */
        WRITE FILE(LADS) FROM(S) ,.    END ,.
        IF OLDKEY NE KEYBITS THEN DO ,.    /* EACH TIME DATE CHANGES */
        IOLDATE=0 ,.                /* CLEAR LEFT HALF OF IOLDATE.    */
        OLDKEY=KEYBITS ,.           /* PUT CODED DATE IN RIGHT HALF.  */
        N=MOD(IOLDATE,372) ,.       /* GET TABLE ADDRESS FOR DAY OF YR. */
        IDTRK(N)=KLAD ,.            /* SEARCHES FOR SECTIONS THAT WERE  */
        END ,.          /* ON DAY OF YEAR N WILL START AT TRACK KLAD. */
        GOTO INPUT ,.               /* PROCESS NEXT SECTION, IF ANY.  */
    KEYS .. PROC(IGRP,IOFF) ,.   /* GENERAL PURPOSE ITEM KEYING.      */
        /* ON EXIT IRG PTS TO THE GCB IN WHICH THE FOUND ITEM RESIDES.
        IF IRG=0 THEN NO MATCH WAS FOUND.  ALSO IGL PTS TO THE PREVIOUS
        GCB ON THE CHAIN, IF ANY, OR IS 0.  SIMILARLY IROF PTS TO THE
        ICB OF THE ITEM ON WHICH A MATCH WAS MADE.  IT IS NOT
        NECESSARILY 0 IN CASE OF A NO MATCH.  AND IIL PTS TO THE
        PREVIOUS ICB ON THE CHAIN, IF ANY, OR ELSE TO 0.  RBITS
        INDICATES THE TYPE OF ITEM ON WHICH KEYING WAS DONE.          */
        IRG,IROF=0 ,.   RBITS='0'B ,.
        IF IGP3=0 THEN RETURN ,.
        IF SBITS(2) THEN DO ,.
           CALL RASH(GR,GRN) ,.
           IF IRG NE 0 THEN DO ,.
              RBITS(2)='1'B ,.   RETURN ,.   END ,.   END ,.
        IF SBITS(3) THEN DO ,.
           CALL RASH(GA,GAN) ,.
```

```
            IF IRG NE 0 THEN DO ,.
               RBITS(3)='1'B ,.   RETURN ,.   END ,.
            END ,.
         IF NOT SBITS(3) OR GTM(IGRP)=0 THEN RETURN ,.
         IGL=0 ,.   IRG=IGP3 ,.
         COMPTMP=SUBSTR(S,IP(GTM(IGRP)),IL(GTM(IGRP))) ,.
      MORETIME ..
         IF SUBSTR(S,IP(GTM(IRG)),IL(GTM(IRG)))=COMPTMP THEN RETURN ,.
         IF IRG=IGP4 THEN DO ,.
            IGL=IRG ,.   IRG=GF(IRG) ,.   GOTO MORETIME ,.   END ,.
         IRG=0 ,.       /* INDICATE NO MATCH FOUND AT ALL FOR KEY     */
      RASH .. PROC(IT,ITN) ,.       /* RCIE OR ACTCB SEARCH PROC       */
         DCL (IT(*), ITN(*)) FIXED BIN(15,0) ,.
         IRG,IROF=0 ,.   RBITS='0'B ,.
         IF ITN(IGRP)=0 THEN RETURN ,.      /* IF THE SUBGROUP HAS NO */
                                            /* ITEMS TO MATCH ON THEN */
         IROF=IT(IGRP) ,.           /* IROF NOW PTS TO 1ST ICB IN THE  */
                                    /* SUBGRP CHAIN, THE ITEM'S        */
                                    /* EXISTENCE IS GUARANTEED BY THE  */
                                    /* STATEMENT ABOVE.                */
         DO IIN=1 TO IOFF-1 ,.   IROF=ILK(IROF) ,.   END ,.
           /* THE ABOVE DOLOOP MOVES ALONG SUBGRP CHAIN TO THE     */
           /* PROPER LINK OF THE SUBGRP CHAIN AS DEFINED BY THE    */
           /* IOFF PARM IN THE CALL TO KEYS.                       */
         COMPTMP=SUBSTR(S,IP(IROF),IL(IROF)) ,.
           /* SET UP COMPARISON TEMP OUTSIDE OF LOOP.              */
         IGL=0 ,.   IRG=IGP3 ,.
         GCHK='1'B ,.
         DO IOUT=1 TO IGPSIZ WHILE(GCHK) ,. /* THE OUTER SEARCH LOOP  */
         IF IRG=IGP4 THEN GCHK='0'B ,.
            IF ITN(IRG) NE 0 THEN DO ,.
              IIL=0 ,.   IROF=IT(IRG) ,.
              DO IIN=1 TO ITN(IRG) ,.       /* THE INNER SEARCH LOOP. */
                 IF COMPTMP=SUBSTR(S,IP(IROF),IL(IROF)) THEN RETURN ,.
                 IIL=IROF ,.   IROF=ILK(IROF) ,.
                 END ,.
              END ,.
              IGL=IRG ,.   IRG=GF(IRG) ,.
            END ,.
         IRG,IROF=0 ,.   RETURN ,.
         END RASH ,.
         END KEYS ,.
      ERRMSG .. PROC(NOTE,LOC) ,.  /* GENERAL PURPOSE ERROR MESSAGES.  */
         DCL NOTE CHAR(4) ,.
         IERR=MIN(MAX(LOC-10,1),LENGTH(S)) ,.
         PUT FILE(SYSPRINT) SKIP(1) EDIT
         ('* ',NOTE,LOC,' **********  ''',
         SUBSTR(S,IERR,MIN(I-IERR+1,21)),'''')
         (2 A,F(5),3 A) ,.
         IF IOPT(27) THEN ERR='1'B ,.       /* SET ERR BIT FOR MORE.  */
         IERRNO=IERRNO+1 ,.         /* COUNT ERRORS.                  */
         END ,.
      PROGEND ..          /* FOR NORMAL END OF PGM OR ERROR ON COND.   */
         PUT FILE(SYSPRINT) SKIP(1) EDIT
         ('ERRORS=',IERRNO,' MAX OUTPUT BLKSIZE=',MAXCUT,
         ' MAX ITEMS=',ITPMAX,' MAX GRPS=',IGPMAX,
         ' MAX REL. TRK=',LASTRK,' OUTPUT CHARS=',NCUT)
         (8 A,SKIP) ,.   END ,.
/*
//ROOT     JOB 99999211,WCD,MSGLEVEL=(1,1)
//R EXEC LSPBACMP,MEM=ROOT,PARM.PL1L='C48,NT,SIZE=90K'
//BAKUP.SYSUT1 DD *
      ROOT .. PROC OPTIONS(MAIN) ,.
         DCL IFBIT EXT, /* INDICATES NO ENDFILE ON LADDER INPUT YET.  */
         IAR EXT,       /* PTR TO 1ST FREE UNUSED SLOT IN X.S.        */
         ICHP EXT,      /* PTR TO 'LAST CHAR RETURNED' BY GCHAR.      */
```

```
            ISTOP EXT INIT(0),       /* PROGRAM STOP SWITCH.              */
            IOPT (80) BIT(1) EXT ,.
            ON ERROR SNAP PUT FILE(SYSPRINT) SKIP(1) EDIT
            ('ROOT ERR ONCODE=',ONCODE) (A) ,.
            PUT FILE(SYSPRINT) SKIP(1) EDIT('ROOT STARTED.') (A) ,.
            IFBIT=0 ,.     /* INDICATES PRIME INPUT ECF NOT FOUND YET.   */
            IAR=1 ,.       /* PTS TO LAST SE ENTRY+1 ,.                  */
            ICHP=80 ,.     /* SET TO CAUSE GCHAR TO PRIME CARD BUFR.     */
            ON ENDFILE(SYSIN) GOTO SYSINEND ,.
            GET FILE(SYSIN) EDIT(IOPT) (B(1)) ,.
        NXTRNG .. CALL CTRLCRD ,.
            IF IOPT(10) THEN
            PUT FILE(SYSPRINT) SKIP(1) EDIT('GOT BACK FROM CTRLCRD') (A) ,.
            IF IFBIT NE 0 THEN DO ,.
            PUT FILE(SYSPRINT) SKIP(1) EDIT('IFBIT=1, END OF PGM') (A) ,.
            RETURN ,.   END ,.
            IF ISTOP NE 0 THEN GOTO SYSINEND ,.          /* FOR TYPE 9   */
                                  /* STATEMENT INTERPRETED BY CTRLCRD */
            CALL ITEMGET ,.
            IF IOPT(10) THEN
            PUT FILE(SYSPRINT) SKIP(1) EDIT('GOT BACK FROM ITEMGET') (A) ,.
            GOTO NXTRNG ,.
        SYSINEND ..
            PUT FILE(SYSPRINT) SKIP(1) EDIT('ENDFILE ON SYSIN') (A) ,.
            END ,.
/*
//PL1L.SYSLIN DD DISP=(NEW,KEEP),SPACE=(TRK,(50,10,10))
/*
//CTRLCRD JOB 99999211,WCD,MSGLEVEL=(1,1)
//C EXEC LSPBACMP,MEM=CTRLCRD,PARM.PL1L='C48,NT,A,X,SIZE=90K,FW,M'
//BAKUP.SYSUT1 DD *
            /* CTRL PROGRAM */ (STRG,SUBRG) ..
            /* THIS OVERLAY INTERPRETS SELECTION CONTROL CARDS FOR THE   */
            /* ITEMGET OVERLAY.                                          */
        CTRLCRD .. PROC ,.
            %DCL SUPERSSIZE FIXED ,.
            %DCL SIZECHAR0000 CHAR ,.
            %DCL SUPERSPLUS FIXED ,.
            %DCL MANYSSSIZE FIXED ,.
            %SUPERSSIZE=140 ,.
            %MANYSSSIZE=SUPERSSIZE*2 ,.
            %SUPERSPLUS=SUPERSSIZE+1 ,.
            %SIZECHAR0000='''' CAT SUPERSSIZE CAT '''' ,.
            ON ERROR SNAP PUT FILE(SYSPRINT) SKIP(1) EDIT
            ('CTRL ERR ONCODE=',ONCODE) (A) ,.
            DCL
            ISTOP EXT INIT(0),       /* SWITCH TO STOP PGM ON STMT 9.    */
            KITM FIXED BIN(31,0) STATIC EXT    /* OUTPUT REC COUNT.      */
               INIT(0),
            J FIXED BIN(31,0) INIT(0) STATIC EXT,
            IREC FIXED BIN(31,0) STATIC INIT(0) EXT,     /* FOR TOTAL   */
                                  /* POSSIBLE RECORDS GENERATED, VALID*/
                                  /* IF ALL LEVEL 1 INCLUSIVENESS ONLY*/
            MANY(2,SUPERSSIZE) STATIC INIT((MANYSSSIZE) 0)      EXT
               FIXED BIN(31,0),
                 /* ABOVE IS HIT COUNT + REC COUNT FOR STATISTICS.       */
            IDATE FIXED BIN(31,0),   /* FOR CODED DATE CONVERSION OVERLAY*/
            ILDAT(6) EXT,            /* DATE RANGE TEMP.                 */
          1 X EXT,                   /* SCE TABLE.                       */
            2 IDR(2) FIXED BIN(31,0),        /* DATE RANGE FOR ITEMGET */
            2 D,                     /* CURRENT DEFAULTS FOR SE TABLE.   */
             3 INC CHAR(1),
             3 SEC CHAR(1) VAR,
             3 TIME CHAR(8) VAR,
             3 ROLE CHAR(40) VAR,
             3 ACTOR CHAR(40) VAR,
```

```
        3 TITLE CHAR(83) VAR,
        3 THEATRE CHAR(8) VAR,
      2 S(SUPERSSIZE) LIKE D,  /* SE'S THEMSELVES.                    */
      IBUFBIT EXT INIT(0),
      IOPT(80) BIT(1) EXT,     /* OPTION BITS.                        */
      IAR EXT,       /* PTR TO 1ST FREE UNUSED SLOT IN X.S.          */
      CARD CHAR(80) EXT,       /* GCHAR'S SYSIN BUFFER.               */
      CAR(80) CHAR(1) DEF CARD ,          /* ACCESS TO CARD.          */
      ICHP EXT,      /* PTR TO 'LAST CHAR RETURNED' BY GCHAR.        */
      CAPS CHAR(26) INITIAL('ABCDEFGHIJKLMNOPQRSTUVWXYZ') STATIC,
      SMALLS CHAR(26) STATIC INITIAL('abcdefghijklmnopqrstuvwxyz'),
      SCHAR CHAR(1),
      CCHAR CHAR(1),
      CURT CHAR(83) VAR,       /* TEMP FOR BUILDING NEW ITEMS.        */
      ITMSTMT LABEL ,.
      LF=0 ,.                  /* MEANS PRINT SCE ONLY IN SUMMARY.    */
      IF IBUFBIT NE 0 THEN CALL SUMMARY ,.
      LF=1 ,.                  /* INDICATES PRINT SCE + STATS.        */
NXTBLNK .. CALL GCHAR ,.       /* LOOP TO FIND INCLUSIVENESS CHAR     */
      IF CCHAR=' ' THEN GOTO NXTBLNK ,.  /* SKIPS PAST BLANKS         */
      IF CCHAR='9' THEN DO ,.
         ISTOP=1 ,.                        RETURN ,.   END ,.
      IF CCHAR='7' THEN        /* 'GO' STATEMENT TO RUN ITEMGET.      */
SETUPFIN .. DO ,.
      CALL SUMMARY ,.
      RETURN ,.   END ,.
SUMMARY .. PROC ,.
      IF LF=0 OR (LF=1 AND IOPT(11)) THEN DO ,.    /* LIST SCB OPT   */
      PUT FILE(SYSPRINT) SKIP(2) EDIT(ILDAT) (F(5)) ,.
      PUT FILE(SYSPRINT) SKIP(1) EDIT
      ('NO. INC SEC  TIME  THEATRE TITLE','ROLE','ACTOR')
      (A,COL(74),A,COL(106))
      ((K,S.INC(K),S.SEC(K),S.TIME(K),S.THEATRE(K),S.TITLE(K),
      S.ROLE(K),S.ACTOR(K) DO K=1 TO IAR-1))
      (SKIP(1),F(3),X(1),A(4),A(1),2 A(9),A,COL(74),A,
      COL(104),A(29)) ,.
      END ,.
      IF LF=0 THEN DO ,.
      PUT FILE(SYSPRINT) SKIP(2) EDIT
      ('POSSIBLE RECS=',IREC,'INPUT BLKS=',J,
      'OUTPUT RECS=',KITM,'NO.  HITS RECS PRODUCED BY HITS')
      (3 (SKIP(1),A,COL(20),F(8)),SKIP(2))
      ((J,MANY(1,J),MANY(2,J) DO J=1 TO IAR-1))
      (SKIP(1),F(3),F(6),F(6)) ,.
      IF IOPT(12) THEN MANY=0 ,.   END ,.
      END SUMMARY ,.
      IF CCHAR='0' THEN DO ,.  /* CLEAR ALL DEFAULTS.                 */
         D='' ,.   GOTO NXTBLNK ,.   END ,.
      IF CCHAR='8' THEN DO ,.  /* CLEAR DEFAULTS AND SE TABLE.        */
      D='' ,.
      DO K=1 TO IAR ,.   S(K)=D ,.   END ,.
      IAR=1 ,.   GOTO NXTBLNK ,.   END ,.
      IF CCHAR='5' THEN DO ,.  /* IF DATE RANGE STATEMENT             */
         GET STRING(CARD) EDIT(ILDAT)
         (X(1),2 (X(1),F(4),2 (X(1),F(2)))) ,.
      DO IS=1 TO 2 ,.          /* FOR UPPER AND LOWER DATE RANGES.    */
      IDATE=(ILDAT((IS-1)*3+1)-1659)*372 +
            (ILDAT((IS-1)*3+2)-1)*31 +
             ILDAT((IS-1)*3+3)-1 ,.
      IDR(IS)=IDATE ,.   END ,.
         ICHP=24 ,.   END ,.   /* CHAR AFTER FIXED FORMAT DATE        */
                               /* RANGE STATEMENT.                    */
         ELSE DO ,.
      IF IAR GE SUPERSPLUS THEN DO ,.    /* IF SCE OVERFLOW.          */
      PUT FILE(SYSPRINT) SKIP(1) EDIT
      ('IAR OVERFLOW',SIZECHAR0000) (A) ,.
```

```
      GOTO SETUPFIN ,.    END ,.
      IF CCHAR='6' THEN DO ,.
         D='' ,.                    /* CLEAR ALL DEFAULTS AND SET UP    */
         GET STRING(CARD) EDIT(ISE,LSE) (X(2),2 F(4)) ,.
         ICHP=13 ,.                 /* FOR REPETITIVE SPECIFICATIONS.   */
         CALL ITMLOOP ,.    END ,.
         ELSE DO ,.
      IF CCHAR GE '1' THEN IF CCHAR LE '4' THEN DO ,.
         S.INC(IAR)=CCHAR ,.     ISE,LSE=IAR ,.
         CALL ITMLOOP ,.    END ,.    END ,.    END ,.
      GOTO NXTBLNK ,.
ITMLOOP .. PROC ,. /* THIS PROCEDURE INTERPRETS SELECTION ENTRY  */
                   /* STATEMENTS EXCEPT FOR THE INCLUSIVENESS    */
ITM .. CURT='' ,.             /* INITIALIZE NEW ITEM TO = ''      */
      CALL GCHAR ,.           /* GET THE ITEM TYPE CHAR           */
      SCHAR=CCHAR ,.          /* SAVE ITEM TYPE CHARACTER         */
NXTCHAR ..
      CALL GCHAR ,.           /* GET A CHAR.                      */
      IF CCHAR='/' THEN DO ,.
         IBIT=0 ,.            /* INDICATE NONBLANK ENCOUNTERED.   */
         CALL GCHAR ,.
         CURT=CURT CAT CCHAR ,.    END ,.
         ELSE DO ,.
      ITMSTMT=ITM ,.          /* INDICATE NOT END OF STATEMENT YET*/
      IF CCHAR='&' THEN GOTO CLOSITM ,.  /* END OF ITEM, SET IT UP */
      IF CCHAR=' ' THEN DO ,.  /* IF BLANK THEN.                  */
         IF IBIT=1 THEN DO ,.  /* IF 2ND CONTIGUOUS BLANK THEN    */
            CURT=SUBSTR(CURT,1,LENGTH(CURT)-1) ,.
            ITMSTMT=STMT ,.    /* FOR END OF STMT, NOT JUST ITEM. */
            GOTO CLOSITM ,.    END ,.
            IBIT=1 ,.          /* INDICATE ONE BLANK ENCOUNTERED. */
            CURT=CURT CAT CCHAR ,.    END ,.
      ELSE DO ,.
         CURT=CURT CAT TRANSLATE(CCHAR,SMALLS,CAPS) ,.
         IBIT=0 ,.    END ,.    END ,.
      GOTO NXTCHAR ,.
CLOSITM ..
      IBIT=0 ,.                 /* RESET FOR START OF NEXT ITEM.    */
      IF SCHAR='A' THEN D.ACTOR=CURT ,.    ELSE DO ,.
      IF SCHAR='R' THEN D.ROLE=CURT ,.    ELSE DO ,.
      IF SCHAR='T' THEN DO ,.
      IF IOPT(4) THEN D.TITLE=TRANSLATE(CURT,CAPS,SMALLS) ,.
         ELSE D.TITLE=CURT ,.    END ,.    ELSE DO ,.
      IF SCHAR='C' THEN D.TIME=CURT ,.    ELSE DO ,.
      IF SCHAR='M' THEN D.THEATRE=CURT ,.    ELSE DO ,.
      IF SCHAR='S' THEN D.SEC=CURT ,.
      END ,.    END ,.    END ,.    END ,.    END ,.
      GOTO ITMSTMT ,.
STMT ..
      DO IS=ISE TO LSE ,.
      S.INC(IAR)=S.INC(IS) ,.
      IF LENGTH(D.ACTOR)=0
         THEN S.ACTOR(IAR)=S.ACTOR(IS) ,.
         ELSE S.ACTOR(IAR)=D.ACTOR ,.
      IF LENGTH(D.ROLE)=0
         THEN S.ROLE(IAR)=S.ROLE(IS) ,.
         ELSE S.ROLE(IAR)=D.ROLE ,.
      IF LENGTH(D.SEC)=0
         THEN S.SEC(IAR)=S.SEC(IS) ,.
         ELSE S.SEC(IAR)=D.SEC ,.
      IF LENGTH(D.TIME)=0
         THEN S.TIME(IAR)=S.TIME(IS) ,.
         ELSE S.TIME(IAR)=D.TIME ,.
      IF LENGTH(D.TITLE)=0
         THEN S.TITLE(IAR)=S.TITLE(IS) ,.
         ELSE S.TITLE(IAR)=D.TITLE ,.
```

```
      IF LENGTH(D.THEATRE)=0
         THEN S.THEATRE(IAR)=S.THEATRE(IS) ,.
         ELSE S.THEATRE(IAR)=D.THEATRE ,.
      IAR=IAR+1 ,.
      IF IAR GE SUPERSSIZE THEN DO ,.    /* IF SCE OVERFLOW IMINENT*/
        PUT FILE(SYSPRINT) SKIP(1) EDIT
      ('SCE OVERFLOW',SIZECHAR0000) (A) ,.   GOTO SETUPFIN ,.   END,.
      END ,.
      END ITMLOOP ,.
   GCHAR .. PROC ,.    /* THIS IS AN INPUT BUFFERING PROCEDURE.  IT  */
                       /* AVOIDS REPEATED CALLS TO DATA MANAGEMENT   */
                       /* AND THUS SAVES BOTH SPACE AND TIME         */
      IF ICHP GE 80 THEN DO ,.
         GET FILE(SYSIN) EDIT(CARD) (A(80)) ,.
         ICHP=0 ,.   END ,.
      ICHP=ICHP+1 ,.
      CCHAR=CAR(ICHP) ,.   END GCHAR ,.
   CTRLEND .. END CTRLCRD ,.
/*
//ITEMGET  JOB 99999211,WCD,MSGLEVEL=(1,1)
//I EXEC LSPBACMP,MEM=ITEMGET,PARM.PL1L='C48,NT,M,SIZE=90K'
//BAKUP.SYSUT1 DD *
      /* ITEMSGET PROGRAM */ (SUBRG,STRG) ..
      /* MUCH OF THE CODING IN THE ITEMSGET PROCEDURE OF THIS PGM  */
      /* WAS COPIED FROM LADDR AND THEN MODIFIED, THUS RESULTING   */
      /* IN SOME ODDITIES AND INEFFICIENCIES IN ORDER TO SAVE      */
      /* PROGRAMMING TIME.                                         */
   ITEMGET .. PROC ,.
      %DCL SUPERSSIZE FIXED ,.
      %DCL MANYSSSIZE FIXED ,.
      %SUPERSSIZE=140 ,.
      %MANYSSSIZE=SUPERSSIZE*2 ,.
      DCL
      CAPS CHAR(26) INITIAL('ABCDEFGHIJKLMNOPQRSTUVWXYZ') STATIC,
      DATE8 CHAR(8) DEF Z POS(2),        /* ON DATE IN OUTPUT BUFR */
      IOPT(80) BIT(1) EXT,     /* OPTION BITS ARRAY, READ BY ROOT. */
      IAR EXT,         /* PTR TO 1ST FREE UNUSED SLOT IN X.S.      */
     1 X EXT,          /* MAIN INTERFACE BETWEEN ITEMGET & CTRLCRD. */
      2 IDR(2) FIXED BIN(31,0),          /* CURRENT DATE RANGE.    */
      2 D,                       /* CURRENT SELECTION DEFAULTS.    */
       3 INC CHAR(1),            /* INCLUSIVENESS TAG.             */
       3 SEC CHAR(1) VAR,        /* SECTION TYPE.                  */
       3 TIME CHAR(8) VAR,
       3 ROLE CHAR(40) VAR,
       3 ACTOR CHAR(40) VAR,
       3 TITLE CHAR(83) VAR,
       3 THEATRE CHAR(8) VAR,
      2 S(SUPERSSIZE) LIKE D,  /* SELECTION ENTRY ARRAY.           */
      IBUFBIT EXT INIT(0),     /* FLAG INDICATING 1ST OVERLAY OR NO*/
      IFBIT EXT,               /* END OF FILE FLAG, 1 = END OF     */
                               /* FILE HAS BEEN ENCOUNTERED.       */
      LEV CHAR(1) INIT('1'),   /* CURRENT LEVEL OF INCLUSIVENESS.  */
      TRKNO CHAR(8),           /* TRACK NO. TEMP.                  */
      KEYAR(20) CHAR(12),      /* ARRAY OF KEYS OF CURRENT PERFORM */
      ITRK(20),                /* REL TRKS FOR ABOVE KEYS          */
      KKEY CHAR(2) DEF K ,.    /* RECORDED KEY FOR ITEM REWRITES.  */
      DCL
      SECTYPE CHAR(1) DEF Z,
      IYR CHAR(4) DEF Z POS(2),
      IMON CHAR(2) DEF Z POS(6),
      IDY CHAR(2) DEF Z POS(8),
      THETR CHAR(8) DEF Z POS(10),
      TIME CHAR(8) DEF Z POS(181),
     (GB  FIXED BIN(15,0),     /* THESE VARIABLES ARE DECLARED     */
      GS  FIXED BIN(15,0),     /* STATIC TO SAVE CORE SINCE THEY   */
      GUM FIXED BIN(15,0),     /* ARE NOW OVERLAIN IN STATIC CSECT */
```

```
GSN FIXED BIN(15,0),
GBN FIXED BIN(15,0),
   /* THE NEXT 2 ARRAYS FORM THE ITEM CONTROL BLOCKS (ICE'S) */
   IP   (70),
   IL   (70)) STATIC ,.
DCL (          /* DCL STMT FOR CONSTANTS ETC. (ALL STATIC).  */
THLEGAL CHAR(80) VAR INITIAL(      /* LEGAL THEATRE CHARS.   */
'abcdefghijklmnopqrstuvwxyz '''),
ROLEGAL CHAR(80) VAR INITIAL(
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ ''12()'),
ACLEGAL CHAR(80) VAR INITIAL(
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ .?&()'),
TTLEGAL CHAR(80) VAR INITIAL(
'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ ,''?'),
FA CHAR(1) INITIAL('?'),           /* ITEM-START DELIMITER   */
FC CHAR(1) INITIAL('?'),           /* GROUP-START DELIMITER  */
EBP CHAR(1) INITIAL('p')) STATIC,  /* SMALL 'P' (EBCDIC)     */
SMALLS CHAR(26) STATIC INITIAL('abcdefghijklmnopqrstuvwxyz'),
CAPTITL CHAR(83) VAR,              /* TEMP FOR CAPITAL TITLE */
SYSPRINT PRINT ENV(F(133,133)),
LADA ENV(REGIONAL(3)) KEYED INPUT SEQL,
LADD ENV(REGIONAL(3)) DIRECT INPUT,        /* FOR REREADS. */
ITEM ENV(REGIONAL(3)) OUTPUT SEQL KEYED UNBUF,
LADS INPUT SEQL,
ITSM OUTPUT UNBUF,
ITDM ENV(REGIONAL(3)) DIRECT UPDATE,       /* FOR REWRITES */
ERRMSG ENTRY(CHAR(4),FIXED BIN(15,0)) ,.
DCL            /* DCL STMT FOR WRK & CTRL VARIABLES          */
GTYP CHAR(2),  /* GTYP2 IS DEFINED ON THIS.  GTYP CONTAINS   */
               /* THE CURRENT GRP CONTROL CHARACTERS.        */
GTYP2 CHAR(1) DEF GTYP POS(2),     /* GRP TYPE LETTER.       */
ITYP CHAR(2),              /* CURRENT ITEM DLM CHARS.        */
ITYP1 CHAR(1) DEF ITYP,
ITYP2 CHAR(1) DEF ITYP POS(2),     /* ITEM TYPE LETTER.      */
S CHAR(2000) VAR EXT,      /* PRIMARY INPUT BUF, & WRK BUFFER */
                           /* EXTERNAL FOR OVERLAY SURVIVAL.  */
                           /* SINCE STATIC CSECT IS OVERLAIN. */
Z CHAR(188),               /* OUTPUT BUFFER                   */
(CRTITL CHAR(83 ) VAR,     /* CURRENT SECTION TITLE, IF ANY.  */
ROL  CHAR(83 ) VAR,        /* CURRENT 'ROLE', IF ANY.         */
ACTR   CHAR(83 ) VAR,      /* CURRENT 'ACTOR', IF ANY.        */
TBIT BIT(1),   /* LOCKOUT BIT FOR MORE THAN ONE TIME ENTRY    */
               /* WITHIN ONE GROUP.  2ND ENTRY NOT PROCESSED */
(KITM INIT(0) EXT,         /* CURRENT ITEM OUTPUT FILE BLK    */
MKITM INIT(-1))            /* NUMBER OF BIGGEST BLK YET       */
   FIXED BIN(31,0),        /* WRITTEN OUT, IF ANY.            */
ITI INIT(0),   /* SOFTWARE FLAG INDICATING WHETHER WAIT STMT */
               /* IS REQUIRED FOR ITE EVENT.                 */
K,                         /* HAS KKEY DEFINED ON IT.         */
/* KREPT IS PTR TO ELEMENT OF OLD KEY ARRAY FOR LEVEL 4       */
/* RERUN OF PERFORMANCE.                                      */
(ICPERF,ICSEC,ICGRP) FIXED BIN(31,0),
J FIXED BIN(31,0) INIT(0) EXT) STATIC ,.     /* INPUT BLKS.   */
DCL            /* DCL STMT FOR KEY & DATE MANIPULATION ETC.   */
IREC FIXED BIN(31,0) STATIC INIT(0) EXT,     /* FOR TOTAL     */
                           /* POSSIBLE RECORDS GENERATED, VALID*/
                           /* IF ALL LEVEL 1 INCLUSIVENESS ONLY*/
MANY(2,SUPERSSIZE) STATIC INIT((MANYSSSIZE) 0)      EXT
   FIXED BIN(31,0),
     /* ABOVE IS HIT COUNT + REC COUNT FOR STATISTICS.        */
STRUKEY CHAR(12),          /* INPUT BUF FOR KEY IN STRU READS */
SMOD    CHAR(1) DEF STRUKEY POS(4),
RSOFKEY CHAR(8) DEF STRUKEY POS(5),
STRUDEF BIT(24) DEF STRUKEY,
TRANSBITS BIT(24) INITIAL('111111111111111111111111'B),
KEYBITS BIT(16) DEF TRANSBITS POS(6),
```

```
      IOLDATE FIXED BIN(31,0),
      OLDKEY BIT(16) DEF IOLDATE POS(3 ) ,.
      IF IOPT(4) THEN ICAPO=0 ,.   ELSE ICAPO=1 ,. /* FOR FASTER   */
           /* ACCESS TO OPTION NO. 4.                                 */
      ON ERROR SNAP BEGIN ,.
      PUT FILE(SYSPRINT) SKIP(1) EDIT
      ('ITEMS ERR ONCODE=',ONCODE) (A) ,.
      GOTO ENDPROC ,.   END ,.
      ON ENDFILE(LADS) GOTO ENDPROC ,.
      ON ENDFILE(LADA) GOTO ENDPROC ,.
      IF IOPT(7) THEN DO ,.     /* IF POSSIBILITY OF LEVEL 4 INC-   */
                                /* LUSIVENESS PROCESSING.           */
      OPEN FILE(LADD) TITLE('LADA') ,.
      OPEN FILE(ITDM) TITLE('ITEM') ,.   END ,.
      IF IBUFBIT=0 THEN         /* IF THIS IS THE FIRST TIME THAT   */
                                /* THE ITEMGET OVERLAY HAS BEEN     */
                                /* CALLED.  (PRIMES INPUT BUFFER).  */
INPUT ..              /* READ, OR IF NECESSARY, REREAD A SECTION.   */
      IF LEV='4' AND KREPT LE KTMP THEN DO ,.      /* IF REREAD.   */
      STRUKEY=KEYAR(KREPT) ,.  /* RECORDED KEY FOR REREAD.         */
      READ FILE(LADD) INTO(S) KEY(STRUKEY CAT '00000000') ,.
      KREPT=KREPT+1 ,.   END ,.          /* INCREMENT KEY ARRAY PTR*/
      ELSE DO ,.
      IF IOPT(8) THEN DO ,.     /* IF SEQUENTIAL INPUT OPTION.      */
      READ FILE(LADS) INTO(S) ,.
      STRUKEY=SUBSTR(S,3,12) ,.          /* EXTRACT ENCODED DATE,  */
                                         /* SEC-LETTER, & THEATRE. */
      S=SUBSTR(S,15) ,.         /* RESET S TO GET RID OF KEY        */
      END ,.   ELSE
      READ FILE(LADA) INTO(S) KEYTO(STRUKEY) ,.
      IBUFBIT=1 ,.
      ICSEC=KITM ,.  /* SAVE PTR TO START OF SECTION'S OUTPUT RECS */
      IF SMOD=EBP THEN DO ,.    /* IF NEW PERFORMANCE SECTION.      */
         KTMP=0 ,.   ICPERF=KITM ,.      /* FORGET PREVIOUS ONE.    */
         LEV='1' ,.   END ,.             /* JAM SET LEVEL.          */
      KTMP=KTMP+1 ,.
      KEYAR(KTMP)=STRUKEY ,.
      ITRK(KTMP)=KTRK ,.        /* FOR FUTURE DEVICE FEEDBACK.      */
      J=J+1 ,.   END ,.
      TRANSBITS=STRUDEF ,.      /* KEYBITS NOW CONTAINS THE CODED   */
                                /* DATE FROM STRUKEY (SEE DECLARE   */
                                /* STATEMENTS FOR SIY DEFINING).    */
      IOLDATE=0 ,.              /* CLEAR LEFT HALF OF IOLDATE.      */
      OLDKEY=KEYBITS ,.         /* CONVERSION, PUT CODED DATE INTO  */
                                /* IOLDATE VIA DEFINING.            */
      IF IOLDATE LT IDR(1) THEN GOTO INPUT ,.
                   /* ABOVE STATEMENT SKIPS PROCESSING RECORDS     */
                   /* THAT PRECEDE THE CURRENT DATE RANGE.         */
      IF IOLDATE GT IDR(2) THEN DO ,.
         IF IOPT(10) THEN
         PUT FILE(SYSPRINT) SKIP(1) EDIT
         ('END OF DATE RANGE, EXITING FROM ITEMGET.') (A) ,.
         RETURN ,.   END ,.
      I=LENGTH(S) ,.
      IF IOPT(1) THEN PUT FILE(SYSPRINT) SKIP(1) EDIT
                              ((K DO K=1 TO 9))
                              (SKIP,COL(10),9 (X(9),F(1)))
      ((L,SUBSTR(S,L*100+1,MIN(100,I-1*100))
      DO L=0 TO (I-1)/100)) (SKIP,F(2),COL(10),A(100)) ,.
      S=S CAT FC ,.             /* I NOW=LENGTH(S)-1.               */
      IF LEV='3' THEN LEV='1' ,.
RESEC ..
      CRTITL='' ,.              /* INDICATE NO TITLES DONE YET IN   */
                                /* THIS SECTION.  (PLAY, OPERA, OR  */
                                /* AFTERPEICE TITLES, THAT IS).     */
      IGR=1 ,.        /* INITIAL RIGHT GRP PTR                      */
```

```
      IGROLD=1 ,.     /* RESTART INITIAL RIGHT PTR, (GRP RESTART).  */
      ITP=1 ,.        /* INITIALIZE ICB PTR                         */
      IF IOPT(5) THEN
      PUT FILE(SYSPRINT) SKIP(1) EDIT
      ('INPUT BLK',J,'  LNGTH=',I,'  KEY='''  ,STRUKEY,'''  *** ''',
      SUBSTR(S,1,MIN(60,I)),''' ***') (2 (A,F(4)),5 A) ,.
      GOTO ISTGRP ,.
   GROUPS ..          /* TOP OF LOOP FOR GROUPS, ENTERED AT END OF  */
                      /* INPUT SETUP FOR EACH GRP.                  */
      IF GBN+GSN+GUM GT 0 OR IGR=I+1 THEN DO ,.    /* IF THIS GRP  */
                      /* HAS AT LEAST ONE ROLE, ACTOR, OR TIME ENTRY*/
                      /* OR IF IT'S THE LAST OR ONLY GRP IN THE     */
                      /* SECTION THEN . . . . . .                   */
      ICGRP=KITM ,.   /* SAVE CURRENT NEXTREC OUTPUT PTR IN CASE OF */
                      /* GROUP RESTART.                             */
      IF ITI=1 THEN DO ,.   WAIT(ITE) ,.   ITI=0 ,.   END ,.
      Z=REPEAT(' ',188) ,.
      SECTYPE=SMOD ,.
      IF GUM NE 0 THEN TIME=SUBSTR(S,IP(GUM),IL(GUM)) ,.
                      /* ABOVE STMTS MOVE TIME STRING INTO OUTPUT   */
                      /* BUFFER Z VIA DEFINING, IF IT EXISTS.       */
                      /* THE FOLLOWING PUT STRING CONVERTS THE DATE */
                      /* TO AN 8 BYTE CHAR STRING IN Z.             */
      PUT STRING(DATE8) EDIT
      (IOLDATE/372+1659,TRUNC(MOD(IOLDATE,372)/31)+1,
      MOD(IOLDATE,31)+1) (F(4),2 F(2)) ,.
      THETR=R8OFKEY ,.          /* THEATRE FROM INPUT KEY TO Z BUFR */
      SUBSTR(Z,18,83)=CRTITL ,.
      IR=MAX(GSN,1) ,.
      IRP=GS ,.       /* MOVE PTR TO 1ST ROLE IN GRP INTO A VARIABLE*/
   MOROLES ..
      IF GSN    GT 0 THEN  ROL=SUBSTR(S,IP(IRP),IL(IRP)) ,.
      ELSE  ROL='' ,.
      IA=MAX(GBN,1) ,.
      IAP=GB ,.
   MORACT ..
      IF GBN    GT 0 THEN   ACTR = SUBSTR(S,IP(IAP),IL(IAP)) ,.
      ELSE ACTR='' ,.
      IF LEV GT '1' THEN GOTO PUTREC ,.  /* IF IN A 2ND PASS MODE. */
      IF ICAPO=0 THEN CAPTITL= TRANSLATE(CRTITL,CAPS,SMALLS) ,.
      DO IX=1 TO IAR-1 ,.       /* SCE SEARCH LOOP.                 */
      IF LENGTH(S.ACTOR(IX)) GT 0 THEN
      IF S.ACTOR(IX) NE ACTR THEN GOTO NOHIT ,.
      IF LENGTH(S.ROLE(IX)) GT 0 THEN
      IF S.ROLE(IX) NE ROL THEN GOTO NOHIT ,.
      IF LENGTH(S.TITLE(IX)) GT 0 THEN DO ,.
      IF ICAPO=0 THEN DO ,.
      IF CAPTITL NE S.TITLE(IX) THEN GOTO NOHIT ,.   END ,.
         ELSE DO ,.
      IF CRTITL NE S.TITLE(IX) THEN GOTO NOHIT ,.   END ,.   END ,.
      IF LENGTH(S.SEC(IX)) GT 0 THEN
      IF S.SEC(IX) NE SMOD THEN GOTO NOHIT ,.
      IF LENGTH(S.THEATRE(IX)) GT 0 THEN
      IF S.THEATRE(IX) NE THETR THEN GOTO NOHIT ,.
      MANY(1,IX)=MANY(1,IX)+1 ,.          /* COUNT ACTUAL HITS DUE  */
                                          /* DUE TO SE(IX).         */
      LEV=S.INC(IX) ,.          /* ASSIGN NEW PASS LEVEL            */
      IF LEV='1' THEN GOTO PUTREC ,.      /* RECORD INCLUSIVENESS.  */
      IF LEV='2' THEN DO ,.               /* GROUP INCLUSIVENESS    */
         IGR=IGROLD ,.   KITM=ICGRP ,.
         GOTO REGRP ,.   END ,.
      IF LEV='3' THEN DO ,.               /* SECTION INCLUSIVENESS. */
         KITM=ICSEC ,.   GOTO RESEC ,.   END ,.
      /* SO, LEV MUST EQUAL '4', PERFORMANCE INCLUSIVENESS.         */
      KITM=ICPERF ,.   KREPT=1 ,.
      GOTO INPUT ,.  /* START ANOTHER PASS THROUGH THE PERFORMANCE */
```

```
NOHIT .. END ,.
    IREC=IREC+1 ,.          /* PART OF TOTAL POSSIBLE REC COUNT */
    GOTO MISS ,.
PUTREC ..          /* A HIT HAS OCCURRED, SO PUT OUT THE RECORD. */
    IREC=IREC+1 ,.          /* PART OF TOTAL POSSIBLE REC COUNT */
    MANY(2,IX)=MANY(2,IX)+1 ,.          /* COUNT OUTPUT RECS DUE  */
                                        /* TO HITS ON THIS SE.    */
    IF IOPT(19) THEN
    PUT FILE(SYSPRINT) SKIP(1) EDIT
    (SMOD,IYR,IMON,IDY,TIME,THETR,
    SUBSTR(Z,18,LENGTH(CRTITL)),ROL,ACTR)
    (A(1),X(1),F(4),X(1),F(2),X(1),F(2),
    X(1),A(8),X(1),A(8),COL(32),A,
    COL(69),A,COL(95),A) ,.
    SUBSTR(Z,101,40)=ROL ,.
    SUBSTR(Z,141,40)=ACTR ,.
    K=MOD(KITM,13) ,.          /* PUTS RECORDED KEY INTO KKEY.     */
    PUT STRING(TRKNO) EDIT(KITM/13) (F(8)) ,.
    IF ITI=1 THEN WAIT(ITE) ,.
    IF KITM LE MKITM THEN
    REWRITE FILE(ITDM) FROM(Z) KEY(KKEY CAT TRKNO) EVENT(ITE) ,.
    ELSE IF IOPT(9) THEN                    /* PURE SEQL OUTPUT OPT.  */
    WRITE FILE(ITSM) FROM(Z) EVENT(ITE) ,.
    ELSE WRITE FILE(ITEM) FROM(Z) KEYFROM(KKEY CAT TRKNO)
    EVENT(ITE) ,.
    ITI=1 ,.        /* ITI=1 MEANS WAIT PENDING FOR OUTPUT FILE.  */
    MKITM=MAX(MKITM,KITM) ,. /* MAINTAIN MAX OUT PUT PTR          */
    KITM=KITM+1 ,.           /* INCREMENT OUTPUT BLK PTR          */
    L=VERIFY(THETR  ,THLEGAL) ,.
    IF IOPT(29) THEN
    IF L NE 0 THEN DO ,.
       CALL ERRMSG('ITHE',J) ,.
       PUT FILE(SYSPRINT) EDIT(THETR) (X(2),A) ,.   END ,.
    L=VERIFY(CRTITL ,TTLEGAL) ,.
    IF IOPT(30) THEN
    IF L NE 0 THEN DO ,.
       CALL ERRMSG('ITTL',J) ,.
       PUT FILE(SYSPRINT) EDIT(CRTITL) (X(2),A) ,.   END ,.
    L=VERIFY(ROL    ,ROLEGAL) ,.
    IF IOPT(31) THEN
    IF L NE 0 THEN DO ,.
       CALL ERRMSG('IROL',J) ,.
       PUT FILE(SYSPRINT) EDIT(ROL) (X(2),A) ,.   END ,.
    L=VERIFY(ACTR   ,ACLEGAL) ,.
    IF IOPT(32) THEN
    IF L NE 0 THEN DO ,.
       CALL ERRMSG('IACT',J) ,.
       PUT FILE(SYSPRINT) EDIT(ACTR) (X(2),A) ,.   END ,.
MISS ..
    IAP=IAP+1 ,.
    IA=IA-1 ,.
    IF IA GT 0 THEN GOTO MORACT ,.
    IRP=IRP+1 ,.
    IR=IR-1 ,.
    IF IR GT 0 THEN GOTO MOROLES ,.
    END ,.
ISTGRP ..          /* LOGICAL TOP OF GRP PROCESSING LOOP.          */
    IF IGR=I+1 THEN GOTO INPUT ,.
    IF LEV='2' THEN LEV='1' ,.
REGRP ..
    ITP=1 ,.        /* SET PTR TO FIRST ICE                        */
    TBIT='0'B ,.    /* MEANS 'NO TIME ENTRIES FOUND YET IN GRP.    */
    IGL=IGR ,.      /* LAST BYTE+1 OF PREVIOUS GRP = LEFTMOST      */
                    /* BYTE OF CURRENT/NEW GROUP, IF ANY.          */
    GS,GUM,GB,GSN,GBN=0 ,.    /* RESET GCB                         */
    IGR=INDEX(SUBSTR(S,IGL+1),FC)+IGL ,.          /* FIND GRP-END */
```

```
        IGS=IGR-IGL ,.              /* GET AND SAVE SIZE OF CURRENT GRP */
        IF IGS LT 2 THEN DO ,.      /* IF ZERO LENGTH GRP (OR WORSE).   */
                         CALL ERRMSG('0GRP',IGL) ,.
                         GOTO GROUPS ,.   END ,.
        IF IOPT(3) THEN             /* DEBUG OPTION.                    */
        PUT FILE(SYSPRINT) SKIP(1) EDIT
             ('GRP=''',SUBSTR(S,IGL,IGS)     ,'''')       (3 A) ,.
        GTYP=SUBSTR(S,IGL,2) ,.
        ITR=IGL+2 ,.
    ITEMS ..
        IF ITR=IGR THEN GOTO GROUPS ,.
        ITL=ITR ,.
        ITYP=SUBSTR(S,ITL,2) ,.
        IF ITYP1 NE FA THEN CALL ERRMSG('NEFA',ITI) ,.
        ITR=INDEX(SUBSTR(S,ITL+1,IGR-ITL  ),FA) ,.
        IF ITR=0 THEN ITR=IGR ,.
                 ELSE ITR=ITR+ITL ,.
        IP(ITP)=ITL+2 ,.            /* SAVE PTR TO ITEM                 */
        IL(ITP)=ITR-ITL-2 ,.        /* SAVE LENGTH OF ITEM              */
        IF IOPT(2) THEN
        PUT FILE(SYSPRINT) SKIP(1) EDIT
        ('ITEM = ''',SUBSTR(S,ITL,ITR-ITL),'''') (A) ,.
        IF ITYP2='A' THEN DO ,.             /*$$$$$$$$$ ACTOR $$$$$$$$*/
        GBN=GBN+1 ,.
        IF GBN=1 THEN GB =ITP ,.            /* PTR TO PTR TO 1ST       */
                                /* ACTOR IN A GRP IS NOW SET UP.    */
        GOTO ITEMDONE ,.   END ,.
        IF ITYP2='R' THEN DO ,.             /*$$$$$$$$$$ ROLE $$$$$$$$$*/
        GSN=GSN+1 ,.
        IF GSN=1 THEN GS=ITP ,.
        GOTO ITEMDONE ,.   END ,.
        IF ITYP2='T' THEN DO ,.             /* $$$$$$$ TIME $$$$$$$    */
        IF TBIT THEN DO ,.   CALL ERRMSG('2TIM',ITL) ,.
                     GOTO ITEMS ,.   END ,.
        TBIT='1'B ,.   /* NOTE TIME ENTRY ENCOUNTERED IN CURRENT GRP */
        GUM=ITP ,.
        GOTO ITEMDONE ,.
        END ,.
        IF CRTITL NE '' THEN CALL ERRMSG('2TTL',IP(ITP)) ,.
        CRTITL=SUBSTR(S,IP(ITP),IL(ITP)) ,.
        GOTO ITEMS ,.               /* NO LONGER NEED ICE FOR TITLE.    */
    ITEMDONE ..
        ITP=ITP+1 ,.
        GOTO ITEMS ,.
    ERRMSG .. PROC(NOTE,LOC) ,.
        DCL NOTE CHAR(4) ,.
        PUT FILE(SYSPRINT) SKIP(1) EDIT
        ('****************************** ') (A)
        (NOTE,LOC) (A(4),X(2),F(6)) ,.   END ,.
    ENDPROC ..
        IF KITM LE MKITM THEN CALL ERRMSG('FSEQ',MKITM) ,.
        IFBIT=1 ,.                  /* INDICATE END OF FILE ENCOUNTERED */
        END ITEMGET ,.
/*
//LKED EXEC PGM=IEWL,PARM='LIST,MAP,OVLY,LET'
//SYSPRINT DD SYSOUT=A
//SYSLIB DD DSN=SYS1.PL1LIB,DISP=OLD
//OBJLIB DD UNIT=2311,VOL=SER=GWSJR1,DSN=OBLIB,DISP=OLD
//SYSLMOD DD UNIT=2311,VOL=SER=GWSJR1,DSN=UTLIB,DISP=OLD
//SYSUT1 DD UNIT=2311,DSN=&&SYSUT1,SPACE=(1024,(200,20)),
//            SEP=(SYSLIB,SYSLMOD),DCB=BLKSIZE=1024
//SYSLIN DD *
    INCLUDE OBJLIB(ROOT)
    INCLUDE OBJLIB(CTRLCRD)
    INCLUDE OBJLIB(ITEMGET)
    INSERT ROOT
```

```
     OVERLAY ONE
     INSERT CTRLCRD
     INSERT CTRLCRDA
     OVERLAY ONE
     INSERT ITEMGET
     NAME ITEMS(R)
/*
//FORMAT   JOB 99999211,WCD,MSGLEVEL=(1,1)
//F EXEC LSPUPGEN,MEM=FRMAT
//BAKUP.SYSUT1 DD *,DCB=BLKSIZE=80
     FORMAT .. PROC OPTIONS(MAIN) ,.
         DCL 1 Z,
            5 SECTYPE CHAR(1),
            5 YEAR CHAR(4),
            5 MONTH CHAR(2),
            5 DAY CHAR(2),
            5 THEATRE CHAR(8),
            5 TITLE CHAR(83),
            5 ROLE CHAR(40),
            5 ACTOR CHAR(40),
            5 TIME CHAR(8),
         TITLEB CHAR(83),
         ROLEB CHAR(40),
         ACTORB CHAR(40),
         TITLEA(83) CHAR(1) DEF TITLEB,
         ROLEA(40) CHAR(1) DEF ROLEB,
         ACTORA(40) CHAR(1) DEF ACTORB,
         IOPT(32) BIT(1),
         SYSPRINT PRINT ENV(F(133,133)) ,.
       GET FILE(SYSIN) EDIT(IOPT) (B(1)) ,.
         ON ENDFILE(SRTD) GOTO ENDPROG ,.
     INPUT ..
         READ FILE(SRTD) INTO(Z) ,.
         TITLEB=TITLE ,.   ROLEB=ROLE ,.   ACTORB=ACTOR ,.
         IT=83 ,.
     TITLESIZE ..
         IF TITLEA(IT) NE ' ' THEN GOTO PROLESIZE ,.
         IT=IT-1 ,.
         IF IT GT 0 THEN GOTO TITLESIZE ,.
     PROLESIZE ..
         IR=40 ,.
     ROLESIZE ..
         IF ROLEA(IR) NE ' ' THEN GOTO PRACTORSIZE ,.
         IR=IR-1 ,.
         IF IR GT 0 THEN GOTO ROLESIZE ,.
      PRACTORSIZE ..
         IA=40 ,.
     ACTORSIZE ..
         IF ACTORA(IA) NE ' ' THEN GOTO SIZEDUP ,.
         IA=IA-1 ,.
         IF IA GT 0 THEN GOTO ACTORSIZE ,.
     SIZEDUP ..
         PUT FILE(SYSPRINT) SKIP(1) EDIT
         (SECTYPE,YEAR,MONTH,DAY,TIME,THEATRE,
         SUBSTR(TITLE,1,IT),
         SUBSTR(ROLE,1,IR),SUBSTR(ACTOR,1,IA))
         (7 (A,X(1)),COL(78),A,COL(104),A) ,.
         IF IOPT(1) THEN PUT DATA ,.
         GOTO INPUT ,.
     ENDPROG .. END FORMAT ,.
/*
//TRT      JOB 99999211,WCD,MSGLEVEL=(1,1)
// EXEC LSPUPGEN,MEM=THT,PARM.PL1L='C48,A,X,SIZE=90K,FW,NI',DUTP=OLD
//BAKUP.SYSUT1 DD *
     TRTABLE .. PROC(PRM) OPTIONS(MAIN) ,.
         DCL HXDF CHAR(16) INITIAL('0123456789ABCDEF'),
```

```
      BIT1 BIT(1),
      COMMENTS(4) CHAR(77),
      HX(0 .. 15) CHAR(1) DEFINED HXDF,
      PRM CHAR(100) VAR,
      CARD (80) CHAR(1) INITIAL((14) (1) ' ',''''),
      C1 CHAR(1),
      C2 CHAR(1),
      C3 CHAR(1),
      SYSPRINT PRINT ENV(F(133,133)),
      CFROM CHAR(256),
      CRHALF(2) CHAR(128) DEF CFROM,
      CROMB (256) CHAR(1) DEF CFROM,
      GETDIGIT ENTRY(CHAR(1)),
      TBL(0 .. 255) CHAR(1) INITIAL((256) (1) ' '),
      CNVAL CHAR(1) DEFINED NVAL POSITION(2) ,.
      INCNT,MAP=1 ,.
      L=LENGTH(PRM) ,.
      IF L GE 5 THEN DO ,.
      PRM=PRM CAT ';' ,.
         GET STRING(PRM) DATA(INCNT,MAP) ,.
      END ,.
      DO INCNT=1 TO INCNT ,.
      COMMENTS='' ,.
      IC=1 ,.
      DO NVAL=0 TO 255 ,.
      CROMB(NVAL+1)=CNVAL ,.   END ,.
      I=0 ,.
   SRCHEX ..
      GET FILE(SYSIN) EDIT(C2) (A(1)) ,.
   NEXTEX ..
      IF C2='*' THEN DO ,.
         GET FILE(SYSIN) EDIT(COMMENTS(IC)) (A(77)) ,.
      IC=MIN(IC+1,4) ,.   GOTO SRCHEX ,.   END ,.
      IF C2=' ' OR C2=',' THEN GOTO SRCHEX ,.
      IF C2=';' THEN GOTO PUNCH ,.  /* SEMICOLON DELIMITS END OF   */
                                    /* DATA   */
      IF VERIFY(C2,'0123456789ABCDEF') =1 THEN
      PUT FILE(SYSPRINT) SKIP(1) EDIT('ILLEGAL CHAR',C2)  (A,X(2)) ,.
      C1='0' ,.
      GET FILE(SYSIN) EDIT(C3) (A(1)) ,.
      IF VERIFY(C3,'0123456789ABCDEF')=1 THEN BIT1='1'B ,.
      ELSE DO ,.  BIT1='0'B ,.  C1=C2 ,.  C2=C3 ,.  END ,.
      CALL GETDIGIT(C1) ,.
      NVAL=IVAL*16 ,.
      CALL GETDIGIT(C2) ,.
      NVAL=NVAL+IVAL ,.
      TBL(I)=CNVAL ,.
      IF I=255 THEN GOTO PUNCH ,.
      I=I+1 ,.
      IF BIT1 THEN GOTO NEXTEX ,.
      GOTO SRCHEX ,.
   PUNCH ..
      J=0 ,.
      DO I=1 TO 8 ,.
      DO K=16 TO 71 WHILE(J LT I*32) ,.
      CARD(K)=TBL(J) ,.
      IF CARD(K)='''' THEN DO ,.
      K=K+1 ,.
      CARD(K)='''' ,.
      END ,.
      J=J+1 ,.
      END ,.
      IF J NE I*32 OR K GT 70   THEN DO ,.
      PUT FILE(SYSPRINT) SKIP(1) EDIT
      ('OVERFLOW DUE TO TOO MANY QUOTES,  I=',I) (A,F(4)) ,.
      RETURN ,.
```

```
        END ,.
        CARD(K)='''' ,.
        CARD(K+1)=',' ,.
        DO L=K+2 TO 80 ,.
        CARD(L)=' ' ,.
        END ,.
        PUT FILE(SYSPNCH) EDIT(CARD) (A) ,.
        END ,.
        DO MIP=1 TO MAP ,.
        IF MIP+INCNT GT 2 THEN PUT PAGE ,.
        PUT FILE(SYSPRINT) EDIT(COMMENTS) (CCL(25),A) ,.
        PUT FILE(SYSPRINT) SKIP ,.
        PUT FILE(SYSPRINT) SKIP(1) EDIT((HX(NN) DC NN=0 TO 7))
        (COL(2),8 (A(1),X(15))) ,.
        PUT FILE(SYSPRINT) SKIP(1) EDIT('''',CHHALF(1),'''') (A) ,.
        PUT FILE(SYSPRINT) SKIP(1) EDIT
        ('''',(TBL(NN) DO NN=0 TO 127),'''') (A) ,.
        PUT FILE(SYSPRINT) SKIP(1) EDIT((HX(NN) DC NN=8 TO 15))
        (COL(2),8 (A(1),X(15))) ,.
        PUT FILE(SYSPRINT) SKIP(1) EDIT('''',CHHALF(2),'''') (A) ,.
        PUT FILE(SYSPRINT) SKIP(1) EDIT
        ('''',(TBL(NN) DO NN=128 TO 255),'''') (A) ,.
        PUT FILE(SYSPRINT) SKIP(2) EDIT
        (((CNVAL,HX  (NVAL/16),HX  (MOD(NVAL,16))
        DO NVAL=K TO K+240 BY 16),
        (TBL(NVAL),
        HX(UNSPEC(TBL(NVAL))/16),
        HX(MOD(UNSPEC(TBL(NVAL)),16))
        DO NVAL=K TO K+240 BY 16) DO K=0 TO 15))
        (R(LN),SKIP(1),R(LN),SKIP(2)) ,.
    LN .. FORMAT(16 (X(3),A,X(1),A,A)) ,.
        END ,.
        END ,.

    GETDIGIT .. PROC(C3) ,.
        DCL C3 CHAR(1) ,.
        DO K=0 TO 15 ,.
        IF C3 = HX(K) THEN DO ,.
        IVAL=K ,.
        RETURN ,.
        END ,.
        END ,.
        PUT FILE(SYSPRINT) SKIP(1) EDIT
        ('*****  BAD CHAR INPUT,   ''',C3,'''') (A) ,.
        END ,.
        END ,.
/*
//LNDSTG JOB 99999211,WCD,MSGLEVEL=1
//M EXEC LSPUPGEN,MEM=MAPRINT
//BAKUP.SYSUT1 DD *
    CHARDEM .. PROC OPTIONS(MAIN) ,.
        DCL (C,D) CHAR(240) VAR ,.
        ON ENDPAGE(SYSPRINT) ,.
        GET LIST(J,C) ,.
        D=C ,.
        I=LENGTH(C) ,.
        DO K=1 TO J ,.
        DO L=1 TO I ,.
        DO M=1 TO I ,.
        SUBSTR(D,M,1)=SUBSTR(C,L,1) ,.
        END ,.
        PUT SKIP(1) LIST(D) ,.
        PUT SKIP(0) LIST(C) ,.
        END ,.
        PUT PAGE ,.
        END ,.
```

```
        END ,.
/*
//DOCPRNT JOB 99999221,WCD,MSGLEVEL=(1,1)
//D EXEC LSPUPGEN,MEM=DOCPRINT
//BAKUP.SYSUT1 DD *
     DOCPRNT .. PROC OPTIONS(MAIN) ,.
          DCL CARD CHAR(80) ,.
          DCL CARDBYTE (80) CHAR(1) DEF CARD ,.
          DCL CARD1 CHAR(1) DEF CARD ,.
          ON ENDFILE(SYSIN) GOTO PROGEND ,.
     NEXTCARD ..
          GET FILE(SYSIN) EDIT(CARD) (A(80)) ,.
          IF CARD1='1' THEN PUT PAGE ,.
          IF CARD1='0' THEN PUT SKIP ,.
          IF CARD1='-' THEN PUT SKIP(2) ,.
          IF CARD1='+' THEN PUT SKIP(0) ,.
          IF CARD1='&' THEN DO ,.
          DO I=80 TO 2 BY -1 ,.
          IF CARDBYTE(I) NE ' ' THEN GOTO THEPUT ,.   END ,.
          GOTO NEXTCARD ,.
     THEPUT ..
          PUT EDIT(SUBSTR(CARD,2,I-1)) (A) ,.
          GOTO NEXTCARD ,.
          END ,.
             CARD1=' ' ,.
             PUT FILE(SYSPRINT) SKIP(1) EDIT(CARD) (A) ,.
          GOTO NEXTCARD ,.
     PROGEND ..
          END ,.
/*
//COPYCAT  JOB 99999211,WCD,MSGLEVEL=(1,1)
//C EXEC LSPUPGEN,MEM=COPYCAT,PARM.PL1L='C48,NT,A,X'
//BAKUP.SYSUT1 DD *
     COPYCAT .. PROC(PRM) OPTIONS(MAIN) ,.
          DCL PRM CHAR(100) VAR,.
          DCL CARDIN CHAR(80),CARDOUT CHAR(80) INITIAL((80) ' ') ,.
          ON ENDFILE(SYSIN) GOTO ENDPROG ,.
     NEXTCARD ..
          GET FILE(SYSIN) EDIT(CARDIN) (A(80)) ,.
          IF LENGTH(PRM) GE 1 THEN SUBSTR(CARDIN ,80,1)=PRM ,.
          PUT FILE(SYSPNCH) EDIT(CARDOUT,CARDIN) (2 A(80)) ,.
          GOTO NEXTCARD ,.
     ENDPROG ..
          END ,.
/*
//LSPMAIL  JOB 99999211,WCD,MSGLEVEL=(1,1)
//M EXEC LSPUPGEN,MEM=LSPMAIL
//BAKUP.SYSUT1 DD *
          /* LSPMAIL PROGRAM */ (SUBRG,STRG) ..
     LSPMAIL .. PROC OPTIONS(MAIN) ,.
          ON ERROR SNAP BEGIN ,.
          PUT FILE(SYSPRINT) SKIP(1) LIST('ONCODE=',ONCODE) ,.
          PUT FILE(SYSPRINT) DATA ,.          END ,.
          DCL
          IHESARC ENTRY(FIXED BIN(31,0)),
          IHESRTD ENTRY(CHAR(60),CHAR(28),FIXED BIN(31,0),
          FIXED BIN(31,0),ENTRY,ENTRY),
          IOPT(76) BIT(1),
          IRCNT INIT(0),
          ENT15 RETURNS(CHAR(240)),
          SFIELD CHAR(60),RFIELD CHAR(28),
          SORTREC CHAR(240),
          SORTCRD(3) CHAR(80) DEF SORTREC,
          ISIZE FIXED BIN(31,0) INIT(45056),
          IRTCD FIXED BIN(31,0) ,.
          GET FILE(SYSIN) EDIT(IOPT,IRPT) (76 B(1),F(4)) ,.
```

```
      IF IOPT(1) THEN
      PUT FILE(SYSPRINT) SKIP(1) LIST('ISPMAIL STARTED') ,.
      SFIELD=
      ' SORT FIELDS=(1,40,CH,A) ' ,.
      RFIELD=
      ' RECORD TYPE=F,LENGTH=240 ' ,.
      IBIT=1 ,.
      CALL IHESRTD(SFIELD,RFIELD,ISIZE,IRTCD,ENT15,ENT35) ,.
      IF IOPT(1) THEN
      PUT FILE(SYSPRINT) SKIP(1) LIST('RETURN FROM IHESRTD.') ,.
ENT15 .. PROC    RETURNS(CHAR(240)) ,.
      ICNT=1 ,.   SORTREC=' ' ,.
      IF IBIT=0 THEN DO ,.
      CALL IHESARC(8) ,.
      PUT FILE(SYSPRINT)          EDIT('E15 RET 08') (A) ,.
      RETURN(SORTREC) ,.
     END ,.
      IRCNT=IRCNT+1 ,.
NXTCRD ..
      IF ICNT GE 4 THEN DO ,.
         IAV=40 ,.
         PUT FILE(SYSPRINT) SKIP(1) EDIT
         ('ADDRESS NO.',IRCNT+1,' TOO LONG.') (A,F(4)) ,.
         GOTO FOURSCOR ,.
         END ,.
      GET FILE(SYSIN) EDIT(SORTCRD(ICNT)) (A(80)) ,.
      IAV= INDEX(SORTCRD(ICNT),'*') ,.
      IF IAV GT 0 THEN DO ,.
      IF IAV=1 THEN DO ,.
      IF IOPT(1) THEN
      IF IOPT(1) THEN
      PUT FILE(SYSPRINT)          EDIT('E15 RET 00') (A) ,.
      IBIT=1 ,.
      CALL IHESARC(0) ,.   RETURN ,.   END ,.
      ELSE
FOURSCOR ..  DO ,.
      SUBSTR(SORTCRD (ICNT),IAV,1)=' ' ,.
      IF IOPT(1) THEN
      PUT FILE(SYSPRINT) SKIP(1) EDIT('E15 RET 12') (A) ,.
      CALL IHESARC(12) ,.
      RETURN(SORTREC) ,.   END ,.
      END ,.
      ICNT=ICNT+1 ,.   GOTO NXTCRD ,.   END ,.
ENT35 .. PROC(SORTED) ,.
      DCL SORTED CHAR(240),
     SORTCHR(240) CHAR(1) DEF SORTED,
      SORTDF1 CHAR(40) DEF SORTED,
      SORTDF2(5) CHAR(40) DEF SORTED POS(41) ,.
      IF IOPT(1) THEN
      PUT FILE(SYSPRINT) SKIP(1) LIST('IN ENT35') ,.
      DO L=40 TO 1 BY -1 ,.
      IF SORTCHR(L) NE ' ' THEN GOTO GOTFSIZ ,.   END ,.
      PUT FILE(SYSPRINT) SKIP(2) EDIT('BLANK FIRST LINE.') (A) ,.
GOTFSIZ ..
      IAV=INDEX(SORTDF1,'/') ,.
      DO JJ=1 TO IRPT ,.
      IF LINENO(SYSPRINT) GT 55  THEN PUT FILE(SYSPRINT) PAGE ,.
      IF IAV GT 0 THEN
      PUT FILE(SYSPRINT) SKIP(2) EDIT
      (SUBSTR(SORTDF1,IAV+1,L-IAV),SUBSTR(SORTDF1,1,IAV-1))
      (A,X(1)) ,.
      ELSE PUT SKIP(2) FILE(SYSPRINT) EDIT
      (SORTDF1) (A) ,.
      PUT FILE(SYSPRINT) EDIT((SORTDF2(J) DO J=1 TO 5)) (SKIP(1),A),.
      END ,.
      IF IOPT(2) THEN DO ,.
```

```
         DO J=240 TO 1 BY -1 ,.
         IF SORTCHR(J) NE ' ' THEN GOTO GOTSIZE ,.   END ,.
         PUT FILE(SYSPRINT) SKIP(1) EDIT('BLANK SORTED ADDRESS') (A) ,.
      GOTSIZE ..
       PUT SKIP FILE(SYSPNCH ) EDIT(SUBSTR(SORTED,1,J),'*') (A) ,.
         END ,.
         IF IOPT(1) THEN
         PUT FILE(SYSPRINT) SKIP(1) EDIT('E35 RET 04') (A) ,.
         CALL IHESARC(4) ,.
         END ,.
         END ,.
/*
//GENSPECS JOB 99999211,WCD,MSGLEVEL=(1,1)
//D EXEC LSPRINTS,OPT=CARDOCSN,MEM=DOCGENSP
//PRINT.IN DD DDNAME=INSYS
//PRINT.INSYS DD *
```

GENERAL SPECIFICATIONS

THE PURPOSE OF THE LONDON STAGE PROJECT INPUT SYSTEM IS TO READ IN DATA IN THE FORMAT DESCRIBED IN 'LOGICAL STRUCTURE OF INPUT DATA' AND, AFTER PERFORMING LADDER UPDATES ON 'AS', 'SEE', AND CAST GROUP LADDER ENTRIES, TO PRODUCE A DATA BASE CONSISTING OF SORT RECORDS WITH FIELDS RESERVED FOR THE FOLLOWING ITEMS ON EACH RECORD ..

1) DATE
2) SECTION TYPE
3) THEATRE
4) SYNTACTIC TITLE, IF ANY
5) TIME, IF ANY
6) SYNTACTIC ROLE, IF ANY
7) SYNTACTIC ACTOR, IF ANY

ALL POSSIBLE COMBINATIONS WILL BE PRODUCED IN THESE SORT RECORDS, THAT IS IF THERE IS A CAST GROUP WITH 2 ROLES AND 3 ACTORS THEN 6 SORT RECORDS WILL BE PRODUCED.

SCANNER INPUT DATA FORMAT

SCANNER INPUT IS ORIGINALLY IN THE FORM OF FIXED LENGTH BLOCKS 800 BYTES LONG, DIVIDED INTO 10 LINES OF 80 CHARACTERS EACH. THE LAST 5 CHARACTERS OF EACH LINE ARE SUPPOSED TO ALWAYS BE BLANK. THEY ARE TO BE IGNORED AND DELETED BY THE SCANNER INPUT EDITING PROGRAM, ICISCAN. DIFFERENT BLOCKING IS PERMISSIBLE AS LONG AS THE LRECL IS 80.

BATCHES OF INPUT DATA (ALL THE DATA INPUT TO ONE RUN OF A PROGRAM) MUST START WITH A PERFORMANCE AND MUST CONTAIN ONLY COMPLETE PERFORMANCES IN ORDER TO OBTAIN COMPLETE ACCURATE PROCESSING.

THE OUTPUT IS TO BE U FORMAT BLKS ACCEPTABLE AS INPUT TO A SYNTACTICAL PARSING PROGRAM CALLED STRCT (SHORT FOR 'STRUCTURE').

SINCE THE ICI SCANNER INTERPRETS THE DOUBLE QUOTE CHARACTER AS 2 SINGLE QUOTE CHARACTERS, ICISCAN WILL CONVERT ANY 2 CONSECUTIVE SINGLE QUOTE CHARACTERS IT FINDS INTO ONE DOUBLE QUOTE CHARACTER. OTHER FUNCTIONS OF ICISCAN INCLUDE 'AT' SIGN PROCESSING (DEFINED ELSEWHERE), DELETING MULTIPLE BLANKS EXCEPT FOR 2 AFTER PERIODS, QUESTION MARKS, AND EXCLAMATION POINTS, CHECKING FOR ILLEGAL CHARACTERS, AND REBLOCKING INTO AN INTEGRAL NUMBER OF SECTIONS PER BLOCK, EXCEPT

IN ADDITION, DOUBLE BLANKS WILL BE RETAINED AFTER PERIODS, QUESTION MARKS, OR EXCLAMATION POINTS WHEN ANY OF THESE IS FOLLOWED IMMEDIATELY BY EITHER SINGLE OR DOUBLE QUOTES IN THE LOGICAL TEXT.

STARTING WITH THE FIRST, EVERY THIRD BLK OF SCAN INPUT MUST START WITH THE TYPIST'S INITIAL AND A PAGE NUMBER. THE ENTRY MUST BE THE VERY FIRST PRINTED CHARACTERS ON EACH TYPED PAGE, AND MUST BE IN THE FOLLOWING FORMAT ..

1) AN ASTERISK
2) A CAPITAL LETTER
3) 1 TO 5 DECIMAL DIGITS

4) A BLANK

TYPIST ENTRIES ON ON A LINE WILL BE PROCESSED BEFORE ANY 'AT' SIGN PROCESSING, SO TYPIST ENTRIES MUST BE CORRECT WITHOUT THE HELP OF 'AT' SIGNS. THIS SHOULD BE EASY SINCE ONLY THE FIRST FEW CHARACTERS OF THE PAGE ARE INVOLVED.

IN INPUT TO THE SCAN PROGRAM THE DELETIONS DEFINED BY 'AT' SIGNS WILL BE PERFORMED LINE BY LINE IN THE FOLLOWING ORDER ..

1) IF 3 CONTIGUOUS 'AT' SIGNS ARE FOUND ANYWHERE ON THE LINE THEN THE ENTIRE LINE WILL BE DELETED FORTHWITH.
2) IF 2 CONTIGUOUS 'AT' SIGNS ARE ENCOUNTERED THEN THE 'AT' SIGNS AND ALL CHARACTERS BETWEEN THE 'AT' SIGNS AND THE FIRST BLANK TO THE LEFT OF THE 'AT' SIGNS WILL BE DELETED.
   THE FIRST BLANK TO THE LEFT OF THE 'AT' SIGNS WILL NOT BE DELETED.
   THIS WILL BE DONE LEFT TO RIGHT ACCROSS THE LINE REPEATEDLY UNTIL NO DOUBLE 'AT' SIGNS ARE LEFT.
3) IF AN ISOLATED 'AT' SIGN IS THEN FOUND THEN THE 'AT' SIGN AND THE CHARACTER TO ITS LEFT WILL BE DELETED.
   THIS WILL BE DONE LEFT TO RIGHT ACCROSS THE LINE REPEATEDLY UNTIL NO 'AT' SIGNS ARE LEFT.

SUBTLETIES NOT DONE BY CHINA DATA ..

1) ABNORMAL (UNCERTAIN OR NONEXISTENT) DATES.
2) SOME BOX TAGGING, ESPECIALLY VERY ABNORMAL BOXES.
3) SUBTITLE TAGGING.
4) LADDER CHANGES EXCEPT FOR REPLACEMENT AND ADDITION AS IS MOST COMMONLY FOUND IN THE TEXT.
   (THAT IS NO SIGNED LADDER CHANGES)
5) INDEX TAGGING
6) TIME NOTATION

0 ITEMSGET SELECTION/EXCLUSION SPECIFICATIONS

****

THE ITEMGET PROGRAM WILL HAVE THE ABILITY TO SELECT INDIVIDUAL SORT RECORDS, GROUPS OF SORT RECORDS, SECTIONS OF SORT RECORDS, ENTIRE PERFORMANCES OF SORT RECORDS, AND TO EXCLUDE SORT RECORDS OUTSIDE OF CERTAIN DATE RANGES. ALL OF THE PRECEDING CAPABILITIES MAY BE COMBINED ALTHOUGH IT IS ENVISIONED THAT ONLY 1 OR 2 CAPABILITIES WILL BE USED IN ANY ONE RUN. SELECTION MAY BE BASED ON ANY COMBINATION OF THE ITEMS PRESENT IN A SORT RECORD. THESE ITEMS ARE .. SECTION TYPE LETTER, DATE, THEATRE, TITLE, TIME ENTRY, ROLE, AND ACTOR.

PROCESSING TO BE DONE FOR INDIRECT SORTS ..

1) MR, MRS, MISS, MLLE, SG, SGA, QUOTES, AND CAPITAL FOLLOWED BY A PERIOD WILL BE PLACED BEHIND THE ACTUAL NAME, INCLUDING COMBINATIONS OF THESE. SPACES WILL THEN BE TAKEN OUT AND ALL LETTERS CONVERTED TO UPPERCASE.
   LEADING 'THE', 'A', AND 'AN' WILL BE TREATED SIMILARLY FOR TITLES.
   SUFFIXES WILL NOT BE AFFECTED.

1 *** THE LOGICAL STRUCTUR OF INPUT DATA ***

****

THE LARGEST COMPONENT OF INPUT DATA IS THE S/360 LOGICAL RECORD, REFERRED TO HEREAFTER IN THIS REPORT AS A BLK (BLOCK). FOR SAFETY INPUT EACH BLK SHOULD CONSIST OF ONE OR MORE COMPLETE SECTIONS, ALSO CALLED BOXES. EXAMPLES OF SECTIONS ARE .. PLAY SECTIONS, AFTERPEICE SECTIONS, DANCE SECTIONS, OPERA SECTIONS, SONG SECTIONS, ETC. THE SECTIONS ARE GROUPED INTO PERFORMANCES, WHICH CONSIST OF ONE PLAY SECTION (MAINPEICE PLAY) & ALL THE OTHER SECTIONS THAT FOLLOW THE PLAY SECTION AND PRECEDE THE NEXT PLAY SECTION. ANOTHER WAY TO THINK OF A PERFORMANCE IS 'ALL OF THE BOXES ASSOCIATED WITH ONE THEATRE ON ONE DATE'.

ALL OF THE ABOVE ALSO APPLIES TO SCANNER INPUT EXCEPT THAT THERE IS NO RESTRICTION ON SECTIONS BEING CONTAINED WITHIN ONE SCANNER INPUT BLK, ETC.

IT SHOULD BE NOTED THAT SOMETIMES A PLAY SECTION IS USED FOR SPECIAL PURPOSES OTHER THAN PROVIDING INFORMATION ABOUT A PLAY

GIVEN AT A CERTAIN THEATRE ON A CERTAIN DATE. THESE SPECIAL USES OF THE PLAY SECTION WILL BE DESCRIBED BELOW.

EACH SECTION STARTS WITH A THREE CHARACTER SEQUENCE THAT DEFINES BOTH THE START OF A SECTION AND THE END OF A PREVIOUS SECTION IF THERE WAS A PREVIOUS ONE IN THE SAME BLK. THE 1ST CHAR OF THIS TAG IS A BLANK. THE SECOND CHAR IS A UNIQUE CHARACTER THAT IS NEVER TO OCCUR FOLLOWING A BLANK AND IMMEDIATELY PRECEDING A SMALL LETTER ANYWHERE IN THE TEXT EXCEPT TO DEFINE THE START OF A SECTION. IT IS CALLED THE SECTION TAG CHARACTER. FOR MTST INPUT THIS CHARACTER IS THE PREFIX. FOR SCANNER INPUT AN ASTERISK IS USED. THE THIRD CHARACTER OF THE TAG IS A SMALL LETTER. ONLY CERTAIN LETTERS ARE ALLOWED. THESE ARE LISTED ELSEWHERE.

ALL SECTIONS EXCEPT COMMENT SECTIONS MAY CONTAIN 2 KINDS OF TEXT, STRUCTURED TEXT, AND EXTRANEOUS TEXT. A COMMENT SECTION CONSISTS ENTIRELY OF EXTRANEOUS TEXT. EXTRANEOUS TEXT CONTAINS ONLY ONE KIND OF ITEM OF SIGNIFICANCE TO PROGRAMMING, THE SIU TAGGED INDEX ENTRY. INDEX ENTRIES OCCUR ONLY IN EXTRANEOUS TEXT. THEY ARE DELIMITED ON THE LEFT EITHER BY A '$' OR BY A '+', AND ON THE RIGHT BY AN EQUAL SIGN FOLLOWED BY 1 OR 2 TAG CHARACTERS. IF THERE ARE 2 TAG CHARACTERS THEN THE FIRST ONE IS A DECIMAL DIGIT. IF THE TAG IS A BLANK THEN THE TYPE OF INDEX ENTRY IS UNKNOWN. OTHERWISE THE TAG INDICATES SEMANTICALLY THE TYPE OF INDEX ENTRY, EG., ROLE, AUTHOR, STREET NAME, ETC. THE '$' TYPE INDEX ENTRY INDICATES A PERSON OR PLACE ,. THE '+' INDICATES AN ITALICISED INDEX ENTRY.

INDEX ENTRIES WITH A DOUBLE CHARACTER TAG MAY NOT EXCEED 125 CHARACTERS IN LENGTH. SIMILARLY INDEX ENTRIES WITH ONE-CHARACTER TAGS MAY NOT EXCEED 124 CHARACTERS IN LENGTH. BOTH THE ABOVE LIMITS INCLUDE THE INDEX DELIMITERS AND THE EQUAL SIGN.

FOR SAVEDT INDEX ENTRIES ALWAYS START WITH A BLANK-PREFIX-LARGE LETTER OR QUOTE. AN UNTAGGED OR UNKNOWN TYPE OF INDEX ENTRY CAN END WITH ANOTHER PREFIX-SMALL LETTER OR QUOTE. A TAGGED INDEX ENTRY ON THE MTST WILL BE INDICATED BY AN EQUALS SIGN AS THE RIGHT-HAND DELIMITER WITH THE TAG IMMEDIATELY FOLLOWING THE EQUALS SIGN. A BLANK IN THIS LOCATION IS ANOTHER WAY OF INDICATING AN UNKNOWN TYPE OF INDEX ENTRY, AS WITH THE SCANNER.

EXCEPT FOR COMMENT SECTIONS EXTRANEOUS TEXT IS DEFINED IN EITHER OF THE FOLLOWING 2 WAYS ..

1) ON THE LEFT A LEFT BRACKET, AND ON THE RIGHT BY ONE OF THE FOLLOWING ..
   A) FIRST OCCURRENCE OF A PERCENT SIGN FOLLOWING THE LEFT BRACKET.
   B) FIRST OCCURENCE OF A RIGHT BRACKET FOLLOWING THE LEFT BRACKET PROVIDED THAT ANOTHER LEFT BRACKET DOES NOT LIE BETWEEN THE ORIGINAL LEFT BRACKET AND THE RIGHT BRACKET.
2) ON THE LEFT WITH A LEFT PAREN, AND ON THE RIGHT WITH A RIGHT PAREN.

EXTRANEOUS TEXT DELIMITED IN ONE OF THESE 2 WAYS MAY CONTAIN DELIMITERS USED BY THE OTHER METHOD,. THAT IS, FOR EXAMPLE, EXTRANEOUS TEXT DELIMITED BY BRACKETS MAY CONTAIN PARENTHESES.

THE DELIMITERS ARE CONSIDERED PART OF THE EXTRANEOUS TEXT. EACH PAIR OF DELIMITERS MUST DEFINE EXTRANEOUS TEXT TO BE ENTIRELY WITHIN ONE SECTION. IF THERE IS NO MATCHING (RIGHT PAREN FOR LEFT PAREN, RIGHT BRACKET OR PERCENT SIGN FOR LEFT BRACKET) RIGHT DELIMITER FOLLOWING THE LEFT DELIMITER IN THE SAME SECTION THEN THE ENTIRE REMAINDER OF THE SECTION FOLLOWING THE LEFT DELIMITER WILL BE CONSIDERED EXTRANEOUS TEXT, HOWEVER AN ERROR MESSAGE WILL BE GENERATED.

PAGE ENTRIES ARE A WAY OF INFORMING THE COMPUTER OF WHAT PAGE OF THE LONDON STAGE ANY ITEM OF TEXT CAME FROM. PAGE ENTRIES ARE IDENTIFIED ANYWHERE IN THE TEXT BY A ' P' (BLANK-SMALL P) FOLLOWED IMMEDIATELY BY AN UNSIGNED DECIMAL NUMBER WHICH IS ENDED BY A BLANK ON THE RIGHT .. FOR EXAMPLE ' P132 '. NOTE THAT THIS PAGE ENTRY IS 6 CHARACTERS LONG INCLUDING THE BLANKS, WHICH ARE IMPORTANT AND NECESSARY.

PAGE ENTRIES CAN OCCUR IN FRONT OF MOST ITEMS, HOWEVER THE FOLLOWING RESTRICTIONS AND CONVENTIONS MUST BE FOLLOWED ..

A) A PAGE ENTRY MAY NOT OCCUR INSIDE A DATE ENTRY, IT MUST EITHER PRECEDE THE ENTIRE DATE OR ELSE PRECEDE THE THATRE ENTRY WHICH FOLLOWS THE DATE.

B) SIMILARLY PAGE ENTRIES CANNOT OCCUR INSIDE A LADDER ENTRY.
C) PAGE ENTRIES MAY NOT FOLLOW A 'BUT' - THAT IS 'BUT P312 ITEM' IS NOT ALLOWED, EVEN THOUGH THE PAGE ENTRY DOES IMMEDIATELY PRECEDE AN ITEM.
D) THE PAGE ENTRY MUST IMMEDIATELY PRECEDE THE ITEM, SEPARATED FROM THE ITEM ONLY BY ONE OR MORE BLANKS OR THE EQUIVALENT (CARRIAGE RETURNS, LINEFEEDS, AND MTST RUBOUT CHARACTERS).
E) NOTE THAT A SECTION DELIMITER SEQUENCE OR ANY OTHER TYPE OF PUNCTUATION IS NOT AN ITEM.
F) PAGE ENTRIES MAY NOT FOLLOW A TIME ENTRY. INSTEAD THEY MUST EITHER PRECEDE THE TIME ENTRY OR PRECEDE THE ITEM THAT FOLLOWS THE ITEM THAT FOLLOWS THE TIME ENTRY.

ROMAN PAGE ENTRIES, ANALOGOUS TO ORDINARY PAGE ENTRIES, ARE USED IN EXTRANEOUS TEXT DERIVED FROM INTRODUCTIONS. IT IS SIMILAR TO THE ORDINARY PAGE ENTRY IN THAT IT IS PRECEDED AND FOLLOWED BY AT LEAST ONE BLANK. IT IS SYNTACTICALLY DEFINED BY A BLANK-EXCLAMATION POINT COMBINATION IMMEDIATELY PRECEDING A SERIES OF ONE OR MORE LEGITIMATE ROMAN NUMERAL CHARACTERS AND ENDING WITH A BLANK. EXCEPT THAT IT IS ALLOWED ONLY IN INTRODUCTIONS, IT MAY OCCUR ANYWHERE THAT AN ORDINARY PAGE ENTRY COULD OCCUR IN A COMMENT SECTION.

THERE ARE 4 BASIC KINDS OF SECTIONS ..

1) PLAY SECTIONS. THIS KIND OF SECTION IS REQUIRED TO HAVE A THEATRE PRECEDING ITS TITLE. IT MAY ALSO HAVE ONE OR MORE PARTS OF A DATE PRECEDING THE THEATRE.
2) SIMPLE TITLED SECTIONS DO NOT HAVE A DATE, THEATRE, OR TITLE TIME ENTRY. ALL TITLED SECTIONS EXCEPT PLAY SECTIONS MAY BE SIMPLE TITLED SECTIONS.
3) TITLE SECTIONS WITH TITLE TIME ENTRIES HAVE A TIME ENTRY PRECEDING THE TITLE OF THE SECTION ,. THE TIME ENTRY IS OF THE SAME FORM AS FOR CAST GROUP TIME ENTRIES, THAT IS THEY MUST PRECEDE THE TITLE IN A SECTION , AND ARE SEPARATED FROM THE TITLE BY THE COLON-BLANK DEFINING THE END OF THE TIME ENTRY. WHEN A TITLE TIME ENTRY IS ENCOUNTERED THE EFFECT IS ..
   A) ALL GROUPS IN THE CAST LIST WITH AT LEAST ONE ROLE OR ACTOR HAVE THE TIME ENTRY APPENDED TO THE GROUP.
   B) IF THERE ARE NO ROLES OR ACTORS IN THE CAST LIST THEN A GROUP CONTAINING ONLY THE TIME ENTRY WILL BE APPENDED TO THE END OF THE CAST LIST.
4) UNTITLED SECTIONS DO NOT HAVE SECTION TITLES, DATE ENTRIES, OR THEATRE ENTRIES. IF ANY OF THESE SORTS OF ITEMS HAPPEN TO BE IN AN UNTITLED SECTION THEN IT WILL BE TREATED AS SOMETHING IN A CAST GROUP.

THERE MAY BE MORE THAN ONE TITLED SECTION ONE ONE DATE AT ONE THEATRE (IN ONE PERFORMANCE), BUT THERE MAY NOT BE MORE THAN ONE EACH OF ANY UNTITLED KIND OF SECTION. THERE MUST BE EXACTLY ONE PLAY SECTION, AND OF COURSE, IT COMES FIRST IN EACH PERFORMANCE.

THE FOLLOWING SECTION TYPES ARE SPECIFIED ..

P	PLAY	PLAY	MUST HAVE THEATRE & TITLE, MAY HAVE DATE ENTRY ,. NO TIME ENTRIES OF ANY TYPE ALLOWED AT ALL.
A	AFTERPEICE	TITLED	NO TIME ENTRIES OF ANY TYPE ALLOWED AT ALL.
C	COMMENT	EXTRANEOUS	MAY CONTAIN SIU INDEX ENTRIES.
D	DANCE	UNTITLED	
S	SONG	UNTITLED	
M	MUSIC	UNTITLED	
E	ENTERTAINMENT	UNTITLED	
I	INSTRUMENTAL	TITLED	
B	BALLET	TITLED	
T	TRICK	TITLED	
U	MONOLOGUE WITH PARTS	TITLED	
O	OPERA	TITLED	

AN OPERA SECTION IS REALLY A SONG SECTION HAVING A SYNTACTICAL TITLE AND CAST LIST INSTEAD OF JUST A CAST LIST. IT WILL NOT BE DISTINGUISHED BY CHINA DATA FROM OTHER SONG SECTIONS.

ANY OF THE FOLLOWING IS TO BE CONSIDERED A GROUP OF STRUCTURED INPUT ..

PERFORMANCE HEADER - DATE, IF ANY, AND THEATRE (A PAGE ENTRY IS GENERALLY NOT CONSIDERED A PART OF THE GROUP).
TITLE OF A SECTION
BALLET PART-BALLERINA/BALLERINO
MUSICAL INSTRUMENT-MUSICIAN
OPERA PART-OPERATOR
TRICK PART-TRICKSTER
ROLE-ACTOR GROUPS
ENTERTAINMENT-ENTERTAINER GROUPS
SONG-SINGER GROUPS
DANCE-DANCER GROUPS
MUSIC-MUSICIAN GROUPS
'AS' TYPE LADDER REFERENCES
'SEE' TYPE LADDER REFERENCES

FOR THE PURPOSES OF THE TYPIST ANY OF THE FOLLOWING THINGS IS AN ITEM ..

ROLE
ACTOR
TIME ENTRY
SYNTACTIC TITLES
SYNTACTIC SUBTITLES
(PLAY, AFTERPEICE,OPERA,BALLET,TRICK, ETC.)
OPERA
BALLET
INSTRUMENTAL MUSIC
TRICK
THEATRE
EACH PART OF A DATE (BUT FOR SOME PURPOSES THE ENTIRE DATE IS CONSIDERED TO BE AN ITEM)
LADDER REFERENCE
INDEX ENTRY
DANCE
DANCER
MUSICAL PEICE
MUSICIAN
SONG
SINGER
ENTERTAINMENT
ENTERTAINER

THE UNTITLED TYPES OF SECTIONS (E,D,M,S) MAY EITHER CONTAIN A CAST LIST OR THEY MAY BE EMPTY, EXISTING ONLY TO SHOW THAT THERE WAS SOME OF THEIR PARTICULAR TYPE OF ACTIVITY ON THE GIVEN DATE AT THE GIVEN THEATRE.

A SYNTACTIC TITLE MUST ALWAYS BE PRESENT IN A TITLED SECTION. IF THE TITLE IS NOT KNOWN THEN EITHER 'NONE' OR 'LADR' MUST BE PUT IN AS THE TITLE. IF SUCH A SECTION HAS A TITLE LADDER REFERENCE THEN 'LADR' IS USED, OTHERWISE 'NONE' IS USED. THESE ARE SPELLED WITH THE FIRST LETTER CAPITALIZED, THE REST SMALL. THEY ARE USED BOTH TO INDICATE THESE SPECIAL CASES AND TO SATISY THE REQUIREMENT THAT EACH TITLED SECTION HAVE A SYNTACTIC TITLE.

THE SAME THING GOES FOR THEATRE ENTRIES IN PERFORMANCE SECTIONS, EXCEPT THAT 'NONE' IS ALWAYS USED, AND IT IS TYPED ENTIRELY IN SMALL LETTERS. SIMILARLY, IF ALL OF PART OF A DATE IS UNKNOWN THEN '0' MUST BE PUT IN FOR EACH UNKNOWN PART. THE VALUE OF EACH PART OF A DATE THAT IS WRITTEN AS ZERO REMAINS THE SAME AS IT WAS BEFORE, BUT THAT PART OF THE DATE IS MARKED IN THE OUTPUT DATA AS BEING ACTUALLY UNKNOWN. THE ENTIRE DATE THEN SERVES ONLY AS AN ESTIMATE OR GUESS FOR THE REAL DATE. SINCE NO PART OF A DATE THAT IS WRITTEN AS ZERO IS CHANGED, YOU MUST WRITE SUCCESSIVE UNCERTAIN DATES EXPLICITLY WITH 0'S IN THE UNCERTAIN PARTS OF THE DATE.

NO LADDER REFERENCE CAN REFER TO A PERFORMANCE THAT HAS AN UNCERTAIN DATE, THEATRE, OR TITLE.

LADDER REFERENCES ARE OF THE FORM ..

'AS/SEE XX MONTH (XXXX)'

WHERE 'XX' IS A NUMBER BETWEEN 1 AND 31 REPRESENTING THE DAY OF THE MONTH, 'MONTH' IS A 3 TO 5 CHARACTER STRING REPRESENTING THE MONTH OF THE YEAR, AND 'XXXX' IS A NUMBER REPRESENTING THE YEAR. THE PARENTHESES AROUND THE YEAR INDICATE THAT THE YEAR PART IS OPTIONAL. IF THE YEAR IS NOT ENTERED THEN THE CURRENT YEAR FROM WHICH THE LADDER REFERENCE IS BEING MADE IS TAKEN TO BE THE YEAR REFERRED TO. BOTH 'AS ' AND 'SEE ' MAY START WITH EITHER LARGE OR SMALL LETTERS. A PERIOD MAY FOLLOW THE MONTH AND/OR YEAR PARTS, BUT NO OTHER PUNCTUATION, ESPECIALLY COMMAS, SHOULD APPEAR BETWEEN THE PARTS OF A LADDER REFERENCE.

NO SECTION MAY HAVE MORE THAN ONE TITLED TYPE LADDER REFERENCE IN IT. NO TITLE TYPE LADDER REFERENCE CAN BE PRECEDED BY ANY GROUPS IN THE SAME CAST LIST. OF COURSE IT MAY BE PRECEDED BY A TITLE, THEATRE, OR DATE ENTRY IN THE CASE OF A PERFORMANCE SECTION OR A TITLE IN THE CASE OF A TITLED SECTION.

THE END OF A LADDER ENTRY MAY BE DELIMITED EITHER BY A SEMICOLON, COMMA, MINUS SIGN, OR END OF SECTION, DEPENDING ON CIRCUMSTANCES.

'AS' TYPE LADDER REFERENCES WILL BE DONE AS DESCRIBED IN THE 5 MAY OCRB II 'TYPING INSTRUCTIONS FOR THE LONDON STAGE PROJECT'. THE LADDER CHANGE CAPABILITIES ARE TO BE ..

1) UNATTACHED ROLE SUBGROUP ADD
2) UNATTACHED ROLE SUBGROUP DELETE
3) UNATTACHED ACTOR SUBGROUP ADD
3) UNATTACHED ACTOR SUBGROUP DELETE
5) GROUP DELETE (PAIRED OR UNPAIRED)
6) ACTOR SUBGROUP & ROLE SUBGROUP REPLACE
7) ATTACHED ROLE ADD NO QUESTIONS ASKED
8) ATTACHED ROLE DELETE
9) ATTACHED ACTOR ADD NO QUESTIONS ASKED
10) ATTACHED ACTOR DELETE
11) PAIRED GROUP ADD / SAME FORMAT AS REPLACE

'SEE' TYPE TITLE LADDER ENTRIES

1) ALL FIRM GROUPS (THOSE THAT ARE NOT BROUGHT FORWARD BY THE LADDER REFERENCE) WILL REMAIN AS THEY ARE, ALL ITEMS AND GROUPS BEING CONSIDERED UNSIGNED.
2) A MATCH WILL BE ATTEMPTED ON EACH ROLE IN THE FIRM GROUPS. IF THE MATCHING ATTEMPT SUCCEEDS FOR ANY ROLE THEN THAT ROLE IS DELETED FROM THE QUESTIONABLE SET BEING BROUGHT FORWARD ,. IF THE ROLE DELETED IS THE ONLY ROLE IN ITS GROUP THEN THE ENTIRE GROUP IS DELETED.

CAST GROUP LADDER ENTRIES

1) MUST IMMEDIATELY PRECEDE THE FIRST DASH IN THE GROUP IF THE GROUP HAS A DASH IN IT (BECAUSE EVERY ITEM FOLLOWING THE CAST GROUP LADDER REFERENCE IS A SYNTACTIC ACTOR, NO USE CONFUSING PEOPLE).
2) KEYING IS DONE ON ALL SYNTACTIC ROLES IN THE GROUP DOING THE REFERRING. IF SOME UNMATCHED ROLES ARE STILL LEFT OVER IN THE FIRST GROUP HAVING A MATCH IN THE SECTION REFERRED TO THEN NO ERROR IS IS GENERATED. IF ANY OF THE THE ROLES IN THE REFERRING GROUP DO NOT FIND A MATCH IN THE FIRST GROUP FOUND HAVING ANY MATCH IN THE REFERRED TO SECTION THEN AN 'INKY' ERROR MESSAGE IS ('INKY' MEANS 'INCOMPLETE KEYING IN GROUP IN WHICH SOME KEYING HAS ALREADY BEEN ACHEIVED') GENERATED. IF NO MATCHES AT ALL ARE FOUND THEN A 'NCGM' ERROR ('NCGM' MEANS THAT NO MATCH FOR ANY ROLE IN THE LADDER GROUP WAS FOUND IN THE REFERRED TO SECTION) MESSAGE IS GENERATED. THE SUCCESS OF THIS SCHEME DEPENDS ON

NOT HAVING A SYNTACTIC ROLE MATCHING ANY OF THE ROLES IN THE REFERRING GROUP PRESENT PRECEDING THE ACTUAL GROUP ON WHICH THE KEYING IS DESIRED IN THE REFERRED TO SECTION.
RELATIVELY MINOR PROGRAMMING CAN SUBSTANTIALLY ELIMINATE THIS RESTRICTION.

3) ONLY 'AS' TYPE REFERENCES ARE ALLOWED.

4) TIME ENTRIES WILL BE BROUGHT FORWARD INTACT UNLESS THE CURRENT GROUP HAS ITS OWN .

5) ARE ALLOWED ONLY IN UNTITLED SECTIONS

6) AFTER SUCCESSFUL KEYING IS ACHEIVED, THEN ANY ACTOR IN THE REFERRED-TO GRP THAT IS MATCHED BY AN ACTOR PRECEDED BY A MINUS SIGN IN THE REFERRING GRP IS DELETED, ALONG WITH THE ACTOR PRECEDED BY THE MINUS SIGN IN THE REFERRING GRP, OF COURSE.
IF THERE IS AN ACTOR PRECEDED BY A MINUS SIGN IN THE REFERRING GRP THAT HAS NO MATCH IN THE REFERRED-TO GROUP THEN A 'CGAD', 'CAST GROUP ACTOR DELETE' ERROR MESSAGE WILL BE GENERATED. (NOT YET IMPLEMENTED).

7) THE RESULT OF A SUCCESSFUL MATCH IS ALL OF THE SYNTACTIC ACTORS IN BOTH GROUPS, INCLUDING POSSIBLE REPETITIONS WITH NO QUESTIONS ASKED, EXCEPT FOR DELETIONS AS NOTED ABOVE. REPETITIONS MAY DELETED WITH RELATIVELY MINOR PROGRAMMING LATER.

1 PROGRAMMING SPECIFICATIONS
****

SPECIFICATIONS TO BE DEVELOPED ..

1) SYSTEM FOR TAGGING SUBTITLES TO BE USED IN LADDER KEYING.

2) SYSTEM FOR PATCHING, IF ANY.

NOTES ..

1) UNTITLED SECTIONS MAY HAVE 'TITLE' TYPE LADDER REFERENCES.
NATURALLY, ONLY ONE UNTITLED SECTION OF ANY ONE TYPE IS ALLOWED IN ANY PERFORMANCE.

2) NO REPEATED TITLES, INCLUDING 'NONE', ARE ALLOWED IN ANY PERFORMANCE (FOR THE SAME TYPE OF SECTION).

3) IN LADDER UPDATES TIME ENTRIES WILL BE BROUGHT FORWARD INTACT UNLESS THE CURRENT GROUP ALREADY HAS A TIME ENTRY.

4) THE MAXIMUM BLKSIZE ALLOWED FOR ANY OF THE PROGRAMS IS GUARANTEED TO BE LESS THAN OR EQUAL TO 3625, HOWEVER THE DATA LENGTH OF BLOCKS IN THE LADDER PROGRAM IS ALLOWED TO BE AS SMALL AS 3593 BYTES INCLUDING CONTROL CHARACTERS. LARGER SECTIONS THAN THIS WILL REQUIRE SPECIAL EDITING AND SMALL TO MEDIUM SIZED CHANGES IN THE EDT, SAVEDT, STRUCTUR, LADDER, & ITEMSGET PROGRAMS.

5) A SLASH WILL BE USED TO INDICATE THE END OF A LINE OF POETRY IN ALL CASES. A SLASH WILL BE USED TO INDICATE THE FIRST LINE OF A POEM. A SLASH HAS NO PROGRAMMING SIGNIFICANCE.

6) IN THE FUTURE ITALICS THAT DO NOT SIGNIFY TITLES WILL PROBABLY BE WIPED OUT.

7) TIME ENTRIES ARE NOT ALLOWED IN PLAY OR AFTERPIECE SECTIONS, BUT AN ERROR MESSAGE WILL NOT NECESSARILY BE GENERATED ON DETECTING A TIME ENTRY IN THESE SECTIONS.

FOR THE MTST SYSTEM THERE ARE 7 PROGRAM STEPS IN THE DATA ENTRY AND RETRIEVAL CYCLE ..

1) EDT
2) SAVEDT
3) STRUCTUR
4) LADDER
5) ITEMSGET
6) SORT
7) FORMAT

A PATCH STAGE WOULD PROBABLY WORK BEST FOLLOWING THE LADDER STEP. PERHAPS PATCHES COULD BE IN A FORMAT LIKE THE LADDER CHANGES THEMSELVES.

*** DEFINITIONS ***
****

BALLET SECTION A TITLED SECTION TAGGED WITH THE SMALL LETTER 'B'. ITS DATA SHOULD COME FROM A DANCE BOX.

CAST GROUP IS A CHAR STRING IN THE CAST LIST PART OF A SECTION. IT IS DELIMITED ON THE RIGHT EITHER BY THE END OF THE SECTION OR BY A SEMICOLON. IT IS DELIMITED ON THE LEFT BY EITHER THE BEGINNING OF THE CAST LIST OR THE END OF THE PREVIOUS GROUP (BY A SEMICOLON).

CAST GROUP LADDER ENTRY A LADDER ENTRY IN AN UNTITLED SECTION THAT FOLLOWS A SYNTACTIC ROLE IN A CAST GROUP AND IMMEDIATELY PRECEDES THE FIRST DASH IN THE GROUP IF THE GROUP HAS A DASH.

CAST LIST ITEM TITLE LADDER ENTRY ('AS' OR 'SEE'), CAST GROUP LADDER ENTRY, CAST GROUP TIME ENTRY, SYNTACTIC ROLE, OR SYNTACTIC ACTOR.

CAST LIST ENTRY CAST GROUP.

CAST LIST TIME ENTRY AN ITEM IN A CAST GRP PRECEDING ALL ROLES AND ACTORS IN THE GRP. IT IS DELIMITED ON THE RIGHT BY BY A COLON-BLANK. IT CONSISTS OF THE COLON-BLANK AND USUALLY SOME OTHER CHARACTERS FROM THE SET, '0123456789ABDTIVX', WHERE A, B, D, AND T ARE SMALL LETTERS. ALSO CALLED A CAST GROUP TIME ENTRY.

EXTRANEOUS TEXT A) COMMENT SECTIONS
B) TEXT IN OTHER SECTIONS THAT IS DELIMITED AS EXTRANEOUS TEXT.

MONOLOGUE WITH PARTS A TITLED SECTION TAGGED WITH THE SMALL LETTER 'U'.

OPERA SECTION A TITLED SECTION TAGGED BY THE SMALL LETTER 'O'. ITS DATA SHOULD COME FROM A SONG BOX IN THE CALENDER.

PERFORMANCE ALL OF THE SECTIONS STARTING WITH A PLAY SECTION (EITHER REAL OR SPECIAL) AND ENDING EITHER WITH END OF DATA OR START OF ANOTHER PLAY SECTION. ALTERNATIVELY ALL OF THE SECTIONS FOR ONE THEATRE ON ONE DATE.

SECTION ALL CHARACTERS BETWEEN ONE BLANK-ASTERISK-SMALL LETTER SEQUENCE AND THE NEXT SUCH SEQUENCE, EXCEPT THAT THE LAST SECTION IN A BLK ENDS AT THE END OF THE BLK. THE FIRST SECTION IN A BLK STARTS WITH THE FIRST BLANK-ASTERISK-SMALL LETTER SEQUENCE IN THE BLK. ALL CHARACTERS OF LOGICAL DATA PRECEDING THIS SEQUENCE ARE IGNORED, EXCEPT FOR ILLEGAL CHAR CHECKS. IN MTST INPUT THE '*' IS REPLACED BY THE PREFIX. IN SCANNER INPUT THE BLANK PRECEDING THE ASTERISK IS NOT ALWAYS REQUIRED, ALTHOUGH TO BE PROPER IT SHOULD ALWAYS BE THERE.

SECTION HEADER THE ITEMS FOLLOWING THE SECTION DELIMITER CHARACTER AND PRECEDING THE CAST LIST, IF ANY. FOR UNTITLED SECTIONS THIS IS THE SECTION TYPE LETTER. FOR TITLED SECTIONS THE HEADER STARTS WITH THE SECTION TYPE LETTER AND ENDS WITH THE FIRST OCCURRENCE OF ANY OF THE FOLLOWING ..
1) END OF THE SECTION.
2) PERIOD-BLANK-BLANK 3 CHAR SEQUENCE, '. '.
3) QUESTION MARK-BLANK-BLANK 3 CHARACTER SEQUENCE, '? '.
4) EXCLAMATION POINT-BLANK-BLANK 3 CHAR SEQUENCE, '! '.

SECTION TYPE LETTER (SECTYPE) THE CHARACTER FOLLOWING THE BLANK-ASTERISK OR BLANK-PREFIX DEFINING THE START OF A SECTION. THIS CHARACTER SHOULD BE A SMALL LETTER DESIGNATED TO REPRESENT A PARTICULAR TYPE OF SECTION.

PERFORMANCE DATE ENTRY IN A PLAY SECTION 1 TO 3 NUMBERS SEPARATED FROM EACH OTHER AND ALL OTHER ITEMS BY AT LEAST ONE BLANK. THE 1 TO 3 NUMBERS ARE CONSIDERED ONE ITEM, AND HENCE ANY PAGE ENTRY MUST PRECEDE ALL OF THEM. IF PRESENT THE DATE MUST LIE BETWEEN THE SECTION TYPE LETTER AND THE THEATRE, IF ANY (ABSENCE OF A THEATRE IN A PLAY SECTION IS AN ERROR).

THEATRE ENTRY A CHARACTER STRING IN THE HEADER PART OF A PERFORMANCE SECTION THAT FOLLOWS THE DATE ENTRY, IF ANY, SEPARATED FROM IT BY AT LEAST ONE BLANK, AND PRECEDES THE TITLE, IF ANY, SEPARATED FROM IT BY AT LEAST ONE BLANK. THE THEATRE BEGINS WITH THE FIRST NONBLANK CHARACTER, EXCEPTING PERIODS AND CERTAIN OTHER PUNCTUATION CHARACTERS, FOLLOWING THE DATE ENTRY, IF ANY. THE THEATRE ENDS WITH THE FIRST BLANK FOLLOWING ITS START, OR ELSE

ENDS WITH THE END OF THE SECTION, WHICHEVER COMES FIRST.
THE LEGAL CHARACTERS FOR A THEATRE ARE THE SMALL LETTERS
AND THE APOSTROPHE.

TIME ENTRY TITLE TIME ENTRY OR CAST LIST TIME ENTRY.

TITLE AREA FOR A PLAY SECTION THE AREA FOLLOWING THE THEATRE ENTRY AND ENDING WITH THE END OF THE SECTION HEADER. FOR ANY OTHER TITLED SECTION THE AREA BETWEEN THE SECTYPE LETTER AND END OF SECTION HEADER.

FULL TITLE STRING A CHARACTER STRING IN THE TITLE AREA. IT STARTS WITH THE FIRST CHARACTER IN THE AREA THAT IS NOT A PERIOD, BLANK, COMMA, SEMICOLON, QUESTION MARK, MINUS SIGN, OR EXCLAMATION POINT. IT ENDS WITH THE END OF THE TITLE AREA.

TITLE (SECTION TITLE) FOR PLAY AND AFTERPIECE SECTIONS THIS IS THE PART OF THE FULL TITLE STRING PRECEDING THE SUBTITLE IN THE STRING, IF ANY. FOR OTHER TITLED SECTIONS THIS IS THE PORTION OF THE FULL TITLE FOLLOWING THE TIME ENTRY, IF ANY, AND PRECEDING THE SUBTITLE, IF ANY.
HOWEVER ONLY 120 CHARS ARE PUT OUT FROM THE STRUCTUR PROGRAM, AND ONLY 83 ARE AVAILABLE IN SORTS.

TITLE TIME ENTEY A CHARACTER STRING IN THE FULL TITLE OF A TITLED SECTION. IF PRESENT IT STARTS WITH THE FULL TITLE STRING AND ENDS WITH THE FIRST 2 CHARACTER SEQUENCE OF COLON-BLANK, ': ', IN THE FULL TITLE STRING. THE STRING ITSELF MUST BE A LEGITIMATE CAST LIST TIME ENTRY IN OTHER RESPECTS.

TRICK SECTION A TITLED SECTION TAGGED BY THE SMALL LETTER 't'. ITS DATA SHOULD COME FROM TITLED ENTERTAINMENT BOXES.

TITLED SECTION A SECTION THAT IS SUPPOSED TO HAVE A TITLE.
AT PRESENT THESE SECTIONS ARE TAGGED BY ..
A, P, I, O, T, U. DISCOUNTING PAGE ENTRIES, ALL THESE SECTIONS SHOULD HAVE A TITLE OF AT LEAST THE SPECIAL VARIETY.

STRUCTURED TEXT TEXT THAT IS NOT EXTRANEOUS TEXT.

STRUCTURED SECTION ANY KIND OF SECTION EXCEPT A COMMENT SECTION.

TITLE LADDER ENTRY A LADDER ENTRY IN EITHER A TITLED OR UNTITLED SECTION THAT IS NOT PRECEDED BY ANY CAST GROUP ITEM. ,.
THERE ARE TWO TYPES .. 'SEE' & 'AS'.

SYNTACTIC ROLE IS ANY ITEM THAT IS ..

1) IN A CAST GROUP THAT HAS A DASH OR CAST GROUP LADDER ENTRY IN IT.
2) TO THE LEFT OF ALL DASHES AND THE CAST GROUP LADDER ENTRY, IF ANY, IN ITS GROUP.
3) IS NOT A TIME ENTRY.
4) IS NOT A PAGE ENTRY.
5) IS NOT A LADDER ENTRY.

SYNTACTIC ACTOR IS ANY ITEM IN A CAST LIST GROUP THAT ..

1) IS TO THE RIGHT OF THE 1ST DASH IN THE GROUP IF THE GROUP HAS A DASH IN IT, OR IS TO THE RIGHT OF THE CAST GROUP LADDER ENTRY, IF ANY, IN A GRP THAT HAS NO DASH.
3) IS NOT A TIME ENTRY.
4) IS NOT A LADDER ENTRY.
5) IS NOT A PAGE ENTRY.

THE RIGHT HAND DELIMITER OF A SYNTACTIC ACTOR MAY BE ..

1) A SEMICOLON.
2) END OF GROUP (EITHER SEMICOLON OR END OF SECTION).
3) COMMA.

*** STANDARD DATA FILES ***

THERE ARE 11 BASIC KINDS OF FILES ENCOMPASSING 7 GENERAL DATA FORMATS IN USE BY THE LSP INPUT SYSTEM. THE 11 TYPES OF FILE ARE NAMED AND BRIEFLY DESCRIBED BELOW. MORE DETAILED DESCRIPTIONS ARE GIVEN IN THE INDIVIDUAL PROGRAM WRITEUPS.

LSPIC ORIGINAL DATA FILE OF SCANNER OUTPUT. LSPIC FILES ARE INPUTS TO THE ICISCAN, ICIFRONT, AND ICIFIX PROGRAMS,

AND OUTPUT FROM THE ICIFIX PROGRAM.

LSPCR CORRECTION TEXT PORTION OF AN LSPIC FILE. FOR USE WITH WITH ICIFRONT. ALSO PRODUCED BY THE SLASH PROGRAM FROM CARD FORMAT INPUT.

LSPDT DATA TEXT PORTION OF AN LSPIC FILE. INPUT FOR ICISCAN OR ICIFRONT OR ICIFIX. OUTPUT FROM ICIFIX OR SCANNER (OR LSPCNLSL ETC.).

LSPMT ORIGINAL DATA FILE. LSPMT IS AN INPUT FOR THE SAVEDT PROGRAM, IF IT IS DESIRED TO CREATE A BACKUP OF THE INPUT THEN AN LSPMT FILE CAN ALSO BE OUTPUT FROM THE SAVEDT PROGRAM. SIMILARLY LSPMT FILES ARE BOTH INPUT AND OUTPUT FOR EDT.

LSPCL STANDS FOR 'CLEANED'. THIS IS OPTIONAL INPUT/OUTPUT OF THE ICISCAN PROGRAM. IT IS A LINE BY LINE FORM OF TEXT THAT HAS BEEN PUT THROUGH THE LINE READIN PROCESS AND CLEANED OF 'AT' SIGNS AND TYPIST ENTRIES ETC. EACH LINE IS HEADED BY THE FOLLOWING PIECES OF INFORMATION IN A FIXED FORMAT ..

1) TYPIST'S LETTER, 1 CHAR.
2) PAGE NUMBER, 4 CHAR.
3) LINE NUMBER WITHIN PAGE, 5 CHAR.
4) ABSOLUTE LINE NUMBER, 6 CHAR.
5) LENGTH OF LINE, 4 CHARS.

THUS A LSPCL FILE IS SUITABLE FOR USE AS A STREAM FILE ALLOWING EASY MANIPULATION IN A VARIETY OF WAYS BY A FORTRAN PROGRAM, EITHER BATCH OR RAX.

LSPPU STANDS FOR 'PURE'. THIS IS THE MAIN OUTPUT OF SAVEDT OR ICISCAN AND INPUT TO THE STRUCTUR PROGRAM. IT IS TEXT CLEANED OF COMMENT SECTIONS, AND EXTRANEOUS TEXT, AS WELL AS RUBOUT CHARS, ILLEGAL CHARS, AND CERTAIN CONTROL TEXT. EACH BLK SHOULD BE 1 OR MORE COMPLETE SECTIONS, AND EACH BLK SHOULD CONTAIN ONLY COMPLETE SECTIONS, NOT JUST PART OF A SECTION.

LSPST STANDS FOR STRUCTURED TEXT. IT IS OUTPUT FROM THE STRUCTUR PROGRAM, AND INPUT TO THE LADDER PROGRAM. THE DATA IN THIS FILE CONSISTS OF TAGGED GROUPS AND ITEMS, ONE SECTION PER BLK. THIS IS A REGIONAL(3) U FORMAT KEYED SEQUENTIAL FILE.

LSPSS STANDS FOR 'STRUCTURED SEQUENTIAL', LIKE LSPST, EXCEPT THAT IT HAS THE DIRECT ACCESS KEY OF THE LSPST FILE APPENDED TO THE FRONT OF THE BLK, AND IT IS PADDED IN FRONT OF THE KEY WITH 2 BLANKS TO MAKE A MINIMUM BLKSIZE OF 18 BYTES. THIS FORMAT IS INTENDED FOR STRUCTURED OUTPUT TO TAPE. THIS IS AN ORDINARY UNKEYED U FORMAT SEQUENTIAL FILE.

LSPLD STANDS FOR 'LADDER'. THIS IS OUTPUT FROM THE LADDER PROGRAM AND AND INPUT TO THE ITEMSGET PROGRAM. IT IS A KEYED DIRECT ACCESS FILE HOLDING SECTIONS WITH LADDER REFERENCES CONVERTED TO EXPLICIT CAST LISTS AND TITLES. LSPLD FILES ARE VERY SIMILAR TO LSPST FILES.

LSPLS STANDS FOR 'LADDER SEQUENTIAL'. JUST LIKE LSPLD EXCEPT THAT, AS IN THE LSPSS FILE, THE KEY IS APPENDED TO THE FRONT OF THE BLK AND PRECEDED BY 2 BLANKS TO MAKE THE 18 BYTE MINIMUM BLKSIZE FOR TAPE. THIS IS AN ORDINARY U FORMAT SEQUENTIAL FILE.

LSPIT STANDS FOR 'ITEMS'. IT IS OUTPUT FROM THE ITEMSGET PROGRAM AND INPUT TO THE SORT STEP. THE FORMAT IS FIXED UNBLOCKED KEYED REGIONAL(3) DIRECT ACCESS, AND EACH RECORD USUALLY CORRESPONDS TO A SYNTACTIC ROLE/ACTOR PAIR. HOWEVER THE RECORD MAY HOLD AS LITTLE AS THE SECTION TYPE, DATE, AND THEATRE.

LSPIS STANDS FOR 'ITEMS SEQUENTIAL'. SIMILAR TO LSPIT, BUT INTENDED FOR UNKEYED SEQUENTIAL TAPE OUTPUT FROM THE ITEMGET PROGRAM. THIS IS AN UNBLOCKED F FORMAT FILE.

LSPSR STANDS FOR 'SORTED'. OUTPUT FROM THE SORT PROGRAM. SAME FORMAT AS LSPIS, EXCEPT THAT MAY BE BLOCKED, IF BLOCKED, IS USUALLY BY A FACTOR OF 19. INPUT FOR THE FORMAT PROGRAM.

LSPIX SIU INDEX ENTRIES. OUTPUT FROM THE SAVEDT PGM. AT PRESENT ONLY CONTAINS THE INDEX ENTRY ITSELF. SHOULD ALSO CONTAIN

THE PAGE NUMBER, DATE, AND TAG. ICISCAN CAN EASILY BE
ALTERED TO PUT OUT LEGITIMATE SIU INDEX ENTRIES ENCOUNTERED
IN APPROPRIATE CONTEXTS.

1 CURRENT STATUS OF PROGRAMMING
****

PROGRAMMING TO BE DONE ..

1) INDIRECT SORT SYSTEM.
2) OPTIONAL CORRECTION OF ROLES OR ACTORS THAT START WITH 'END' ETC.
3) IMPLEMENT QUESTION MARK AS TITLE DELIMITER FOR TITLED SECTIONS.
4) ALTER LADDER TO PREVENT CGLE UPDATES FROM HAVING QUESTION
MARKS APPENDED TO THEM.
5) MINOR IMPROVEMENTS IN ICIFRONT.
6) PUT RECORD COUNT IN FRMAT, OPTIONAL PRINT OUTPUT,
AND OPTIONAL CHECK FOR IDENTICAL RECORDS.

STATUS OF PROGRAMS ..

CARDIN	DISCONTINUED
EDT	DONE
SAVEDT	DONE
ICIFIX	DONE
ICIFRONT	MAINT STAGE
ICISCAN	MAINT STAGE
STRUCTUR	MAINT STAGE
LADDER	MAINT STAGE
ITEMGET	MAINT STAGE
FORMAT	MAINT STAGE
LSPRINT	DONE
DOCPRINT	DONE
PLIST	DONE
TRT	DONE
MAP	DONE
COPYCAT	DONE
ICIORDER	NOT DONE
SLASH	MAINT STAGE

DOCUMENTATION ..

PROGRAMMING SPECS	MAINT STAGE
INPUT SPECS	MAINT STAGE
GENERAL	HALF DONE
BUG NOTES	NOT DONE
CHARACTER SET TABLES	DONE
OPERATION INSTRUCTIONS	MAINT STAGE
LISTINGS	DONE
RECOMMENDATIONS	NOT DONE

PROCEDURES ..

LSPCNLSL	DONE
LSPICFIX	DONE
LSPSLASH	NOT DONE
LSPFRONT	DONE
LSPSCLIN	DONE
LSPRINTS	DONE
LSPISCAN	DONE
LSPSTRCT	DONE
LSPLADDR	DONE
LSPITEMS	DONE
LSPFRMAT	DONE
LSPSORT4	DONE
LSPSORTT	DONE
LSPMERGE	DONE
LSPBACMP	DONE
LSPUTGEN	DONE
LSPUPGEN	DONE
LSPROCUP	DONE

/*

```
//OPERLIST JOB 99999211,WCD,MSGLEVEL=(1,1)
//D EXEC LSPRINTS,OPT=CARDOCSN,MEM=DOCOPERL
//PRINT.IN DD DDNAME=INSYS
//PRINT.INSYS DD *
                        THE GROOVE
      FOR STANDARDIZATION AND SIMPLICITY OF OPERATION, IT IS RECOMMENDED
   THAT THE FOLLOWING UNITS, VOLUMES, AND DSN'S BE USED ..

            INPUT                       OUTPUT

   PROCNAME    UNIT VOLUME DSNAME    UNIT VOLUME DSNAME

   IEHINITT    ANY  VARIES (NONE)
   LSPBAKUP NL 182  ------ ------ SL 180  VARIES VARIES
   LSPICFIX SL 182  VARIES VARIES SL 180  VARIES VARIES
   LSPFRONT SL 181  VARIES LSPDT
      ''    SL 182  VARIES LSPCR  SL 180  VARIES ISPCL
   LSPISCAN SL 180  VARIES VARIES SL 182  LSP010 LSPPU
   LSPSCLIN SL 180  VARIES LSPCL  SL 182  LSP010 LSPPU
   LSPSTRCT SL 182  LSP010 LSPPU  SL 180  LSP011 STRS
   LSPLADDR SL 180  LSP011 STRS   SL 182  LSP012 LSPLD
   LSPITEMS SL 182  LSP012 LSPLD  SL 180  LSP013 LSPIT
   LSPSORTT SL 180  LSP013 LSPIT  SL 182  LSP014 LSPSR
   LSPMERGE SL ANY  VARIES LSPSR
      ''    SL ANY  VARIES LSPSR  SL ANY  VARIES LSPSR
   LSPFRMAT SL 182  LSP014 LSPSR  PRINTER
   LSPRINTS SL 181  LSP002 SYSOUT PRINTER

      OPTION BITS FOR ICIFRONT .. (IOPT)
   1)  CLEAN INPUT FROM FILE DATX INSTEAD OF SCANNER FORMAT INPUT.
   2)  ENABLE LOG MESSAGES.
   3)  ENABLE 'ORG FOR C/D' LOG MESSAGES.
       EVERY COLUMN OF THE OPTION CARD SHOULD HAVE A '0' IN IT
       UNLESS IT IS INDICATING A '1' OPTION.

      OPTION BITS FOR ICISCAN ..
   1)  SIMULATE END OF DATA INPUT IMMEDIATELY (STOP).
   2)  PRINT ALL STRUCTURED OUTPUT SECTIONS ALONG WITH THEIR LINE NUMBERS.
   3)  LIST INPUT LINES AND CROSS REFERENCE INFO ON SYSPRINT.
   4)  SUPPRESS 'BAD TYPIST ENTRY' MESSAGE FOR ENTRIES ENDING WITH '$@'
       OF THE USUAL VARIETY.
   5)  LIST SECTIONS THAT ARE PROBABLY IN ERROR DUE TO THEIR ASSOCIATION
       WITH AN ILLEGAL DELIM OR THEIR NOT HAVING A SMALL LETTER AFTER THE '*'.
   6)  NOTE TYPIST PAGE ENTRIES AS THEY ARE ENCOUNTERED IN LINE READIN.
   7)  CLEANED INPUT FROM FILE CLIN.
   8)  PRINT PIECE OF TEXT PRECEDING ILLEGAL DELIM, FROM DELIM BEFORE ILLEGAL
       ONE TO THE ILLEGAL ONE INCLUSIVE, IMMEDIATELY FOLLOWING THE ILLEGAL
       DELIM MESSAGE ITSELF.
   9)  PRINT A LINE OF CERTAIN DEBUGGING INFO ON ENCOUNTERING A DELIM.
   11)  PERFORM ILLEGAL CHAR CHECK.
   12)  ENABLE ILLEGAL DELIM ERROR MESSAGE.
   13)  PRINT INDEX ENTRIES LEGITIMATELY PROCESSED.  CERTAIN INCORRECTLY
        DELIMITED ONES OR OUT OF CONTEXT ONES ARE NOT CAUGHT.
   14)  CLEANED OUTPUT TO FILE CLOUT.
   15)  SKIP INPUT LINES, BUT STILL COUNT THEM AS USUAL - IN PARTICULAR
        BLANK LINES ARE STILL NOT COUNTED.
   16)  REVERT TO REGULAR TEXT MODE IF IN CORRECTION TEXT MODE ONLY
        UPON ENCOUNTERING A PAGE ENTRY ON A LINE THAT HAS NO SLASHES
        (DESIRABLE FOR FIRST RUN WHEN CORRECTION TEXT IS LIKELY TO BE PRESENT).

      OPTION BITS FOR SAVEDT .. (IOPT)
   1)  PRINT TRANSLATED INPUT BLKS CLEANED OF EXTRA FEED CHARS.
   2)  PRINT ERR MSG CONCERNING MTST ENTRY STARTING TOO FAR AFTER
       BEGINNING OF BLK.
   4)  PRINT ERR MSG CONCERNING LACK OF '1/2' CHAR WITHIN 70 BYTES OF EOB.
   5)  PRINT ERR MSG CONCERNING ILLEGAL(0.9.6) CHARS IN TEXT.
```

6) PRINT MSG CONCERNING START OF EACH NEW SECTION.
7) PRINT MAP OF EACH INPUT BLK.
8) PRINT EACH PURIFIED TEXT BLK WRITTEN TO 'PURE' FILE.
11) PRINT EXPANDED ERR MSG CONCERNING NO MATCHING RIGHT BRACKET.
12) PRINT LOG-TYPE ERR MESSAGES ..
(FUNP,BNDX,ICSB,NORX,IXOB,NORB,MTCO).
14) ENABLE 'FUNP' ERROR MESSAGES.
15) PRINT ERR MSG CONCERNING NONCONSECUTIVE MTST TAPES.
17) LIST INDEX ENTRIES PRODUCED.

OPTION BITS FOR STRUCTUR .. (IOPT)
1) PRINT EACH RAW INPUT SECTION.
2) PRINT ONE LINE SECTION HEADER MESSAGE.
3) PRINT MAP OF INPUT BLK WITH COORDINATES, ETC FOR LOCATING ERRORS.
4) PRINT EACH SECTION OF RAW OUTPUT.
5) SCANNER TYPE INPUT, WITH '*' INSTEAD OF PREFIX FOR SECTION DLM.
6) PRINT MSG AFTER SUCCESSFULLY PROCESSING ICNDCN STAGE PAGE ENTRY.
7) CREATE UNKEYED SEQL OUTPUT, KEY DATA APPENDED TO FRONT OF BLKS.
8) PUT OUT DIRECT ACCESS KEYED BLKS.
9) ACCEPT MONTHS IN LADDER REFERENCES IN ANY COMBINATION OF UPPER OR LOWER CASE (USED IN CONJUNCTION WITH ALL UPPER CASE MONSTR CARDS).
21) LIST EACH ITEM RETURNED BY GITM.
25) SKIP INPUT BLK OPTION.
27) SET INDICATOR TO CAUSE PRINTING OF MAP AND RAW OUTPUT FOR SECTIONS ASSOCIATED WITH ERRORS.

OPTION BITS FOR LADDER ..
1) PRINT MAP WITH COORDINATES OF EACH INPUT BLK.
2) CLEAR LADA FILE AT START OF EACH NEW SEASON. USED IN ORDER TO PROCESS UNLIMITED SIZE FILES ON TAPE.
3) SKIP INPUT BLKS OPTION (ALSO IGNORES SEASON CHANGES). FOR DEBUGGING OR SKIPPING AROUND SECTIONS THAT CAUSE PROGRAM TO BOMB ETC.
4) PRINT EACH BLK OF RAW INPUT READ DURING LADDER SEARCH.
5) ONE LINE HEADER FOR EACH INPUT BLK, GIVING LENGTH, SECTYPE, THEATRE, BLK NUMBER, DATE, AND 1ST PART OF BLK ITSELF.
6) SEQUENTIAL FORMAT INPUT FROM FILE STRS.
7) SEQUENTIAL FORMAT OUTPUT TO FILE LADS.
17) PRINT OLD AND NEW TITLES AFTER LADDER REFERENCE READ-IN.
24) PRINT EACH BLK OF RAW OUTPUT.
27) SET INDICATOR FOR PRINTING OF EXTRA INFO FOR SECTIONS WITH ERROR MESSAGES.
LADDR HAS A DEBUG CHAIN IN COLS 33-36.

OPTION BITS FOR ITEMGET .. (IOPT)
1) PRINT MAP OF EACH INPUT SECTION IN LINES OF 100 CHARS EACH WITH COORDINATES FOR EASY LOCATION OF CHARACTERS IN ERROR.
2) PRINT EACH ITEM AS IT IS ENCOUNTERED IN INPUT/SETUP. (DEBUG).
3) PRINT EACH GRP AS IT IS ENCOUNTERED IN INPUT SETUP. (DEBUG)
4) SECTION TITLES IN CAPS.
5) PRINT HEADER FOR EACH INPUT SECTION, INCLUDES INPUT SECTION NUMBER, KEY UNTIL IT'S TAKEN OUT, LENGTH OF SECTION, AND FIRST FEW CHARS OF SECTION ITSELF.
7) ALLOW FOR PERFORMANCE SELECTION VIA LEVEL 4 INCLUSIVENESS.
8) UNKEYED SEQUENTIAL INPUT FROM LADS FILE INTENDED FOR TAPE INPUT).
9) UNKEYED SEQUENTIAL OUTPUT TO ITSM FILE (INTENDED FOR TAPE OUTPUT).
10) PRINT CONTROL FLOW TRACE FOR SELECTED TRANSFERS.
11) PRINT THE SELECTION CONTROL BLK BEFORE EACH EXIT FROM THE CONTROL CARD INTERPRETER IN NEAT FORMAT WITH HEADINGS.
12) CLEAR STATS ARRAY AFTER EACH PRINTING OF IT.
19) PRINT EACH OUTPUT RECORD IN FORMAT WHICH USUALLY FITS ON 1 LINE.
29-32) ALLOW ERROR MESSAGE FOR ILLEGAL CHARACTER TO BE PRINTED THEATRE, TITLE, ROLE, AND ACTOR RESPECTIVELY.

OPTION BITS FOR FORMAT (IOPT) ..
1) LIST IN DATA DIRECTED FORMAT ALL VARIABLES IN PROGRAM

AFTER EACH RECORD IS PRINTED (DEBUG).

OPTION BITS FOR LSPRINT .. (IOPT)

64) GET DATA TO BE LISTED FROM FILE SYSIN RATHER THAN FILE IN.
65) NO PRINTER CARRIAGE CONTROL CHARACTER.
66) TYPE NUMBER OF LINES COUNTED ON 1052 AFTER EACH ENDFILE OR END OF RANGE.
69) PUT LINE NUMBERS IN F(7) FORMAT STARTING IN CCL(83) OF OUTPUT.
71) DO NOT COUNT ENTIRELY BLANK LINES (CARRIAGE CTRI CHAR IS CONSIDERED HERE).
72) CREATE BACKUP FILE OF INPUT.
73) NO TRANSLATION FOR FIRST PRINTING OF LINE.
74) NO OVERPRINT ON TOP OF FIRST LINE.
75) ENABLE STANDARD PL/1 SYSTEM ACTION FOR ENDPAGE(SYSPRINT) ON-CONDITION, I.E. DON'T IGNORE ENDPAGES.

LSPRINT'S OPTION CARD FOLLOWS THE TRANSLATION TABLES. THERE IS NO OPTION CHAIN, AND LSPRINT CHECKS FOR NO ERRORS.

PLIST OPTIONS

PLIST HAS ONLY 1 OPTION .. LIST LENGTH OF INPUT RECORDS. THIS LENGTH LISTING IS SPECIFIED BY PUTTING ONE OR MORE CHARACTERS IN THE PARM FIELD OF THE EXEC STATEMENT. THERE SHOULD BE NO PARM FIELD IF A PLAIN LISTING IS DESIRED.

ICIFRONT ..

1) NO '*' AT LOGICAL BEGINNING OF CORRECTION STATEMENT.
WHEN LOOKING FOR THE '*' THAT DEFINES THE START OF EACH CORRECTION STATEMENT, THE PROGRAM FOUND ANOTHER NONBLANK CHARACTER FIRST. THIS IS A FLUSHING ERROR.

2) ILLEGAL END OF PAGE ENTRY IN LINE N
WHERE N IS THE CURRENT ABSOLUTE LINE NUMBER OF THE DATA TEXT. THIS IS A FAIRLY UNLIKELY ERROR INDICATING THAT SOMETHING IS HORRIBLY WRONG EITHER WITH THE CORRECTION TEXT OR WITH THE PROGRAM. THE PROGRAM STOPS AFTER THIS MESSAGE.

3) NO FCN
AFTER THE BEGINNING OF A STATEMENT NO FUNCTION OR COMMAND CHARACTER CANDIDATES COULD BE FOUND AT ALL. A FATAL ERROR.

4) BAD PAGE SEQUENCE M,N IN LINE X.
PAGE N FOLLOWS PAGE M IN THE DATA TEXT. PAGE N STARTS ON ABSOLUTE LINE X.

5) BAD TYPIST ENTRY IN LINE N.
IN ABSOLUTE LINE N THERE IS AN APPARENT TYPIST ENTRY (AN ASTERISK-CAPITAL LETTER IN COLS 1-2), BUT EITHER IT DIDN'T HAVE ANY DIGITS OR ELSE ITS DIGIT PORTION ENDED WITH A NONBLANK CHARACTER THAT WAS NOT PART OF A '$@' SEQUENCE.

6) ERRONEOUS END OF CORRECTION.
THIS MESSAGE OCCURS IN THE EVENT OF AN UNSUCCESSFUL FLUSHING ERROR. THE PROGRAM STOPS AFTER THIS MESSAGE.

7) ILLEGAL FCN CHAR
AN APPARENT FUNCTION OR COMMAND CHARACTER THAT IS NOT A C, D, OR I, EITHER SMALL OR CAPITAL. THIS IS A FLUSHING ERROR.

8) NONBLANK FOLLOWS FCN CHAR
THE CHARACTER FOLLOWING A LEGAL APPARENT FUNCTION CHARACTER IS NOT A BLANK. THIS IS A FLUSHING ERROR.

9) NO LINE NUMBER
IN AN APPARENT CORRECTION STATEMENT STATEMENT WITH A LEGAL FUNCTION CHAR FOLLOWED BY A BLANK THERE IS NOT A LEGAL LINE NUMBER SPECIFICATION. THIS IS A FLUSHING ERROR.

10) MULTILINE I CMND
A CORRECTION STATEMENT WITH A FUNCTION CODE OF 'I' AND MULTILINE LINE SPECS. THIS IS A FLUSHING ERROR.

11) NEGATIVE 1ST WORD SPEC.
IN A PHRASE COMMAND THE LOGICAL LOCATION OF THE FIRST WORD SPECIFIED IS BEFORE THE FIRST WORD OF THE LOGICAL LINE. THE LOGICAL LINE STARTS AFTER THE TYPIST ENTRY, IF ANY. THIS ERROR USUALLY OCCURS ON A LINE WITH A TYPIST ENTRY WHEN THE CORRECTION TRIES TO CORRECT THE TYPIST ENTRY.

12) NEGATIVE 2ND WORD SPEC
SIMILAR TO THE PRECEDING ERROR, BUT EVEN THE LAST WORD OF THE PHRASE PRECEDES THE FIRST LOGICAL WORD OF THE LINE.

13) NO '/ ' (SLASH-BLANK)
EITHER THERE WAS NO '/ ' TO DELIMIT THE END OF NEW CHANGE OR INSERTION TEXT IN A CORRECTION TSTATEMENT, OR ELSE THE '/ ' OCCURRED IMMEDIATELY SO THAT THERE WAS TEXT OF ZERO LENGTH. THIS IS A FLUSHING ERROR.

ICISCAN ERROR MESSAGES

'****** ILLEGAL CHAR 'X' IN LINE J, Y' X IS THE ILLEGAL CHAR, J IS THE LINE NUMBER, AND Y IS THE DECIMAL VALUE OF THE CHARACTER IN CASE THE CHARACTER DOESN'T PRINT (AS IS USUALLY THE CASE WITH ILLEGAL CHARACTERS). THE ILLEGAL CHARACTER IS CONVERTED TO A SLASH SO THAT IT MAY BE NOTICED MORE EASILY IN LATER STAGES OF THE INPUT SYSTEM.

'$$$$$$$$$ APPARENT TYPIST ENTRY IS X LINES FROM PREVIOUS LINE J, PREVIOUS PAGE WAS N' SELF EXPLANATORY. OCCURS WHENEVER A NEW PAGE DOES NOT START 30 LINES FROM THE PREVIOUS ONE.

'$$$$$$ BAD PAGE SEQUENCE XY' X IS THE PREVIOUS PAGE AND Y IS THE CURRENT PAGE. THIS OCCURS WHENEVER THE NUMBER OF A PAGE DOES NOT FOLLOW THAT OF THE PREVIOUS PAGE.

'$$$$$ BAD TYPIST ENTRY, LINE J LLLLL' WHERE 'LLLLL' IS AN IMAGE OF THE LINE AS IT CURRENTLY EXISTS IN MEMORY AND J IS THE LINE NUMBER. UNLESS OPTION 4 IS SPECIFIED THIS MESSAGE IS GENERATED WHENEVER THERE IS AN ASTERISK-CAPITAL LETTER PAIR IN THE FIRST 2 PLACES OF A LINE AND THESE ARE NOT FOLLOWED BY ONE OR MORE DECIMAL DIGITS ENDING WITH A BLANK. IF OPTION 4 IS SPECIFIED THEN THE MESSAGE IS NOT GENERATED IN THE SPECIFIC INSTANCE THAT THE DECIMAL DIGITS END WITH A '$@' (DOLLAR SIGN-AT SIGN).
NOTE THAT ALL 3 TYPIST ENTRY ERROR MESSAGES START WITH DOLLAR SIGNS, AND THAT ONLY TYPIST ENTRY ERROR MESSAGES START WITH DOLLAR SIGNS.

'********* BAD TRIPLE 'AT' IN REC J, LLLLL' WHERE J IS THE LINE OR RECORD NUMBER, AND 'LLLLL' IS THE CURRENT IMAGE OF THE LINE IN MEMORY. THIS MESSAGE OCCURS WHEN A LINE HAS A TRIPLE 'AT' FOLLOWED BY ANY NONBLANK CHARACTERS SUCH AS A FOURTH 'AT' ON THE SAME LINE. THE LINE IS ENTIRELY LEFT OUT, HOWEVER PAGE ENTRY PROCESSING HAS ALREADY BEEN DONE.

'********* ILLEGAL SECTION '*X' IN LINE J WHERE X IS THE CHAR FOLLOWING AN ASTERISK IN THE STRUCTURED TEXT THAT IS NOT A SMALL LETTER.

THE SECTION IS PUT INTO THE OUTPUT FILE ANYWAY, BUT IF THE CHAR WAS A CAPITAL LETTER THEN IS IS PUT OUT AS A SMALL LETTER.

' E=*XLLLLL' WHERE 'X' IS THE SECTION TYPE CHARACTER AND 'LLLLL' IS THE BODY OF THE SECTION. THIS MESSAGE, SIMILAR TO THE ' M= *XLLLLL' MESSAGE, IS USED IN PLACE OF ' M= *XILILL' WHEN THE SECTION IS ASSOCIATED WITH AN ILLEGAL DELIM OR ILLEGAL SECTYPE ERROR. IF X IS A SMALL LETTER FOR AN ILLEGAL SECTYPE ERROR THEN THE ORIGINAL CHARACTER WAS CAPITAL X.

'///// ILLEGAL DELIM 'X' IN LINE J, MODE= N' WHERE X IS ONE OF THE 9 BASIC DELIMITERS USED BY ICISCAN, AND N IS THE MODE OF THE PIECE OF TEXT PRECEDING THE ILLEGAL DELIMITER. THE MODE OF A PEICE DETERMINES WHAT THAT TEXT IS CONSIDERED TO BE. FOR EXAMPLE IN MODE 0 THE TEXT IS CONSIDERED PART OF SOME STRUCTURED OUTPUT SECTION, HENCE MODE 0 IS CALLED 'STRUCTURED TEXT MODE. MODE -3 IS (INTENDED) FOR INDEX ENTRIES FOLLOWING 2 LEFT BRACKTES. THE MODE -3 STARTS WITH A PLUS SIGN OR DOLLAR SIGN, AND ENDS WITH AN EQUAL SIGN (PREFERABLY), HOWEVER IT MAY ALSO END WITH A RIGHT BRACKET, PERCENT SIGN, OR ASTERISK. NATURALLY, ENDING WITH ANYTHING OTHER THAN AN EQUAL SIGN CAUSES AN ILLEGAL DELIM ERROR.

'*** PAGE NUMBER TOO LONG IN OR BEFORE LINE J' THIS MESSAGE IS FOR A LONDON STAGE PAGE NUMBER OF THE DECIMAL VARIETY WITH MORE THAN 4 DIGITS. THE NEW PAGE NUMBER IS NOT ACCEPTED. NO ERROR MESSAGE IS GIVEN FOR APPARENT PAGE ENTRIES NOT ENDING WITH BLANKS - THEY ARE IGNORED.

'********** BAD PAGE IN OR BEFORE LINE J' AN APPARENT ROMAN PAGE ENTRY HAD A CHARACTER IN IT THAT WAS NOT A CAPITAL I, V, OR X. NO ERROR MESSAGE IS GEVEN FOR APPARENT ROMAN PAGE ENTRIES WITH BLANKS. THEY ARE IGNORED.

- SAVEDT LOG-TYPE ERROR MESSAGES ..

FUNP 'FUNNY PAGE NUMBERING' PAGE ENTRIES, OTHER THAN THE FIRST ONE IN A RUN ARE NOT IN CONSECUTIVE ORDER. THIS CHECKING IS UNRELATED TO MTST TAPE ORDER OR DATE SEQUENCE CHECKING.

MTOC NONCONSECUTIVE MTST ENTRIES.

NORB NO RIGHT BRACKET.

IXOB INDEX OUTSIDE OF BRACKETS.

NORX NO RIGHT HAND DELIMITER OF AN INDEX ENTRY THAT HAS A IEFT

ICSE INDEX CAUGHT CROSSING SECTION OR BRACKET

BNDX LAST CHARACTER OF AN INDEX ENTRY IS NOT A SMALL LETTER OR QUOTE.

STRUCTUR ERROR NOTE/LOG TYPE MESSAGE CODES ..

BDYR IN A LADDER REFERENCE THE APPARENT YEAR WAS EARLIER THAN 1660 OR LATER THAN 1800. THE LADDER REFERENCE WAS TREATED AS A CAST ITEM.

NLI1 LEGATIVE LENGTH ITEM. VERY RARE. FAIRLY HARMLESS TO STRUCTUR, BUT DON'T KNOW IF IT WILL HARM IADDER. MAY OCCUR WHEN THE DELIMITER IS THE FIRST CHAR OF A POTENTIAL ITEM (IF THAT FIGURES) IN A GITM CALL AND THE DELIMITER IS FOLLOWED ONLY BY BLANKS OR PERIODS OR OTHER DELIMITERS WITHIN THE RANGE. IF IN ANY OTHER CASE YOU MAY WANT TO CALL WILL, ESPECIALLY IF LADDER BOMBS ON THE DATA. NLI1 IS FOLLOWED INVARIABLY BY NLI2.

NLI2 RIGHT HAND DATA FOR NLI1 ERROR. NLI1 GIVES LEFT HAND DATA AS WITH NIT1/NIT2 PAIR OF ERROR MESSAGES.

NSEC BLK HAS NO SECTION DELIMITER. THE BLK IS PROCESSED ANYWAY. IF THIS ERROR OCCURS WITH ICISCAN OUTPUT THEN ICISCAN SHOULD BE AT FAULT, IF WITH SAVEDT OUTPUT THEN SAVEDT'S INPUT DATA IS MORE LIKELY TO BE THE CAUSE.

NIT1 FIRST ERROR MESSAGE FOR MISSING ITEM WHERE ONE IS REQUIRED. THE TEXT SHOULD BE THE LEFT HAND AREA ASSOCIATED WITH THE MISSING ITEM. NIT2 ALWAYS FOLLOWS NIT1.

NIT2 SECOND ERROR MESSAGE FOR MISSING ITEM. ALWAYS FOLLOWS NIT1.

GIVES THE RIGHT HAND AREA ASSOCIATED WITH THE ITEM. THE NUMBERS ARE SUPPOSED TO BE THE LEFT AND RIGHT HAND PTRS TO THE ITEM RESPECTIVELY.

ISEC ILLEGAL SECTION TYPE. A BLANK-PREFIX WAS ENCOUNTERED THAT WAS NOT FOLLOWED BY ONE OF THE SMALL LETTERS THAT DENOTE A LEGAL TYPE OF SECTION.

NTTL A PERFORMANCE OR AFTERPEICE SECTION HAS NO '. ' THREE CHAR LENGTH STRING IN IT (THE 1ST OF WHICH DELIMITS THE PLAY TITLE IN THE SECTION).

XDAT 4 DECIMAL NUMBERS WERE FOUND IN THE PRE-THEATRE PART OF A PLAY SECTION. THE 1ST THREE WERE INTERPRETED AS YEAR, MONTH, & DATE RESPECTIVELY. THE FOURTH WAS IGNORED.

DPRT ONE OR MORE PARTS OF THE MOST RECENT NEW DATE IS OUTSIDE ITS LEGAL RANGE. THE OLD VALUE OF THE OUT OF RANGE PART IS NOT CHANGED.

DCHG THE DIFFERENCE BETWEEN THE LAST PREVIOUS DATE AND THE NEWEST ONE IS GREATER THAN ABOUT 100 DAYS, OR ELSE IT'S EARLIER THAN THE OLD DATE. THE NEW DATE IS ACCEPTED ANYWAY. TOO BAD, IT'S PROBABLY WRONG, BUT MAYBE IT WILL BE SET RIGHT AGAIN SOON.

UNVT IN A SYNTACTIC ROLE-ACTOR GROUP AN APPARENT TIME ENTRY CONTAINED A CHARACTER THAT IS NOT ALLOWED IN TIME ENTRIES. THE APPARENT TIME ENTRY WAS TAKEN AS A TIME ENTRY ANYWAY, COMPLETE WITH THE ILLEGAL CHARACTER.

DRNG IN AN APPARENT LADDER REFERENCE THE DAY OF THE MONTH WAS GREATER THAN 31. THE APPARENT LADDER REFERENCE WAS TREATED AS A SYNTACTIC ROLE OR ACTOR.

MNTH IN AN APPARENT LADDER REFERENCE THE MONTH OF THE YEAR COULD NOT BE DETERMINED. THE MONTH MUST BE STATED BY ONE OF THE FOLLOWING CODES .. JAN,FEB,MARCH,APRIL,MAY,JUNE,JULY,AUG,SEPT,OCT,NOV,DEC. EACH MONTH MUST START WITH A CAPITAL , WITH THE OTHER LETTERS BEING SMALL. A PERIOD MAY FOLLOW ANY OF THESE CODES.

NOMN STILL TO BE EXPLAINED.

SUSP A PAGE NUMBER ENTRY WAS FOUND IN WHICH THE NEW PAGE NUMBER WAS MORE THAN ONE GREATER THAN THE PREVIOUS NUMBER OR ELSE WAS SMALLER THAN THE PREVIOUS PAGE NUMBER. THE NEW PAGE NUMBER IS ACCEPTED ANYWAY.

LADDER ..

IXGO SOMETHING IS WRONG. PROGRAM ERROR. VARIABLE IXG HAS BECOME 0 DURING FINAL OUTPUT FORMATTING WHILE INTERPRETING THE OUTPUT CHAIN. CALL WILL. (NEAR LABEL ENDIN, STMT 521).

OGRP ON INPUT A GROUP WITH A LENGTH OF 0 WAS ENCOUNTERED . IT WAS IGNORED AND PROCESSING WAS CONTINUED WHTH THE NEXT GROUP, IF ANY. (AFTER LABEL ISTGRP, NEAR STMT 130).

NEFA SOMETHING WRONG. A CHAR WHICH IS SUPPOSED TO BE HEX 'FA' IS NOT HEX 'FA'. EITHER LADDER OR STRUCTUR IS TO BLAME.

RKEY NO MATCH FOR ROLE IN KEYING ATTEMPT IN A PAIRED ROLE-ACTOR GROUP THAT HAS A SIGNED ACTOR. THE GROUP IS IGNORED. PROCESSING CONTINUES WITH THE NEXT GRP. THE NUMBER POINTS TO THE ROLE.

AKEY NO MATCH IN KEY SEARCH KEYING ON AN ACTOR IN A PAIRED GRP IN WHICH THE 1ST ROLE IS SIGNED. THE GRP IS IGNORED. PROCESSING CONTINUES WITH THE NEXT GRP. THE NUMBER POINTS TO THE 1ST ROLE IN THE GROUP.

NGCB NO MORE GROUP CONTROL BLOCKS LEFT. REQUIRES RECOMPILATION WITH NUMBER OF GCB'S INCREASED. THE NUMBER PTS TO THE START OF THE 1ST GRP LEFT OUT. BE SURE TO CHECK THE NUMBER OF ICB'S TOO. SUCCEEDING GRPS ARE NOT PROCESSED, AND THUS NOT WRITTEN OUT.

NICB SAME AS NGCB, BUT FOR ITEM CONTROL BLOCKS.

XLDR EXTRA TITLE LADDER ENTRY ENCOUNTERED IN SECTION. ONLY ONE LADDER LADDER ENTRY IS ALLOWED IN A TITLED SECTION. THE SECOND AND LATER ENTRIES ARE IGNORED.

ISEC ILLEGAL SECTION LETTER.

LERA LADDER GRP REFERRED-TO HAS NOTHING TO DELIMIT END OF THE TITLE, IF ANY, THEREFORE THE TITLE, IF ANY, WAS THE ONLY DATA THAT MIGHT HAVE BEEN GAINED BY THE LADDER REF, USUALLY LADDER GRPS ARE FOR MORE THAT JUST A TITLE.

NCGM 'CAST GRP NO MATCH'. A REFERRED-TO SECTION FOR A CAST GRP LADDER REF WAS FOUND, BUT THE SECTION HAD NO MATCH FOR THE FIRST ROLE

ROLE IN THE REFERRING GRP.

RNMP ROLE NO MATCH PROGRAM ERROR. SHOULD NEVER OCCUR. CALL WILL.

OGCE A COMPLETELY EMPTY GRP HAS OCCURRED OR BEEN FOUND SOMEWHERE IN THE MIDST OF LADDER PROCESSING. IT MAY BE AN ERROR, BUT IN ANY CASE SHOULD CAUSE LITTLE TROUBLE. CALL WILL.

INKY INCOMPLETE KEYING ON A CGLE. ONE OR MORE ROLES DO NOT MATCH WHILE AT LEAST THE FIRST ROLE IN THE REFERRING GRP DOES HAVE A MATCH.

GDLT INABILITY TO KEY ON ANYTHING IN A GRP DELETION ATTEMPT.

URDT INABILITY TO KEY ON ROLE TO BE DELETED IN UNPAIRED ROLE DELETION ATTEMPT.

UADT SAME AS URDT, BUT FOR ACTOR.

KRNF KEYED RECORD NOT FOUND. A LADDER SEARCH FAILED TO FIND THE SECTION SOUGHT BECAUSE IT WAS NOT ON THE DISK. IF THIS MESSAGE IS IMMEDIATELY PRECEDED BY ONE OR MORE 'CRTITI=/CIDTITL=' MESSAGES THEN IT MAY BE JUST THE BAD MATCHES INDICATED IN THOSE MESSAGES, THAT IS, IT FOUND THE RECORD, BUT GOT A BAD COMPARE ON THE 1ST TRY, SO IT TRIED AGAIN ON THE NEXT TRK, BUT THEN THERE WAS NO RECORD AT ALL WITH THE RIGHT KEY. THE NUMBER TELLS ON WHICH TRY THE FAILURE OCCURRED. 1=1ST TRY AND INDICATES NOT EVEN A BAD COMPARE.

IDLT IN AN ITEM DELETION IN A PAIRED GRP THE ITEM TO BE DELETED COULD NOT BE FOUND IN THE REFERRED-TO GRP. THE ORIGINAL ITEM IS LEFT INTACT BUT WITHOUT THE MINUS SIGN (THAT IS, IT IS NOT DELETED EITHER), AND NO FURTHER PROCESSING IS ONE ON THE GRP.

NTRK THE DATE/TRACK MAP INDICATES THAT NO SECTION HAS BEEN WRITTEN IN THE LADA FILE WITH THE DATE FOUND IN A LADDER REFERENCE KEY. THE LADDER REFERENCE IS IGNORED, AND PROCESSING CONTINUES WITH THE NEXT ITEM.

FREF FORWARD REFERENCE IN A LADDER ENTRY. THE DATE INVOLVED IS ON OR LATER THAN THE DATE OF THE SECTION IN WHICH THE LADDER ENTRY OCCURS. THE ENTRY IS IGNORED. PROCESSING CONTINUES WITH THE NEXT ITEM.

TTLC NO TITLE MATCHING THE ONE IN A BOX CONTAINING A LADDER ENTRY WAS FOUND IN ANY OF THE LADDER BLOCKS SEARCHED AS A RESULT OF THE LADDER ENTRY. THE NUMBER POINTS TO THE LEFT-HAND CHARACTER OF THE LADDER ENTRY.

2TIM MORE THAN ONE TIME ENTRY ENCOUNTERED IN A SINGLE GROUP. THE SECOND ENTRY IS IGNORED. PROCESSING CONTINUES WITH THE NEXT ITEM.

OTSZ SIZE OF AN OUTPUT BLK IS GREATER THAN 3588 BYTES, SO IT CAN'T FIT ON A TRACK. REPROGRAMMING WILL BE NECESSARY UNLESS THE THE BLOCK'S SIZE CAN BE REDUCED TO 3588 OR LESS (OR SOMETHING). THE BLK WAS NOT WRITTEN AT ALL. THE BLK WAS PROCESSED COMPLETELY IN OTHER RESPECTS. PROCESSING CONTINUES NORMALLY. THE NUMBER IS THE BLK'S SIZE.

ITEMSGET ..

ITHE ILLEGAL CHARACTER IN THEATRE, USUALLY A CAPITAL

ITTI ILLEGAL CHARACTER IN TITLE

IROL ILLEGAL CHARACTER IN ROLE, OFTEN AN '&C'

IACT ILLEGAL CHARACTER IN ACTOR

OGRP 'GRP' LENGTH IS LESS THAN 2, INCLUDING CONTROL CHARS. PROGRAMMING ERROR SOMEWHERE, EITHER IN ITEMSGET OR BEFORE.

NEFA ANOTHER PROGRAMMING ERROR, DEFINITELY IN ITEMSGET.

2TTL TWO TITLES IN THE SAME BLK, OR AT LEAST 2 ITEMS WHICH ARE NOT ROLE, ACTORS, OR TIME ENTRIES, WHICH ARE SUPPOSED TO BE ALL THAT EXIST BESIDES TITLES AT THIS STAGE.

2TIM 2 TIME ENTRIES ENCOUNTERED IN ONE GROUP, SECOND ONE IS IGNORED.

FSEQ PROGRAM ERROR. ITDM OUTPUT HAS NOT CAUGHT UP TO ITEM OUTPUT ON THE DA OUTPUT FILE AT THE END OF THE PROGRAM.APPARENT FAILURE ASSOCIATED WITH 2ND PASS FOR INCLUSIVENESS LEVELS 2, 3, OR 4. ONLY QUALITY OF OUTPUT SHOULD BE AFFECTED, BUT CALL WILL DUE TO PROGRAM ERROR. RERUNNING WITH INCLUSIVENESS LEVELS 2, 3, AND 4 LEFT OUT MAY HELP.

FORMAT ..

```
        NO ERRORS DETECTED BY FRMAT AT PRESENT.

           FORMAT OF SORT RECORDS ..
        BYTE    1        SECTION TYPE CHARACTER.                                  1  BYTE
        BYTES   2-9      DATE IN FORMAT 'YYYYMMDD'.                               8  BYTE
        BYTES   10-17    THEATRE, PADDED ON RIGHT WITH BLANKS, IN LOWER CASE.  8 BYTE
        BYTES   18-100   TITLE PADDED ON RIGHT WITH BLANKS.                      83 BYTE
        BYTES   101-140  ROLE, INCLUDING 2 CHARS WHICH INDICATE
                         QUESTIONABLE STATUS (VIA 'SEE' TYPE LADDER REFERENCE) 40 BYTE
        BYTES   141-180  ACTOR, IN SAME FORMAT AS ROLE.                          40 BYTE
        BYTES   181-188  TIME ENTRY                                               8  BYTE
           TOTAL LENGTH IS 188 BYTES.  PREFERABLY BLOCKED BY FACTOR OF 19.
        /*
        //OLSPIS   JOB 99999211,WCD,MSGLEVEL=(1,1)
        //D EXEC LSPRINTS,OPT=CARDOCSN,MEM=DOCOLSPS
        //PRINT.IN DD DDNAME=INSYS
        //PRINT.INSYS DD *
                     OPERATING THE LONDON STAGE PROJECT INPUT SYSTEM
                     ***********************************************

                                  SUMMARY OF STEPS
                                  ****************

                                *** INITIAL PROCESSING ***
           LSPBAKUP  TO CREATE SL BACKUP OF NL TAPE FROM SCANNER COMPANY.
           ICIFIX    TO PRODUCE A FILE OF DATA TEXT FROM THE SL BACKUP,
                     DELETE BLANK LINES, AND LOCATE BAD TYPIST ENTRIES.
           ICIFIX    TO PRODUCE A FILE OF CORRECTION TEXT FROM THE SL,
                     BACKUP, DELETING BLANK LINES.
           LSPRINT   TO LIST THE FILE OF CORRECTION TEXT.
           ICISCAN   TO PRINT ONTO TAPE A LIST/LOG OF THE DATA TEXT AND TO
                     PRODUCE PURE FILE OUTPUT FOR THE STRUCTUR PROGRAM.
           LSPRINT   TO LIST ICISCAN'S LIST/LOG PRINTOUT WITH OVERPRINTING
                     ON THE SN CHAIN FOR ERROR ANALYSIS AND CROSS REFERENCE.
           STRCT     FOR LOCATING DCHG ERRORS AND SEASON CHANGES THAT
                     DON'T WORK RIGHT.
           LADDR     TO LOCATE ANY SINGLE ERRORS THAT CAUSE MANY ERROR
                     MESSAGES, BUT ARE EASY TO FIX.   THIS STEP SHOULD
                     NOT BE RUN UNLESS THE STRCT STEP HAS NO BAD DCHG ERRORS,
                     AND NO SEASON CHANGES MISSED.

                                  *** FIRST CYCLE ***
           ANALYSIS  (DONE BY THE OPERATOR)  TO LOCATE IMPORTANT ERRORS,
                     AND DETERMINE REMEDIAL ACTION FOR THEM.
                        THE FOLLOWING SPECIAL AREAS FOR THE OPERATOR TO
                     CHECK ARE LISTED IN ORDER OF IMPORTANCE ..
                     1)  THE FIRST LINE.
                     2)  THE ENDING OF EACH SEASON EXCEPT THE LAST IN THE RUN.
                     3)  THE START OF EACH MONTH OR YEAR.
                     4)  TYPIST ENTRIES.
                     5)  SEASONS NOT STARTING WITH A PLAY SECTION.  IN THIS
                     EVENT THE ITEMGET PROGRAM MAY BOMB OUT, BUT LADDR
                     SHOULD WORK PRETTY WELL.
           ICIFIX    TO FIX SERIOUS ERRORS, AND POSSIBLY OTHER ERRORS,
                     DISCOVERED BY ICIFIX, ICISCAN, STRCT, AND POSSIBLY LADDR.
           ICIFRONT  (AN OPTIONAL STEP) TO APPLY CORRECTIONS TO THE DATA TEXT.
           ICISCAN   (EITHER USING LSPSCLIN FOR CLEANED FORMAT INPUT OR
                     LSPISCAN FOR SCANNER FORMAT DATA TEXT) TO CHECK FOR
                     SERIOUS ERRORS, ESPICIALLY FROM APPLICATION OF CORRECTIONS.
                     OPTION 6 SHOULD BE USED INSTEAD OF OPTION 3.
           STRCT     SAME AS FOR INITIAL PROCESSING STAGE.
           LADDR     SAME AS FOR INITIAL PROCESSING STAGE.

                               *** SECOND (& LATER) CYCLES ***
           ANALYSIS  AS IN FIRST CYCLE, BUT MAY INVOLVE DEVELOPMENT OF FIXES
                     ON CARDS FOR CLEANED FORMAT OUTPUT FROM ICIFRONT, TO
```

BE CORRECTED AGAIN WITH ICIFRONT.

ICIFIX — FOR DATA TEXT. AS IN FIRST CYCLE, BUT MAY BE SUPERSEDED BY USE OF SLASH PROGRAM.

ICIFIX — FOR CORRECTION TEXT. MAY BE SUPERSEDED BY SLASH PROGRAM.

SLASH — MAY BE USED TO CONVERT CORRECTION TEXT ON CARDS TO PROPER FORMAT INPUT TO ICIFRONT'S CORRECTION TEXT FILE.

ICIFRONT — MAY BE USED (OR NOT) EITHER WITH SLASH OUTPUT OR ORIGINAL CORRECTION TEXT, EITHER WITH CLEANED FORMAT INPUT OR SCANNER FORMAT INPUT (4 COMBINATIONS).

ICISCAN — SAME AS FIRST CYCLE.

STRCT — SAME AS OTHER RUNS.

LADDR — SAME AS OTHER RUNS.

*** RETRIEVAL STEPS ***

ITEMS — TO SELECT SOME OR ALL OF SORT RECORDS REPRESENTED BY LADDR OUTPUT.

SORT — TO SORT SORT RECORDS IF NECESSARY.

MERGE — (ACTUALLY PART OF SORTING PROCESS) TO COMBINE OUTPUT OF SEVERAL SORT RUNS IF NECESSARY.

FRMAT — TO LIST SORT RECORDS.

**** INITIAL PROCESSING SEQUENCE ****

WHEN THE ORIGINAL MAGTAPE ARRIVES FROM THE SCANNER COMPANY THE FIRST THING TO DO IS TO MAKE A BACKUP COPY ON A STANDARD LABELLED TAPE USING THE LSPCNLSI PROCEDURE. BE SURE TO TAKE THE WRITE RING OFF THE SCANNER COMPANY'S TAPE BEFORE USING IT, FOR SAFETY.

THREE PARAMETERS MAY HAVE TO BE SPECIFIED ON THE EXEC CARD IN THE USUAL CASE ..

1) OUTVOL= IS THE VOLUME SERIAL NUMBER OF THE STANDARD LABELLED OUTPUT TAPE. DEFAULT IS 'A'.
2) OUTSEQ= IS THE OUTPUT FILE SEQUENCE NUMBER. DEFAULT IS 1.
3) OUTDSN= IS THE DATASET NAME OF THE OUTPUT FILE. THE DEFAULT IS 'LSPIC'.

LSPIC FILE NAMES ARE DESIGNED TO DESCRIBE THE INFORMATION IN THE FILE. THEY ARE 8 CHARACTERS LONG. EACH CHARACTER HAS THE FOLLOWING MEANING ..

(1) THE YEAR IN WHICH THE DATA IS STORED. A=71, B=72 ETC.
(2) THE MONTH IN THIS YEAR IN HEX, 5 = MAY, C = DEC., ETC.
(3) THE DAY OF THE MONTH BY THE FOLLOWING SYSTEM .. 1-9 = 1-9, A=10, B=11, ETC.
(4-6) THE LAST THREE DIGITS OF THE YEAR IN WHICH THE DATA IN THE FILE STARTS.
(7) THE MONTH IN WHICH THE DATA STARTS, USING THE SAME CODES AS AS FOR CHARACTER (2) ABOVE.
(8) THE DAY ON WHICH THE DATA STARTS, SAME CODES AS (3) ABOVE.

OF COURSE USE OF THIS NAMING CONVENTION IS NOT REQUIRED, IT IS ONLY INTENDED FOR SAFETY AND CONVENIENCE IN KEEPING TRACK OF TAPES. IF THE USER FINDS IT INCONVENIENT THEN IT SHOULD BE IGNORED.

THE SCANNER COMPANY'S TAPE SHOULD BE MOUNTED ON UNIT 182, AND THE OUTPUT TAPE ON UNIT 180. THESE CAN BE ALTERED BY USING INUNIT= AND OUTUNIT= ON THE EXEC CARD. EXAMPLE

```
//JJJK EXEC LSPCNLSL,OUTVOL=LSP009,OUTDSN=A4B777C6,INUNIT=181
```

HERE THE INPUT UNIT IS 181 INSTEAD OF THE DEFAULT 182, AND THE OUTSEQ= PARAMETER WAS NOT NEEDED SINCE THE FILE SEQUENCE WAS TO BE 1.

AFTER THE BACKUP STEP IS COMPLETED REMOVE THE SCANNER COMPANY'S TAPE. KEEP IT UNTIL 2 OTHER COPIES OF ITS DATA HAVE BEEN MADE AND CHECKED.

AFTER THE BACKUP, THE NEXT STEP IS TO SEPARATE THE CORRECTION TEXT FROM THE DATA TEXT, CREATING TWO NEW FILES WITH THE ICIFIX PROGRAM. WHEN CREATING THE DATA TEXT ENABLE TYPIST ENTRY CHECKING AND LEAVE A RECORD OF BAD PAGE ENTRIES ON THE CONSOLE

PRINTOUT. AT THIS POINT
DO NOT ALTER THE TEXT IN ANY WAY EXCEPT FOR SEPARATING THE
TWO KINDS OF TEXT AND POSSIBLY DELETING BLOCK FILLER AT THE
END OF THE FILE. THE REASON FOR NOT TAMPERING IS TO PRESERVE
THE ORIGINAL DATA TEXT FOR ACCOUNTING PURPOSES.
ICIFIX OPERATION IS DESCRIBED IN THE ICIFIX PROGRAM WRITEUP.
WHEN A REVISION IS MADE OF A BATCH THEN THE DATE OF STORAGE,
GIVEN IN COLUMNS 1-3 OF THE NEW DSN, SHOULD REFLECT THE DATE OF
THE REVISION.
AFTER SEPARATION PRINT THE CORRECTION TEXT USING THE LSPRINTS
PROCEDURE WITH OPT=DIRECTSN. IF YOU ARE ANXIOUS TO SEE THE
ORIGINAL DATA TEXT THEN, OF COURSE, GO RIGHT TO THE ICISCAN
LIST/LOG STEP.

THE VERY FIRST ICISCAN RUN, THE LIST/LOG STEP, IS DONE USING
THE LSPISCAN PROCEDURE, WHOSE COMMONLY REQUIRED PARAMETERS ARE ..

1) INVOL INPUT VOLUME LABEL.
2) INDSN INPUT DATASET NAME.
3) INSEQ INPUT FILE SEQUENCE NUMBER, DEFAULT IS 1.
4) OUTSEQ OUTPUT FILE SEQUENCE NUMBER, DEFAULT IS 1.

THE RUN DECK SHOULD HAVE 2 DD CARDS FOR SYSPRINT, 1 FOR THE
PRINTER DIRECTLY, THE OTHER FOR OUTPUT TO LSP002 ON UNIT 181.
PLACE THE ONE THAT YOU WANT TO BE USED AHEAD OF THE OTHER ONE
IN THE DECK. IF YOU ARE USING TAPE FOR SYSPRINT THEN IT IS
SOMEWHAT MORE EFFICIENT TO PLACE THE LSPRINT STEP IMMEDIATELY
AFTER THE ICISCAN STEP AND HAVE THE TAPE PRINTED IMMEDIATELY.
IN GENERAL A TAPE SHOULD BE USED FOR SYSPRINT ON THE VERY
FIRST RUN. THIS SPEEDS UP PROCESSING AND MAKES POUND SIGNS,
BRACKETS, AND EQUAL SIGNS VISIBLE WHEN LSPRINT PRINTS THE TAPE.
THIS WILL ALSO HELP LOCATE ANY ILLEGAL CHARACTERS THAT MAY
BE PRESENT. THE PRINT FILE LISTS, IN ADDITION TO THE RAW
INPUT TEXT, ALL ERROR MESSAGES, THE ABSOLUTE LINE NUMBER OF EACH
LINE, AN APPROXIMATE APPARENT DATE ASSOCIATED WITH EACH LINE,
AND THE APPROXIMATE STRCT/LADDR/ITEMS INPUT BLK NUMBER ASSOCIATED
WITH EACH LINE. THE TYPED PAGE NUMBER AND LINE NUMBER CAN
EASILY BE DETERMINED BY INSPECTION OF THE TEXT. IN SHORT THIS
LISTING PROVIDES A CROSS-REFERENCE TOOL CLOSELY CONNECTING
SUCH DIVERSE ENTITIES AS THE ORIGINAL DATA IN THE ICNDCN STAGE
ITSELF AND THE FINAL SORTED FORMATTED OUTPUT. FORMATTED OUTPUT
HAS GONE THROUGH AT LEAST 6 MAJOR TRANSFORMATIONS SINCE ITS
'ORIGIN' IN THE LONDON STAGE.
PRINTING OF ICISCAN'S LIST/LOG SHOULD BE DONE WITH THE
LSPRINTS PROCEDURE USING OPT=PRINTSN.
IT IS ALLRIGHT IF THE PRINTER IS ALLOWED TO RUN OUT OF
PAPER IN THIS STEP. THE OUTPUT WILL BE BULKY ANYWAY, BUT THE
NUMBER OF PIECES SHOULD NOT EXCEED 3 IF POSSIBLE. THE PRINTOUT
IS INTENDED MAINLY FOR REFERENCE IN THE ERROR CORRECTION PROCESS.
IF THE QUALITY IS NOT TOO LOW THEN, AFTER CORRECTION NOTES
ETC. HAVE BEEN MADE ON IT, THIS PRINTOUT SHOULD BE SENT TO DR.
SCHNEIDER. SCRAP PAPER SHOULD BE USED FOR THIS PRINTOUT IF
THERE ARE ANY SIZEABLE CHUNKS AVAILABLE.

HERE IS A LIST OF ITEMS REQUIRING SPECIAL ATTENTION IN
RUNNING STRCT, ESPECIALLY FOR THE FIRST TIME ON A NEW BATCH ..

1) CHECK THE MONTH ABBREVIATIONS USED IN LADDER REFERENCES,
AND CHANGE THE MONSTR CARDS ACCORDINGLY. SOME PARTS USE 'APR'
WHILE OTHERS USE 'APRIL', ETC. (MARCH, APRIL, MAY, JUNE, JULY,
AND AUGUST ARE THE ONES TO WATCH).

2) IT IS IMPORTANT TO BE SURE ABOUT THE SEASON ENDS IF YOU HOPE
TO RUN THE OUTPUT THROUGH LADDR. ONE LIKELY ERROR IS COUNTING
SECTIONS IN THE LAST PERFORMANCE OF A SEASON THAT ARE NOT PUT
OUT BY ICISCAN. ANOTHER IS OVERLOOKING THE DELETION OF AN ASTERISK
BY AT SIGNS. ILLEGAL SECTIONS THAT HAVE CAPITAL SECTYPE LETTERS
ARE PUT OUT BY ICISCAN AS REGULAR SECTIONS WITH LEGAL SMALL

LETTERS AS LONG AS THE SMALL LETTER IS THAT OF A LEGAL SECTYPE THAT HAS BEEN SELECTED ON THE OPTION CARD.

3) CHECK THE JCL CURSORILY TO MAKE SURE NO UNWANTED OVERRIDE CARDS ARE THERE. STRCT AND LADDR ARE SOMEWHAT DANGEROUS IN THIS RESPECT SINCE THEY ARE EXECUTED BY STRAIGHT JCL, NOT PROCEDURES. SPECIAL CARDS LEFT OVER FROM THE PREVIOUS RUN ARE LIKELY TO CAUSE TROUBLE.

THE LADDER STEP SHOULD NOT BE RUN UNTIL THE STRUCTURE STEP PRODUCES NO BAD 'DCHG' MESSAGES. THIS IS ONE IN WHICH THE DATE IS OFF BY 5 DAYS OR MORE. ADDITIONALLY ALL SEASON CHANGES MUST OCCUR PROPERLY. A MERE CHANGE IS NOT ENOUGH, IT MUST BE ENTIRELY CORRECT.

*** ERROR CORRECTION CYCLES ***

AFTER THE FIRST CYCLE THE ERROR CORRECTION LOOP ANALYSIS STEP MAY INCLUDE ANALYSIS OF CORRECTION TEXT ERRORS AND A COMPARISON OF ICISCAN AND/OR STRCT ERRORS IN USING ICIFRONT VS. NOT USING IT. THE MOST LIKELY SERIOUS ERROR IN CORRECTION TEXT INVOLVES A CORRECTION SPECIFICATION THAT INDICATES AN INCORRECT TYPIST OR ADVANCES THE PAGE NUMBER TOO MUCH. THIS CAN USUALLY BE SPOTTED WHEN THERE ARE VERY MANY 'CORRECTIONS' DONE TO ONE LINE IN THE UPDATE LOG PRINTOUT, OR WHEN THE PROGRAM STOPS AND THERE ARE STILL CORRECTIONS LEFT TO DO.

THE ICIFIX DATA TEXT STEP IS AS DESCRIBED FOR THE INITIAL PROCESSING SEQUENCE, EXCEPT THAT THE INPUT IS THE MOST RECENT VERSION OF THE DATA TEXT.

IF ICIFRONT IS TO BE USED AGAIN AND ERRORS HAVE BEEN FOUND PREVIOUSLY THEN THE ICIFIX CORRECTION TEXT CORRECTION TEXT STEP WILL HAVE TO BE DONE. DO NOT ENABLE TYPIST ENTRY CHECKING FOR THIS STEP. THIS MUST BE DONE BY HAND FOR CORRECTION TEXT.

USE THE LSPFRONT PROCEDURE TO RUN ICIFRONT. THE STANDARD UNITS ARE .. 181 FOR DATA TEXT, 182 FOR CORRECTION TEXT, AND 180 FOR CLEANED FORMAT OUTPUT. THE RESULTS MAY LOOK FAIRLY BAD AT FIRST, BUT INITIALLY ICIFRONT'S OUTPUT IMPROVES DRAMATICALLY WITH JUST A SMALL AMOUNT OF FIXING IN BOTH THE DATA AND CORRECTION TEXTS.

IF ICIFRONT WAS NOT USED THEN THE ICISCAN STEP SHOULD BE RUN WITH THE LSPISCAN PROCEDURE AS IN THE INITIAL PROCESSING STAGE. THE ICIFRONT CASE IS DESCRIBED IN THE FOLLOWING PARAGRAPHS ..

LSPSCLIN IS USED TO RUN THE CLEANED FORMAT OUTPUT THROUGH THE ICISCAN PROGRAM. THE INPUT FILE IS NAMED CLIN. THE DEFAULT INPUT UNIT IS 180 AS WITH THE LSPISCAN PROCEDURE. THE STANDARD OUTPUT UNIT AND VOLUME ARE ALSO IDENTICAL TO THOSE OF THE LSPISCAN RUN. TO AVOID OVERWRITING PREVIOUS OUTPUT YOU MAY USE A HIGHER OUTSEQ NUMBER (AS LONG AS THERE IS ROOM ON THE TAPE).

ICIFRONT SEEMS TO INCREASE THE ERROR RATE WHEN USED ON RAW DATA AND CORRECTION TEXT. HOWEVER WHEN THE DATA TEXT HAS HAD ALL OF ITS BAD TYPIST ENTRIES FIXED AND CERTAIN HORRIBLE CORRECTION TEXT ERRORS HAVE BEEN FIXED THEN USE OF ICIFRONT IS LIKELY TO BE BENEFICIAL. THE REAL STICKLER IS THAT IT MAY CREATE A NEW BAD DCHG ERROR. ANOTHER PROBLEM IS THAT AN ERROR MAY BE CORRECTED BY BOTH ICIFRONT AND ICIFIX. THE SOLUTION HERE IS TO DELETE THE CORRECTION FROM THE CORRECTION TEXT AND CORRECT THE DATA TEXT WITH ICIFIX.

ALTHOUGH NOT ESSENTIAL, IT IS PROBABLY BETTER TO PUT THE SYSPRINT OUTPUT ON TAPE AND USE LSPRINT RATHER THAN PRINT DIRECTLY, SINCE BRACKETS ARE SO RELEVENT AT THIS STAGE.

AFTER RUNNING ICISCAN A DECISION MUST BE MADE AS TO WHETHER TO GO ON WITH THE PRESENT CYCLE OR TO GO BACK AND USE ICIFIX TO CORRECT MAJOR ERRORS AND THEN RERUN ICISCAN AGAIN. THIS DECISION SHOULD BE BASED PRIMARILY ON THE SAME DETAILS AS ARE DESCRIBED IN THE ANALYSIS STEP. USUALLY IT IS BEST TO GO AHEAD AND RUN STRCT REGARDLESS OF ERRORS.

IN RUNNING STRUCTUR IT IS PROBABLY BEST TO JUST PRINT DIRECTLY ON THE PRINTER RATHER THAN USE UNIT 181 UNLESS IT IS ONE OF THE FINAL CYCLES OF THE DATA ENTRY STAGE. THERE IS NO TAPE PROCEDURE FOR THIS STEP, A STRAIGHT PROGRAM IS USED SINCE IT IS ANTICIPATED THAT NO FLEXIBILITY WILL BE NEEDED IN TAPE ALLOCATION ETC.

IN RUNNING STRCT THE SAME CONSIDERATIONS FOR OVERWRITING PREVIOUS OUTPUT APPLY AS WITH THE ICISCAN PROGRAM. SINCE STRCT IS RUN BY STRAIGHT JCL THE ACTUAL DD CARD IN THE RUN DECK WILL HAVE TO BE REPLACED OR SUPERSEDED BY ONE WITH THE SAME NAME IN THE DECK.

CONDITIONS FOR CONTINUING WITH LADDR RATHER THAN RESTARTING AT THE ANALYSIS STEP ARE ALWAYS THE SAME - NO BAD DCBG ERRORS, AND NO SEASON CHANGES MISSED.

*** RETRIEVAL STEPS ***

IF SORT RECORDS ARE TO BE PRODUCED THEN AFTER A SUCCESSFUL LADDR RUN YOU SHOULD ALMOST CERTAINLY RUN ITEMS RATHER THAN RESTART THE CYCLE. ONLY RESTART IF THERE IS AN EASY WAY TO GET RID OF A SUBSTANTIAL PORTION OF ERRORS.

TO SELECT ALL RECORDS IN A BATCH USE THE FOLLOWING STATEMENTS AFTER THE OPTION CARD ..

1) DATE RANGE, '5 1659 07 01 1800 12 31'
2) SE CARD TO CAUSE SELECTION OF ALL RECORDS, '1T'
3) GO CARD, '7'
4) STOP CARD, '9'

ITEMS SHOULD TAKE APPROXIMATELY 40 MINUTES PER OUTPUT REEL, PLUS SEVERAL MINUTES CHANGEOVER. BATCH 1 TOOK APPROXIMATELY 2 REELS. BEFORE RUNNING CHECK AND SEE IF ANY BATCH (ESP NO. 3) HAS A MUCH BIGGER OUTPUT CHAR COUNT IN THE LADDR PROGRAM, AND IF SO THEN ADD MORE NAMES TO THE OUTPUT VOLUME LIST ON THE ITSM DD STATEMENT. THE STEPNAME IS 'GO'.

THE SORT STEP, IF ANY, NEED NOT USE ANYTHING PECULIAR TO THE LONDON STAGE PROJECT SINCE THE PROGRAM INVOLVED IS SUPPLIED BY IBM, HOWEVER THERE ARE 2 PROCEDURES PROVIDED, LSPSORTT AND LSPMERGE. LSPSORTT IS USED FOR SORTS HAVING TAPE INPUT AND OUTPUT, USING DISKS AS INTERMEDIATE STORAGE AREAS. WITH SUFFICIENT SPACE ON THE DISKS IT CAN SORT APPRX 50,000 RECORDS PER RUN. THIS IS MORE THAN 1 TAPEFULL (UNBLOCKED) BUT LESS THAN 2 TAPES FULL. BATCH NUMBER 1 (WITH ONLY SLIGHT CORRECTIONS IN IT) HAS APPROXIMATELY 64,000 SORT RECORDS. A FAIRLY GOOD WAY TO GET AROUND THIS IS TO 1) DO N SEPARATE SORTS, EACH PRODUCING ONE TAPE OF OUTPUT FROM ONE TAPE OF INPUT, AND 2) THEN DO SUCCESSIVE 2 WAY MERGES OF THESE UNTIL ONLY ONE FILE (OF 1 OR MORE TAPES) IS LEFT. THIS MERGING PROCESS IS FACILITATED BY THE LSPMERGE PROCEDURE.

THE OUTPUT OF THE SORT STEP CAN BE USED AS INPUT TO ANOTHER SORT STEP, BUT IF OUTPUT IS BLOCKED THEN ONE SORT STEP MAY NOT BE ABLE TO HANDLE AN ENTIRE TAPE AT ONE SHOT. ONE SOLUTION IS TO USE THE LSPDBLOK PROCEDURE TO UNBLOCK THE DATA ONTO SEVERAL TAPES. AN UNBLOCKED TAPE CAN HOLD APPRX. 32,000 RECORDS. A TAPE BLOCKED BY A FACTOR OF 19 CAN HOLD APPRX. 96,000 RECORDS.

THE FRMAT PROGRAM PRINTS, IN A HOPEFULLY NEAT MANNER, OUTPUT FROM THE ITEMS OR SORT STEPS. THE TAPE PROCEDURE FOR THIS IS LSPFRMAT. THE NORMALLY REQUIRED PARAMETERS ARE INUNIT=, & INVOL=. INVOL= MAY ACTUALLY BE A LIST OF SEVERAL VOLUMES ENCLOSED IN QUOTES, FOR EXAMPLE ..

```
//F EXEC LSPFRMAT,INUNIT=180,INVOL='LSP018,LSP034'
```

DEFAULTS ARE INUNIT=182,INVOL=LSP018.

```
/*
/*
//ICIFORM  JOB 99999211,WCD,MSGLEVEL=(1,1)
```

```
//D EXEC LSPRINTS,OPT=CARDOCSN,MEM=DOCSFORM
//PRINT.IN DD DDNAME=INSYS
//PRINT.INSYS DD *
              **** ORIGINAL MAGTAPE INPUT FROM SCANNER ****
      THE SCANNER COMPANY PRODUCES NONLABELLED TAPE FORMATTED THUS ..
   DCB=(RECFM=FB,BLKSIZE=800,LRECL=80).
      EACH LOGICAL RECORD NOMINALLY CORRESPONDS TO A LINE OF TEXT
   ON THE CHINA DATA SHEETS.  THERE MAY BE AN OCCASIONAL EXTRA
   LINE OF 80 BLANKS, USUALLY OCCURRING AT THE END OF A PAGE.
   EACH PAGE IS SUPPOSED TO HAVE 30 LINES.  EACH BLK OF TEN 80 BYTE
   RECORDS IS ONE THIRD OF A PAGE.  IF THERE IS AN EXTRA BLANK LINE
   THEN THE OTHER LINES FOLLOWING IT ARE PUSHED DOWN EVEN INTO THE
   NEXT BLK IF NECESSARY TO KEEP THE BLKSIZE AT 800.
      THE TYPED PORTION OF EACH LINE IS NOMINALLY THE FIRST 75
   CHARS OF THE LINE, THE REMAINING 5 CHARS BEING BLANK ON THE CHINA
   DATA SHEETS EXCEPT IN CASE OF ERROR.  AFTER SCANNING TO TAPE
   THIS IS STILL GENERALLY TRUE, BUT OFTEN A TYPED DOUBLE QUOTE CHAR
   IS SCANNED AS 2 APOSTROPHES, THUS CAUSING THE 75TH TYPED CHAR TO
   SPILL OVER INTO COLUMN 76.  TO SOLVE THIS PROBLEM ANY PAIR OF
   CONTIGUOUS  APOSTROPHES FOUND ON A LINE IS REPLACED WITH A
   DOUBLE QUOTE CHAR BY THE ICISCAN PROGRAM.  OCCASIONALLY NONBLANK
   CHARS ARE PUT INTO COLUMN 76 OR BEYOND FOR OTHER REASONS, ALL
   ERRORS OF SOME SORT.  THESE ERRORS ARE ACCEPTED UP TO THE LAST
   NONBLANK CHAR ANYWAY.  NO ERROR MESSAGE  IS GENERATED FOR THIS.
      THE CHARACTERS ARE STANDARD EBCDIC EXCEPT FOR THE FOLLOWING ..
      THE BRITISH POUND SIGN IS AN EBCDIC NUMBER SIGN, HEX '7B'.
   IT IS PRINTED BY THE LSPRINT SN TABLES AS AN 'L' OVERPRINTED
   ON A MINUS SIGN.  IN THE ICIFIX PROGRAM THE 1052 PRINTS IT AS JUST WHAT
   IT IS INTERNALLY, THE NUMBER SIGN.  NEITHER THE HN NOR THE SN
   CHAIN CAN PRINT THE NUMBER SIGN CHARACTER.
      THE LEFT AND RIGHT BRACKETS ARE HEX 'AD' AND 'BD' RESPECTIVELY.
   THEY PRINT AS THE PARENTHESES OVERPRINTED WITH THE MINUS SIGN
   USING THE LSPRINT SN TABLES.  ON THE 1052 THEY ARE PRINTED AS
   THE 'LESS THAN' AND 'GREATER THAN' SYMBOLS RESPECTIVELY BY ICIFIX.
      THE CHARACTER THE SCANNER COMPANY IS SUPPOSED TO USE FOR
   NONRECS IS THE HEX '00'.  THIS IS PRINTED BY THE SN LSPRINT TABLES
   AS A ZERO OVERPRINTED ON THE PREFIX CHAR.  OTHER ILLEGAL CHARS
   PRINT AS THE USUAL 'H' ON 'I' CHAR.
/*
//FIXLIST  JOB 99999211,WCD,MSGLEVEL=(1,1)
//D EXEC LSPRINTS,OPT=CARDOCSN,MEM=DOCFIX
//PRINT.IN DD DDNAME=INSYS
//PRINT.INSYS DD *
                          ICIFIX GENERAL DESCRIPTION
       ICIFIX IS AN INTERACTIVE TEXT EDITING PROGRAM OPERATED FROM THE
   OPERATOR'S CONSOLE OF AN OS/360 SYSTEM.  IT IS LINE ORIENTED AND
   SPECIALIZED FOR FIXING A SMALL NUMBER OF ERRORS IN RAW DATA IN ICI
   FORMAT.  ITS INPUT IS A SEQUENTIAL FILE, IN, WITH AN LRECL OF 80.
   PRINCIPAL OUTPUT IS TO ANOTHER SEQUENTIAL FILE, TOUT, WITH THE SAME
   LRECL.  OUTPUT MAY ALSO GO TO THE PRINTER, AND, OF COURSE, COMMANDS
   AND PROGRAM RESPONSES ARE MADE ON THE CONSOLE.
       HERE IS A DESCRIPTION OF THE COMMANDS ..
            (PARTS OF COMMAND EXAMPLES INSIDE PARENTHESES ARE OPTIONAL)
   1)  'CHOP'     TERMINATES AN EDITING SESSION IMMEDIATELY, NOT PUTTING
         OUT ANY MORE LINES TO THE OUTPUT FILE.  THE CURRENT LINE IS NOT
         PUT OUT.  LINES PRECEDING THE CURRENT LINE HAVE ALREADY BEEN PUT
         OUT.  'CHOP' IS TYPED IN SMALL LETTERS.
   2)  'BACKUP'     TRANSFERS CURRENT LINE AND REMAINDER OF INPUT FILE TO
         OUTPUT FILE, THEN TERMINATES PROGRAM.  TYPED IN SMALL LETTERS.
   3)  'ND'     WHERE 'N' IS A NONNEGATIVE INTEGER, AND 'D' IS A SMALL'D'.
         CAUSES THE DELETION OF N LINES BY READING N LINES FROM THE
         INPUT FILE AND NOT PUTTING THEN OUT TO THE OUTPUT FILE.  IF
         'D' ALONE IS TYPED THEN '1D' IS ASSUMED.
   4)  'N'     WHERE 'N' IS A NONEGATIVE INTEGER.  READS N LINES,
         PUTTING INPUT INTO THE OUTPUT FILE, STARTING WITH THE CURRENT
         LINE.  THIS IS CALLED THE 'ADVANCE' COMMAND.
```

5) 'NL' WHERE 'N' IS A NONNEGATIVE INTEGER, AND L IS THE SAMLL LETTER, 'L'. SETS THE LINE COUNT LIMIT FOR A SEARCH TO N. N=0 MEANS NO SEARCH AT ALL. N=1 MEANS SAME LINE ONLY.

6) '¢C ('CENT SIGN-C') '¢' IS THE CENT SIGN. 'C' IS ANY CHARACTER STRING EXCEPT ONE CONTAINING A CENT SIGN OR ONE CONSISTING ONLY OF BLANKS. THIS COMMAND CAUSES THE PROGRAM TO SEARCH FOR THE CHARACTER (NOT THE STRING) FOLLOWING THE CENT SIGN FOR UP TO THE NUMBER OF LINES ALLOWED BY THE SEARCH LIMIT (SEE COMMAND NO. 5, ABOVE). IF THE CHAR IS FOUND THEN THE LINE IS TYPED UP TO AND INCLUDING THE CHAR OF THE SEARCH. YOU MAY THEN OVERLAY NEW TEXT, BEGINNING WITH THE SEARCH CHAR, UP TO THE END OF THE LINE. THIS IS DESCRIBED IN THE REPLACE COMMAND BELOW.

7) 'X¢(Y(¢)) THE REPLACE (AND POSSIBLY SEARCH AGAIN) COMMAND. THE BASIC FORM OF THIS COMMAND IS 'X¢' WHERE 'X' IS A CHAR STRING, AND '¢' IS THE CENT SIGN. THIS COMMAND IS INTENDED TO BE GIVEN AFTER A SEARCH COMMAND, FOR EXAMPLE COMMAND NO. 6. THE REPLACE COMMAND ALLOWS YOU TO REPLACE CHARACTERS, BEGINNING WITH THE SEARCH CHAR ITSELF (THE LAST CHAR TO BE TYPED IF THE SEARCH IS SUCCESSFUL) AND EXTENDING FOR THE LENGTH OF X OR TO THE END OF THE LINE, WHICHEVER IS SHORTER. IF THIS COMMAND IS NOT GIVEN IMMEDIATELY AFTER & IN RESPONSE TO A SEARCH COMMAND OR EQUIVALENT THEN THE REPLACEMENT STARTS WITH THE FIRST CHAR OF THE CURRENT LINE.

IF THE OPTIONAL PART, 'Y', IS TYPED (Y IS ANOTHER CHAR STRING NOT CONTAINING '¢', THE CENT SIGN) THEN, AFTER THE FIRST PART OF THE COMMAND IS EXECUTED, A SEARCH WILL BE MADE FOR THE FIRST CHAR OF Y. IF THE FIRST CHAR IS A BLANK THEN THE BLANK MUST BE FOLLOWED BY A NONBLANK CHARACTER, SUCH AS ANOTHER '¢' (CENT SIGN).

8) '¢X¢(Y(¢)) A CENT SIGN AS THE FIRST CHAR OF A COMMAND CONTAINING ANOTHER CENT SIGN MEANS TO CREATE A NEW LINE INSERTED BEFORE THE CURRENT ONE. THE NEW LINE MAY BE OF ANY LENGTH UP TO 75 CHARACTERS. IT WILL BE PADDED ON THE RIGHT WITH NULL CHARACTERS (PREFIXES, OR THE PL/1 'OR' SYMBOL). IF THERE ARE ANY NONBLANK CHARACTERS AFTER THE SECOND CENT SIGN THEN THE CHAR AFTER THE CENT SIGN IS TAKEN TO BE A NEW SEARCH CHAR AS IN THE REPLACE COMMAND.

CREATING A LINE, OR DELETING THEM FOR THAT MATTER IS TO BE AVOIDED IF POSSIBLE SINCE IT CAUSES MORE ICISCAN ERROR MESSAGES AND UPSETS THE LINE NUMBER CROSSREFERENCE SYSTEM.

9) 'CHECK' TURNS ON THE ERROR CHECKING FUNCTION. THIS IS TYPED IN SMALL LETTERS.

10) 'NOCHECK' TURNS OF THE ERROR CHECKING FUNCTION. WHEN THE PROGRAM STARTS, THIS FUNCTION IS OFF. TYPED IN SMALL LETTERS.

11) 'PRINT' AFTER EXECUTION OF THIS COMMAND LINES PUT INTO THE OUTPUT FILE WILL ALSO BE PUT INTO THE SYSPRINT FILE (IN OTHER WORDS, THEY WILL BE PRINTED ON THE PRINTER). THIS COMMAND IS TYPED IN SMALL LETTERS. THE 'PRINT' FUNCTION IS OFF WHEN THE PROGRAM STARTS. IT IS VERY EASY TO FORGET AND LEAVE THE PRINT ON WHEN EXECUTING A BIG LINE CHANGE COMMAND, THUS PRINTING A LARGE PORTION OF A BATCH, UNLESS YOU CANCEL. THIS IS THE MOST IMPORTANT POINT TO REMEMBER IN OPERATING THE PROGRAM, AS IT GENERALLY MEANS RESTARTING THE EDITING SESSION.

12) 'NOPRINT' THIS TURNS OFF THE PRINT FUNCTION ABOVE. THE 'PRINT' FUNCTION IS OFF WHEN ICIFIX BEGINS EXECUTION. TYPED IN SMALL LETTERS.

13) 'NG' WHERE 'N' IS A NONNEGATIVE INTEGER AND 'G' IS THE SMALL'G'. IF N IS GREATER THAN THE CURRENT LINE NUMBER THEN ADVANCE CURRENT LINE TO N. OTHERWISE DO NOTHING.

14) 'V' WHERE 'V' IS THE SMALL 'V'. MEANS TYPE THE ENTIRE CURRENT LINE ON THE CONSOLE. THE PURPOSE IS TO VERIFY CURRENT CONTENTS OF A LINE.

SEVERAL CHARACTERS TYPE DIFFERENTLY ON THE 1052 WITH THE ICIFIX PROGRAM THAN THEY DO WITH LSPRINT ON THE SN CHAIN. FIRST THE POUND SIGN, WHICH IS A MINUS OVERPRINTED ON AN 'I' ON THE PRINTER, IS THE NUMBER SIGN ON THE 1052. THE NUMBER SIGN IS USED EVERYWHERE IS THE SYSTEM AS THE INTERNAL CODE FOR THE POUND SIGN. THE LEFT AND RIGHT BRACKETS ARE PRINTED BY ICIFIX AS THE 'SMALLER THAN' AND 'GREATER THAN' SIGNS RESPECTIVELY. THE PREFIX, OR PL/1 'OR' SYMBOL IS THE NULL CHARACTER. IT IS PRINTED THE SAME WAY ON THE 1052 AS THE PRINTER. THE 1052 (AS OPPOSED TO ICIFIX) PRINTS MOST ILLEGAL CHARACTERS AS A QUESTION MARK. ISPRINT USES AN H OVERPRINTED ON AN I.

ICIFIX OVERALL OPERATION

THE SIMPLEST WAY TO USE UNITS FOR ICIFIX, IN ORDER TO KEEP UNIT USAGE FOR LATER PROGRAMS IN THE 180-182 GROOVE IS TO PUT THE INPUT TAPE FOR ICIFIX ON UNIT 182 BEFORE RUNNING. THIS INVOLVES AT MOST ONE REMOVAL AND ONE MOUNTING. THE NEW TAPE WILL THEN BE ON 180 READY FOR IMMEDIATE USE BY ICISCAN AFTER THE FIX.

DUE TO THE WASTE OF COMPUTER TIME INVOLVED WITH ICIFIX USE, ONLY THE FOLLOWING KINDS OF FIXES SHOULD, IN GENERAL, BE DONE ..

1) DATE WRONG OVER A PERIOD OF SEVERAL DAYS (USUALLY EITHER A YEAR OR A MONTH WRONG).
2) SEASON CHANGES NOT WORKING.
3) SECTIONS WHOSE ERROR CAUSES A LARGE NUMBER OF SUBSEQUENT LADDER FAILURES, SAY 20.

TO SAVE COMPUTER TIME, BE SURE TO PLAN AHEAD, LISTING SPECIFIC CORRECTIONS, SEARCH CHARS, IF ANY, AND LINE NUMBERS, AND PUTTING MARKERS IN THE PAGES OF THE DATA LISTING WHERE ERRORS ARE TO BE CORRECTED. REMEMBER THAT ICIFIX CORRECTIONS MAY BE ONLY TEMPORARY. ALL MAY BE REDONE AFTER CORRECTIONS IN CORRECTION TEXT HAVE BEEN APPLIED TO THE ORIGINAL UNREVISED BACKUP FILE FROM ICI. ABOUT 2 CORRECTIONS WITH ICIFIX PER YEAR OF DATA SUFFICED QUITE WELL FOR BATCH NO. 1.

IT IS NOT DESIRABLE TO DELETE THE CORRECTION TEXT, JUST GET AROUND IT WITH ICISCAN SOMEHOW. WE MAY WANT TO HAVE CORRECTION TEXT CONVENIENTLY AVAILABLE ASSOCIATED WITH ITS REVISED DATA TEXT AT SOME POINT. DELETE THE CORRECTION TEXT IF NECESSARY, THOUGH. IF YOU MUST MESS WITH IT YOU MIGHT AS WELL DELETE IT ALL.

YOU MAY WANT TO PRINT THE CORRECTIONS AND STICK THIS IN THE LAST FULL LISTING OF DATA FOR REFERENCE.

WHEN ICIFIX STARTS IT DISPLAYS THE FOLLOWING MESSAGE (THE STANDARD MESSAGE) ON THE CONSOLE ..

N '17C'

WHERE 'N' IS THE LINE NUMBER OF THE CURRENT LINE (ALWAYS LINE 1 AT THE START), AND '17C' REPRESENTS A CHARACTER STRING CONSISTING OF THE FIRST 17 CHARS OF LINE N.

THIS STANDARD MESSAGE IS THE RESPONSE, EXCEPT FOR ERRORS TO EVERY COMMAND NOT RESULTING IN A SEARCH. THE STANDARD MESSAGE IS ALSO GIVEN AFTER EACH DISPLAYED ERROR MESSAGE.

THE OTHER NON-ERROR MESSAGE IS THE SEARCH MESSAGE, WHICH IS IDENTICAL TO THE STANDARD MESSAGE EXCEPT THAT ITS PORTION IS USUALLY NOT 17 CHARS LONG, AND THERE IS A CAPITAL 'S' AT THE END OF THE MESSAGE, IN ORDER TO VERIFY THAT IT IS INDEED THE SUCCESSFUL SEARCH MESSAGE. THE LAST CHAR OF A SUCCESSFUL SEARCH MESSAGE IS THE SEARCH CHAR ITSELF. IF A SEARCH IS UNSUCCESSFUL THEN THE STANDARD MESSAGE IS GIVEN, EXCEPT IN THE EVENT THAT THE SEARCH GOES TO THE END OF THE FILE, IN WHICH CASE NO INDICATION IS GIVEN.

THE STANDARD SEQUENCE FOR CORRECTION BY REPLACEMENT IS AS FOLLOWS ..

1) GO TO THE LINE INVOLVED WITH AN 'NG' COMMAND.
2) IF YOU HAVE DOUBTS, THEN VERIFY THE LINE WITH THE 'V' COMMAND, ALTHOUGH THIS SHOULDN'T BE NECESSARY SINCE YOU SHOULD HAVE THE LISTING TO REFER TO DURING THE SESSION.
3) LOCATE EITHER AT OR SHORTLY BEFORE THE CHAR(S) TO BE REPLACED. IF THE CHARS ARE NEAR THE BEGINNING OF THE LINE THEN YOU ARE ALREADY LOCATED THERE SINCE A REPLACE COMMAND NOT IMMEDIATELY FOLLOWING A SEARCH-TYPE COMMAND (IMMEDIATELY) WILL START REPLACING

AT THE BEGINNING OF THE LINE.
4) REPLACE THE ERRONEOUS CHARS WITH NEW ONES. IF THE CHARS ARE SIMPLY TO BE DELETED THEN TYPE THE STRAIGHT UP AND DOWN CHAR ABOVE THE COMMA KEY. THIS IS CALLED THE NULL CHAR, OR PREFIX, OR PL/1 'OR' SYMBOL. IT WILL TAKE UP SPACE IN THE INPUT TO ICISCAN, BUT ICISCAN WILL DELETE IT VERY QUICKLY, AND IT WILL NOT BE PART OF THE LOGICAL TEXT.
5) YOU MAY WANT TO VERIFY THE CHANGE.

ICIFIX ERROR MESSAGES

ICIFIX HAS 4 STANDARD ERROR MESSAGES WHICH ARE DISPLAYED ON THE 1052 ..

'NO COMMAND' THE LAST RESPONSE TYPED IN DID NOT HAVE ANY NONBLANK CHARS IN IT.

'NO CENT SIGN' THE RESPONSE DID NOT HAVE A CENT SIGN, AND IT WAS NOT ONE OF THE LEGAL COMMANDS NOT CONTAINING A CENT SIGN.

'BAD TYPIST'S ENTRY' AN ASTERISK IN COLUMN 1 OF A LINE WAS FOLLOWED BY A CAPITAL LETTER, BUT EITHER THE LETTER WAS NOT FOLLOWED BY 1 OR MORE DECIMAL DIGITS OR ELSE THERE WERE 1 OR MORE DIGITS BUT THEY WERE NOT FOLLOWED BY AND IMMEDIATELY TERMINATED BY A BLANK. THIS MESSAGE IS DISPLAYED ONLY IN THE 'CHECK' MODE. WHEN THE PROGRAM BEGINS EXECUTION IT IS IN THIS MODE.

'TYPIST'S ENTRY OUT OF ORDER X Y' THE LAST ENTIRELY LEGITIMATE TYPIST ENTRY ENCOUNTERED BY ICISCAN IN CHECK MODE DID NOT HAVE A NUMBER 1 LESS THAN THE VALUE OF THE CURRENT LEGITIMATE PAGE ENTRY. THE NUMBERS, X AND Y, ARE THE PREVIOUS AND CURRENT PAGE NUMBERS RESPECTIVELY.

ICIFIX OPERATES WITH STRINGRANGE AND SUBSCRIPTRANGE ENABLED, AND HAS 'PUT DATA' TYPE SNAP ON CONDITIONS FOR THESE ERRORS.

```
/*
//DOCFRONT JOB 99999211,WCD,MSGLEVEL=(1,1)
//B EXEC LSPRINTS,OPT=CARDOCSN,MEM=DOCFRONT,DSSP=OLD
//PRINT.IN DD DDNAME=INSYS
//PRINT.INSYS DD *
```

*** ICIFRONT AND SLASH **

ICIFRONT IS AN APPENDAGE TO THE DATA ENTRY SYSTEM. ICIFRONT'S MAIN PURPOSE IS TO PERFORM CORRECTIONS SPECIFIED EITHER BY THE CHINA DATA CORRECTION SYSTEM SPECIFICATIONS ARE DESCRIBED IN 'CORRECTION INSTRUCTIONS FOR THE LONDON STAGE PROJECT'.

THE DIFERENCES BETWEEN THESE INSTRUCTIONS AND THOSE FOR CARD INPUT ARE DESCRIBED HERE.

THE MAIN PROBLEM WITH CARDS IS THAT KEYPUNCHES WITH LOWER CASE CAPABILITIES ARE NOT AVAILABLE TO US. TO REMEDY THIS THE FOLLOWING RULE IS USED ..

ANY CHARACTER THAT IS TO BE LEFT UNALTERED IS PRECEDED BY THE SLASH CHARACTER ,. WHENEVER THE SLASH IS PRESENT THEN THE CHARACTER FOLLOWING IT (EVEN ANOTHER SLASH) WILL BE LEFT EXACTLY AS IT IS, AND THE SLASH WILL BE DELETED. TO GET ONE SINGLE SLASH IN THE OUTPUT USE TWO CONSECUTIVE SLASHES. ALL LETTERS WILL BE TRANSLATED TO LOWER CASE UNLESS THEY ARE PRECEDED BY A SLASH. IN THIS CASE THE SLASH WILL BE REMOVED AND THE LETTER WILL NOT BE ALTERED.

THIS SLASH PROCESSING WILL BE DONE BY A PROGRAM CALLED 'SLASH'. SLASH'S INPUT FILE IS SYSIN. THE OUTPUT FILE IS 'CRCT'. SLASH HAS 1 OPTION, OPTION 1. IF OPTION 1 IS SPECIFIED THEN THE OUTPUT WILL BE LISTED ON SYSPRINT IN ADDITION TO GOING TO THE FILE CRCT.

THE SYSIN FILE IS A STREAM FILE. THE FIRST 80 BYTES MUST BE BINARY DIGITS, 0'S EXCEPT FOR POSSIBLY OPTION NO. 1. THE 81ST CHAR MUST BE A '*' STARTING THE FIRST CORRECTION STATEMENT. THE DATA MUST BE FOLLOWED BY AT LEAST 80 BLANKS (ONE CARD) AT

```
THE END OF THE FILE.  THERE IS NO PROCEDURE FOR SLASH.

     ICIFRONT'S PROCEDURE IS CALLED LSPFRONT.  ITS DEFAULTS ARE ..
     DATUNIT=181,DATVOL=LSP025,DATDSN=LSPDT,DATSEQ=1,
     CORUNIT=182,CORVOL=LSP026.CORDSN=LSPCR,CORSEQ=1,
     OUTUNIT=180,OUTVOL=LSP027,OUTDSN=LSPCL,OUTSEQ=1

     THE PARAMETERS USUALLY REQUIRED ON THE EXEC CARD ARE ..
     DATVOL,CORVOL, & OUTVOL.
     IT IS SUGGESTED THAT, IF ICIFRONT IS TO BE USED, IT BE
TRIED FIRST IN THE FIRST CYCLE AFTER THE INITIAL PROCESSING STAGE.
/*
//SCANLIST JOB 99999211,WCD,MSGLEVEL=(1,1)
//D EXEC LSPRINTS,OPT=CARDOCSN,MEM=DOCISCAN
//PRINT.IN DD DDNAME=INSYS
//PRINT.INSYS DD *
                         GENERAL OVERVIEW OF ICISCAN
      ICISCAN'S GENERAL PURPOSE IS TWOFOLD ..  1) TO SERVE AS A FRONT
END FOR THE ODDITIES OF THE REMOTE EDITOR / CHINA DATA / OPTICAL
SCANNER DATA PREPARATION SYSTEM.  2) TO DIVIDE THE TEXT INTO ITS
3 KINDS OF COMPONENTS .. STRUCTURED SECTIONS, EXTRANEOUS TEXT, AND
INDEX ENTRIES IN EXTRANEOUS TEXT.  COMMENT SECTIONS ARE CONSIDERED
EXTRANEOUS TEXT, BUT WITH THE FRONT AND REAR DELIMITERS REMOVED.
ICISCAN ALSO CHECKS FOR PAGE ENTRIES - BOTH DECIMAL AND ROMAN, BUT
THESE ARE NOT SUPPOSED TO BE IN THE DATA AT PRESENT.  THEY ARE
INTENDED FOR FUTURE USE IN THE LONDON STAGE INDEX FOR INTRODUCTIONS.
      ICISCAN ALSO COLLECTS STATISTICAL DATA FOR GENERAL REFERENCE AND
SYSTEM OPTIMIZATION PURPOSES AS WELL AS FOR CALCULATING PAYMENTS TO
THE TYPING AND SCANNING COMPANIES.  ICISCAN OPTIONALLY PRODUCES
EXTENSIVE PRINTOUTS OF PROCESSING TO FACILITATE LATER CORRECTION OF
INPUT TEXT BY A HUMAN.
      ICISCAN HAS AN OPTION CHAIN IN COLUMNS 17-23.  OPTIONS ON A CARD
ARE IN EFFECT UNTIL THE LINE WITH THE LINE NUMBERS IN THE CHAIN
IS READ IN.  AT THAT TIME THE PROGRAM, REALIZING THAT NEW OPTIONS
ARE TO BE APPLIED, READS IN THE NEXT OPTION CARD FROM THE SYSIN
FILE.  THESE OPTIONS ARE THEN IN EFFECT AS THE LINE READIN
PROCESSING IS CARRIED OUT.  TO MAKE OPTIONS ON A CARD REMAIN IN
EFFECT UNTIL THE END OF THE PROGRAM, PUT 0'S IN COLS 17-23, SINCE
LINE 0 DOES NOT EXIST AND WILL THEREFORE NEVER BE READ IN.
      IN ADDITION TO OPTIONS AND THE LINE NUMBER FOR THE NEXT
SET OF OPTIONS, IF ANY, EACH ICISCAN OPTION CARD HAS 2 OTHER
THINGS ON IT ..  1) A LIST OF THE SMALL LETTERS THAT ARE
ACCEPTABLE AS SECTION TYPE LETTERS, LOCATED IN COLS 24-49, AND
2) A LIST OF THE SECTION TYPES THAT ARE TO BE PUT OUT BY ICISCAN,
IN COLS 51-76.  BOTH LISTS OF SMALL LETTERS ARE REQUIRED TO
START IN THE FIRST COLUMN OF THEIR RESPECTIVE FIELDS AND TO
HABE AT LEAST THE 'P' IN THEM.  THE LIST OF LEGAL SECTION
TYPES, IN FACT, SHOULD ALWAYS BE THE SAME.  AT PRESENT THIS MEANS
THE LETTERS, 'PABCDEMOSTUI'.
     AT THE HEAD OF THE OPTION CHAIN IS A SPECIAL OPTION CARD
CALLED THE STARTER CARD.  IT IS MUCH LIKE OTHER OPTION CARDS,
BUT ITS PURPOSE IS ONLY TO START THE PROGRAM WITH THE RIGHT
INPUT FILE, CLIN OR IN.  WHEN ICISCAN STARTS, IT FIRST READS
IN THE STARTER CARD AND PROCESSES IT JUST LIKE ANY OTHER OPTION
CARD.  THIS HAPPENS BEFORE ANY OTHER INPUT.    THE OPTION CARD
SHOULD HAVE A 1 IN COLUMN 15, THE APPROPRIATE OPTION FOR CLIN
OR IN IN COLUMN 7, AND 0000001 IN COLS 17-23.  IT SHOULD ALSO
HAVE A NONBLANK CHAR IN COL 24, AND IN COL 51.  IT MAY AS WELL
HAVE THE FULL SET OF SECTYPE LETTERS IN THESE LAST TWO FIELDS.
       THE FOLLOWING OPTIONS ARE RECOMMENDED FOR A LIST/LOG OF
ICIFIXED SCANNER FORMAT INPUT ..  3, 4, 11, 12, 16.
      THE OPTION CARDS FOR THIS RUN WOULD BE
'00110000001100110000001PABCDEMOSTUI                    PABDEMOSTUI'  AND
'00110000001100010000001PABCDEMOSTUI                    PABDEMOSTUI'
      (THE LETTERS BEING SMALL LETTERS PUNCHED BY USE OF THE
       MULTIPUNCH BUTTON).
```

FOR RERUNS, EITHER FROM CLEAN INPUT OR SCANNER FORMAT INPUT, THE OPTIONS WILL VARY.

1 GENERAL LOGIC OVERVIEW OF ICISCAN

THERE ARE 2 MUTUALLY EXCLUSIVE PRIME INPUT FILES, CLIN & IN. IN'S INPUT IS IN THE FORM OF 80 BYTE LOGICAL RECORDS. EACH INPUT BLK (OR LINE) IS NOMINALLY 75 BYTES OF TYPING FOLLOWED BY 5 BYTES OF BLANKS. A NUMBER OF THINGS MAY AFFECT THIS, HOWEVER. THE PROCEDURES APPLIED TO EACH INPUT LINE ARE DESCRIBED IN 'ORDER OF LINE READIN PROCESSING'.

CLIN'S LINE IS OF VARIABLE LENGTH AND IS IN A STREAM FILE. EACH LINE STARTS WITH A 20 BYTE HEADER OF CONTROL INFORMATION. THE FORMAT IS DOCUMENTED IN 'STANDARD DATA FILES'. THE LAST LINE OF A FILE IS INDICATED BY A NEGATIVE LINE NUMBER WITHIN CURRENT PAGE. THIS LAST LINE CONTAINS NO DATA, ALTHOUGH ITS DATA PORTION MAY BE OF NONZERO LENGTH.

SINCE THE INPUT LINE IS OFTEN ONLY PART OF A SECTION, ICISCAN CONSISTS ESSENTIALLY OF 2 'COROUTINES', THE INPUT COROUTINE AND THE PROCESSING & OUTPUT COUROUTINE. EXCEPT FOR PRIMING BY THE INPUT ROUTINE, THE OUTPUT AND PROCESSING ROUTINE IS THE DOMINANT ONE, INVOKING THE LINE READIN ROUTINE WHENEVER THERE ARE NO MORE DELIMITERS LEFT IN THE CURRENT BUFFER.

PURE FILE OUTPUT IS U-FORMAT BLKS UP TO 3625 BYTES IN LENGTH, EACH BLK BEING ONE SECTION. AT PRESENT, INDEX ENTRIES ARE NOT PUT OUT, ALTHOUGH THEY MAY BE OPTIONALLY PRINTED, SINCE THE FULL INFORMATION NEEDED TO PROCESS THEM HAS NOT YET BEEN EDITED INTO THE TEXT. CLOUT FILE OUTPUT IS WRITTEN AFTER ALL LINE READIN CHANGES HAVE BEEN MADE TO THE INPUT LINE. IT IS POSSIBLE TO CREATE A BACKUP OF A CLEAN FILE BY SPECIFYING OPTIONS 7 & 14 SIMULTANEOUSLY.

THE PROCESSING ROUTINE CONCERNS ITSELF PRIMARILY WITH MODES AND DELIMITERS. AT THE BEGINNING OF A RUN, TEXT STARTS IN STRUCTURED MODE, AND ALL TEXT FROM THE BEGINNING OF THE DATA UNTIL THE FIRST DELIMITER IS STRUCTURED TEXT. DEPENDING ON THE DELIMITER THE NEXT PIECE OF TEXT (BETWEEN THE DELIMITER AND THE DELIMITER AFTER IT) WILL BE IN SOME PARTICULAR MODE, FOR EXAMPLE 'STRUCTURED, OR 'EXTRANEOUS DUE TO LEFT PAREN'. IN THE GENERAL CASE THERE ARE TWO FACTORS DETERMINING A NEW MODE. 1) THE PREVIOUS MODE. 2) THE DELIMITER DEFINING THE START OF THE NEW TEXT. AS A SPECIAL CASE, IF THE DELIMITER IS A '*' AND IT IS IMMEDIATELY FOLLOWED IN THE CLEANED LOGICAL TEXT BY A SMALL 'C' THEN THE NEW MODE WILL BE 'COMMENT SECTION'. THERE ARE 10 MODES, INDICATED BY VALUES RANGING FROM -5 TO +4 IN THE VARIABLE, 'MODE'.

THE 10 MODES ARE NAMED IN THE LISTING BESIDE THE ARRAY, 'NEWMODE', WHICH DEFINES THE NEW MODE OF FOLLOWING TEXT (EXCEPT FOR COMMENT SECTIONS AS NOTED ABOVE) BASED UPON THE PREVIOUS MODE AND THE DELIMITER ENDING THE PREVIOUS PIECE OF TEXT.

1 ORDER OF LINE READIN PROCESSING FOR ICISCAN

1) READ LINE. IF CLEANED INPUT (OPTION 7) THEN GO TO 4.
2) IF BLANK LINE THEN GO TO 1, NOT COUNTING IT AS A LINE.
3) COUNT IT AS A LINE.
4) IF LINE NUMBER IS SAME AS ON CURRENT OPTION CARD THEN READ IN A NEW OPTION CARD AND PROCESS IT, DOING AN IMMEDIATE STOP IF OPTION NO. 1 IS SPECIFIED.
5) IF OPTION 15, THE LINE SKIP OPTION, THEN GO TO 1.
6) IF OPTION 7, THE CLEANED INPUT OPTION THEN GO TO LINE 22.
7) COUNT AS A LINE ON CURRENT PAGE.
8) IF ILLEGAL CHAR CHECK OPTION NO. 11 THEN DO THE CHECK, CONVERTING EACH ILLEGAL CHAR IN LINE TO A QUESTION MARK.
9) IF OPTION 3 THEN LIST LINE AS IT PRESENTLY STANDS.
10) DELETE ANY NULL CHARS IN LINE, KEEPING COUNT.
11) CHECK FOR REGULAR TYPIST ENTRY (VIA 'AT' SIGN-CAPITOL LETTER) & PROCESS IT IF PRESENT. POSSIBLY REENTERING REGULAR TEXT MODE FROM CORRECTION TEXT MODE.
12) GO TO 1 IF IN CORRECTION TEXT MODE.
13) TAKE CARE OF TRIPLE 'AT'S.
14) IF LINE STARTS WITH 2 'AT'S THEN CHANGE THEM TO 2 SPACES.
15) IF LINE STARTS WITH 1 'AT' THEN CHANGE IT TO A SPACE.

16) IF LINE STARTS WITH '**' THEN ENTER CORRECTION TEXT MODE AND GO TO 1 AGAIN.
17) CONVERT PAIRS OF SINGLE QUOTES TO A DOUBLE QUOTE CHAR, KEEPING COUNT AS WITH NULL CHARS.
18) CHECK LAST 5 CHARS OF LINE FOR NONBLANK CHARS AND KEEP ANY FOUND THERE.
19) CHOP OFF TRAILING BLANKS AND TYPIST ENTRY, IF ANY, FROM LINE.
20) PROCESS DOUBLE 'AT'S.
21) PROCESS SINGLE 'AT'S.
22) IF OPTION 14 THEN PUT OUT LINE TO FILE CLOUT.
22A) COUNT UPPER CASE CHARS REMAINING IN LINE (FOR CLEANED CAP COUNT).
23) COUNT TOTAL CHARS REMAINING IN LINE (FOR TOTAL CLEANED CHAR COUNT).
24) APPEND TO WORK BUFFER.

NOTES ..

1) ILLEGAL SECTYPE CHARACTERS THAT ARE CAPITAL LETTERS ARE CHANGED TO SMALL LETTERS . THE RESULTING SECTIONS ARE THEN TREATED LIKE OTHER SECTIONS. IF THE SECTION IS OF A LEGAL TYPE THEN IT WILL BE PUT OUT.
2) AS NOTED IN LINE READIN PROCESS NO. 18, NONBLANK CHARACTERS IN FOLLOWING THESE NONBLANK CHARACTERS ARE NOT RETAINED.

```
//STRUDOC  JOB 99999211,WCD,MSGLEVEL=(1,1)
//B EXEC LSPRINTS,OPT=CARDOCSN,MEM=STRCTDOC
//PRINT.IN DD DDNAME=INSYS
//PRINT.INSYS DD *
```

OVERVIEW OF STRUCTUR

STRUCTUR (PGM NAME= STRCT) TAKES OUTPUT FROM ICISCAN (PURIFIED TEXT) AND SYNTACTICALLY ANALYZES IT INTO ITS COMPONENT ITEMS AND GROUPS. EACH ITEM PUT OUT WILL BE EITHER A ROLE, ACTOR, TIME ENTRY, LONDON STAGE PAGE ENTRY, SECTION TITLE, THEATRE, DATE, CAST GROUP LADDER ENTRY, 'SEE' LADDER ENTRY, OR TITLE LADDER ENTRY. THERE ARE TWO KINDS OF GROUPS, TITLE GROUPS, AND CAST GROUPS. A TITLE GROUP CONTAINS THE TITLE, IF ANY, OF A SECTION, AND THE TITLE LADDER ENTRY OR 'SEE' LADDER ENTRY, IF ANY, OF A SECTION. IT MAY, LIKE CAST GRPS, CONTAIN A LONDON STAGE PAGE ENTRY. A CAST GRP MAY CONTAIN 0 OR 1 TIME ENTRIES, 0 OR MORE ROLES, 0 OR 1 CAST GRP LADDER ENTRIES, AND 0 OR MORE ACTORS IN THAT ORDER, EXCEPT THAT ROLES MAY FOLLOW CAST GROUP LADDER ENTRIES AS LONG AS ONE ROLE PRECEDES THE CGLE IN THE GROUP. THIS EXCEPTION MAY BE DUE TO A BUG, BUT SHOULD DO NO GREAT HARM. A CAST GROUP ALWAYS HAS AT LEAST ONE ITEM, AND ALL ITEMS IN A CAST GROUP SHOULD BE OF NONZERO LENGTH IN THEIR DATA PORTIONS. A TITLE ITEM MAY BE OF ZERO LENGTH IN THE DATA PORTION, IN FACT EACH UNTITLED SECTION HAS A TITLE OF ZERO LENGTH IN ITS OUTPUT BLK.

EACH INPUT BLK MUST CONTAIN 1 OR MORE COMPLETE SECTIONS. EACH OUTPUT BLK IS ONE SECTION. PARTS OF SECTIONS ARE NOT ALLOWED, AND SHOULD NOT BE PRODUCED BY ICISCAN. EACH OUTPUT BLK OF SAVEDT, ON THE OTHER HAND, SHOULD CONSIST OF 1 OR MORE COMPLETE SECTIONS, BUT OCCASIONALLY DUE TO BAD INPUT SAVEDT WILL PUT OUT A SECTIONLESS BLK OR ONE WITH PART OF A SECTION IN IT.

FOR SEQUENTIAL OUTPUT (OPTION 7) THE KEYS (12 BYTES) ARE APPENDED TO THE FRONT OF EACH BLK, AND 2 BLANKS ARE APPENDED IN FRONT OF THE KEYS TO MAKE AN ABSOLUTE MINIMUM OUTPUT BLKSIZE OF 18 BYTES INCLUDING THE MINIMUM POSSIBLE OF 4 DATA BYTES CONSISTING OF AN EMPTY TITLE GRP AND TITLE ITEM. THE BLANKS MAY BE CONSIDERED 'RESERVED' SINCE AT PRESENT THEY ARE DISCARDED ON INPUT BY LADDER.

STRUCTUR REQUIRES 3 SETS OF RECORDS (HEREAFTER CALLED 'CARDS') IN ITS SYSIN FILE. THE FIRST 12 CARDS, CALLED MONSTR CARDS, DESCRIBE THE NAMES USED IN THE BATCH FOR THE MONTH PART OF A LADEER REFERENCE. THE MONTH NAME MAY BE EITHER IN ALL CAPS OR ELSE ENTERED EXACTLY AS IN THE DATA IN UPPER AND LOWER CASE, DEPENDING ON WHETHER OPTION 9 IS SPECIFIED. EACH CARD HAS 2 DATA FIELDS. THE FIRST FIELD IS COLUMN 1, AND CONTAINS A NUMBER BETWEEN 1 AND 9 DEFINING THE LENGTH OF THE MONTH NAME.

THE NAME FIELD ITSELF STARTS IN COLUMN 2. THE REMAINDER OF THE CARD IS IGNORED. THE MONTH NAMES MUST BE IN ORDER FROM JANUARY TO DECEMBER, AND ALL MUST BE PRESENT FOR EVERY RUN OF STRCT. IF OPTION 9 IS USED, THE MONTH NAMES MUST BE COMPLETELY UPPERCASE.

FOLLOWING THE MONSTR CARDS ARE 1 OR MORE SEASON CHANGE CARDS. THE LAST SEASON CHANGE CARD CONTAINS, STARTING IN COLUMN 1, THE FOLLOWING PATTERN OF 0'S AND X'S .. '00000000XXXXXXXX00'. THE REMAINDER OF THE CARD IS IGNORED. ALL OTHER SEASON CHANGE CARDS ARE IN THE FOLLOWING FORMAT, STARTING IN COLUMN 1 ..

YYYYMMDDTTTTTTTTLL WHERE 'YYYYMMDD' IS THE DATE THE LAST PERFORMANCE TOOK PLACE. 'LL' IS THE NUMBER OF SECTIONS THAT ICISCAN PUT OUT FOR THE PERFORMANCE. IT IS EASY TO MAKE A BIG MISTAKE IN THESE CARDS BY 1) INCORRECTLY FIGURING (ESPECIALLY OVERESTIMATING) THE NUMBER OF SECTIONS PRESENT AT THE LAST PERFORMANCE BY FAILING TO NOTICE SIMETHING LIKE A DOUBLE 'AT' THAT DELETES THE ASTERISK STARTING THE SECTION (UNDERESTIMATING IS NOT SO SERIOUS). 2) FAILING TO ACCOUNT FOR THE KINDS OF SECTIONS THAT ARE NOT SELECTED FOR OUTPUT BY ICISCAN, THAT IS, IF THE 'SECTION TYPES WANTED' FIELD OF ICISCAN'S OPTION CARD CONTAINS ONLY 'AP', THEN 'D' SECTIONS WILL NOT BE PUT OUT BY ICISCAN OR COUNTED BY STRCT. 'TTTTTTTT' IS THE THEATRE, TYPED IN LOWER CASE AND PADDED ON THE RIGHT WITH BLANKS. THE REMAINDER OF EACH CARD FOLLOWING THE 'LL' FIELD IS IGNORED.

FOLLOWING THE LAST SEASON CHANGE CARD IS THE OPTION CHAIN, WHICH CONSISTS OF 1 OR MORE CARDS. THE OPTION CARD HAS 32 OPTION LOCATIONS IN COLS 1-32. IN COLS 33-38 IS THE LINK TO THE INPUT BLK, IF ANY, FOR WHICH THE NEXT OPTION CARD WILL BE USED. IF THERE IS NO NEXT SET OF OPTIONS THEN 0'S ARE PREFERRED IN COLS 33-38. THE OPTIONS ON THE NEXT CARD ARE READ IMMEDIATELY AFTER THE READING OF THE BLK WHOSE NUMBER IS LISTED IN THE CURRENT OPTION CARD. IN PARTICULAR, THE BLK NUMBERED IN THE CURRENT OPTION CARD WILL BE SKIPPED IF THE NEXT OPTION CARD SPECIFIES THE SKIP OPTION. THE VERY FIRST OPTION CARD IN THE CHAIN IS READ IN AND TAKES EFFECT RIGHT AFTER THE FIRST BLK OF INPUT FROM THE FILE, PURE.

RECOMMENDED OPTIONS FOR ERROR CHECKING ARE .. 5,6,7,9,27.

IN BATCH 1, STRCT PRODUCED RELATIVELY FEW ERROR MESSAGES, BUT IF ERROR MESSAGES ARE FREQUENT IN SOME OTHER BATCH THEN OPTION 27 SHOULD BE THE FIRST TO GO.

PROGRAM NOTES ..

M IS A SCRATCH VARIABLE USED ONLY IN SHORT SEQUENCES.

IX IS A SCRATCH VARIABLE USED IN NO INNER PROCEDURE OR ON UNIT.

K AND L ARE SCRATCH VARIABLES, THEY ARE USED IN STRUCT, GTCK, AND GITM, BUT NOT IN COGO OR ERRMSG OR ANY ON UNIT.

1 ** SPECIFICATIONS FOR THE STRU/LADA FILES **

THE FOLLOWING 9 TYPES OF GROUPS OCCUR IN THE STRU FILE ..

- A AFTERPEICE TITLE GROUP, ALWAYS THE FIRST IN ITS SECTION
- P PERFORMANCE TITLE GROUP, ALWAYS THE FIRST GROUP IN ITS SECTION
- I INSTRUMENTAL TITLE, ALWAYS 1ST GRP IN SECTION.
- O OPERA TITLE , ALWAYS 1ST GRP IN SECTION.
- U MONOLOGUE WITH PARTS TITLE, ALWAYS 1ST GRP IN ITS SECTION.
- B BALLET TITLE, ALWAYS 1ST GRP IN ITS SECTION.
- E ENTERTAINMENT TITLE, ALWAYS 1ST GRP IN ITS SECTION.
- G 'ROLE-ACTOR GROUP', WITH TIME, ROLE, AND ACTOR ITEMS
- G LADDER GROUP, WITH AN L OR S ITEM.

ITEMS ARE TAGGED BY THE FOLLOWING CHARACTERS ..

- B PAGE ENTRY
- A ACTOR (ALSO MUSICIAN,DANCER, SINGER ETC.)
- R ROLE (ALSO SONG, ENTERTAINMENT, DANCE ETC.)
- T TIME ENTRY
- A (SMALL LETTER) AFTERPEICE TITLE
- P (SMALL LETTER) PLAY TITLE (MAINPEICE)
- U (SMALL LETTER) TITLE OF MONOLOGUE WITH PARTS
- B (SMALL LETTER) BALLET TITLE

T (SMALL LETTER) TRICK TITLE
I (SMALL LETTER) INSTRUMENTAL MUSIC TITLE
O (SMALL LETTER) OPERA TITLE
L LADDER REFERENCE OF THE 'AS' TYPE
D LADDER REFERENCE OF THE 'SEE' TYPE
C LADDER REFERENCE OF THE CGLE TYPE

ALL LADA FILE BLKS HAVE A TITLE OF AT LEAST THE NULL VARIETY. THE TITLE 'UN' IS TREATED JUST LIKE ANY OTHER TITLE. PERHAPS STRUCTUR SHOULD CONVERT 'UN' TITLES TO '' TITLES. STRU/STRS FILES DO NOT HAVE TITLE GRPS EXCEPT FOR TITLED SECTIONS.

THE TIME ENTRY, IF PRESENT, MUST PRECEDE ALL ROLE AND ACTOR ENTRIES IN THE GROUP.

ALL ROLE ENTRIES PRECEDE ANY ACTOR ENTRIES.

THE TITLE, IF ANY, OF AN AFTERPEICE, SONG, OR PERFORMANCE SECTION MUST BE THE FIRST GROUP IN ITS SECTION. THE TITLE ITEM MAY BE PRECEDED BY A PAGE ENTRY, HOWEVER, SINCE PAGE ENTRIES MAY OCCUR ANYWHERE.

A LADDER REFERENCE MAY NOT PRECEDE A PLAY, OPERA, OR AFTERPEICE TITLE.

A CGLE MUST BE PRECEDED BY AT LEAST ONE ROLE.

STRU FILE NOTES .. 9/12/71

1) EVERY TITLED SECTION HAS AT LEAST FC CAT SMOD AS 1ST 2 CHARS.
2) IF A SECTION HAS A TITLE THEN IT IS IN ITEM FORM, & THIS MAY BE OF LENGTH 0.
3) EVERY TITLED SECTION HAS AT LEAST A 0 LENGTH TITLE, (IF THERE ARE ANY CANDIDATES AT ALL AFTER THE THEATRE).
5) 'P GF' TYPES OF SECTIONS HAVE NO TITLE AT ALL, BUT THEATRE IS PRESERVED IN KEY.
6) AN RA GRP WITH NO ITEMS IS POSSIBLE DUE TO INPUT DATA ERROR, ELSE NOT.
7) NULL TIME ENTRIES CANNOT EXIST. THEY ALWAYS HAVE AT LEAST THE COLON.
8) NULL ROLE OR ACTORS (0 LENGTH) EXIST ONLY IF GTCK DELIVERS A 0 LENGTH C TO THE ROLE OR ACTOR LOOP.
(GITM ALWAYS DELIVERS AT LEAST 'ERRONEOUS ITEM'.)
9) LADDER REFERENCES HAVE ONLY S OR I OR C AS THEIR ITEM TYPE LETTERS, AND CAN OCCUR ONLY WHERE A ROLE OR AN ACTOR IS EXPECTED, THAT IS IN G GROUPS.
10) PAGE ITEMS ARE FIXED LENGTH (4 DATA, 6 INC ITEM CTRL) IF PRESENT.

1 FORMAT OF SECTION/BLK KEYS FOR STRUCTUR/LADDER/ITEMGET ..

RECORDED KEY LENGTH IS 12 BYTES.

BYTES 0 TO 2 ARE THE CODED DATE AND DATE SIGNIFICANCE BITS. THE RIGHTMOST 3 BITS DEFINE WHETHER THE CORRESPONDING PART OF THE DATE (YEAR, MONTH, AND DAY RESPECTIVELY) IS DEFINITE OR UNCERTAIN. A ZERO BIT MEANS DEFINITE, ONE MEANS UNCERTAIN. THE BITS CORRESPOND TO YEAR, MONTH, AND DAY RESPECTIVELY LEFT TO RIGHT.

THE ACTUAL DATE REPRESENTATION IS ENCODED BY THE FORMULA, IDATE=((IYEAR-1660)*372+(IMONTH-1)*31+IDAY-1)*8 ,. INTO THE BINARY FULLWORD, IDATE. AFTER THE SIGNIFICANCE BITS ARE ADDED TO IDATE, THE LAST 3 BYTES OF IDATE ARE CONCATENATED TO THE REST OF THE KEY. THE FULL RECORDED KEY IS WRITTEN FROM THE VARIABLE SKEY.

BYTE 3 CONTAINS THE SECTION TYPE LETTER (SMCD) ..

A) AFTERPEICE SECTION
B) BALLET SECTION
D) DANCING SECTION
E) ENTERTAINMENT SECTION
I) INSTRUMENTAL MUSIC SECTION
M) MUSIC SECTION
O) OPERA SECTION
P) PERFORMANCE SECTION
S) SINGING SECTION
T) TRICK SECTION
U) MONOLOGUE WITH PARTS SECTION

BYTES 4 TO 11 CONTAIN THE THEATRE ABBREVIATION, WHICH IS RIGHT-ADJUSTED AND PADDED ON THE RIGHT WITH BLANKS, IF NECESSARY.

```
/*
//LADRDOC  JOB 99999211,WCD,MSGLEVEL=(1,1)
```

```
//D EXEC LSPRINTS,OPT=CARDOCSN,MEM=DOCLADDR
//PRINT.IN DD DDNAME=INSYS
//PRINT.INSYS DD *
```

LADDER OVERVIEW

LADDER HAS THE TWO FUNCTIONS OF RETRIEVING LADDER REFERENCES AND OF PERFORMING COMPLEX UPDATES WITH THESE LADDER REFERENCES. THERE ARE 3 KINDS OF LADDER REFERENCES DISCUSSED SEPARATELY BELOW.

TITLE LADDER REFERENCES (TLE'S) MAY OCCUR AS THE FIRST LADDER REFERENCE IN ANY SECTION. TLE'S ARE IDENTIFIED BY A CAPITAL 'L' AS THEIR ITEM TYPE CHARACTER. THE REFERENCE MUST BE THE FIRST ITEM OF THE FIRST CAST GRP FOLLOWING THE TITLE OF THE SECTION. WHEN A TLE REFERENCE IS ENCOUNTERED DURING INPUT/SETUP IT IS SEARCHED FOR AND, IF FOUND AND SUCCESSFULLY MATCHED ON THE TITLE ITEM, IS PLACED BEHIND THE REFERRING BLK IN THE S BUFFER. INPUT/SETUP OF THE REFERRING BLK THEN CONTINUES, POSSIBLY INVOLVING CGLE'S. WHEN REFERRING BLK'S INPUT/SETUP IS FINISHED THEN THE TLE IS SET UP ON A SEPARATE CHAIN CALLED THE OUTPUT CHAIN, POINTED TO BY IGP3 AND IGP4.

AFTER ALL (ALL) INPUT/SETUP IS COMPLETE THE UPDATE STAGE STARTS. IN THIS STAGE LADDR GOES THROUGH THE INPUT CHAIN, AND HANDLES EACH GRP IN IT ON AN INDIVIDUAL BASIS. FOR CORRECT TLE CASES AND NON-LADDER CASES PROCESSING IS AS DESCRIBED IN THE SPECIFICATIONS, 'TYPING INSTRUCTIONS FOR THE LINDCN STAGE PROJECT', 'DESCRIPTION OF THE CALENDER', 5 MAY OCRR II. THIS IS CARRIED OUT BY RECHAINING GRPS AND ITEMS FROM THE INPUT CHAIN TO THE OUTPUT CHAIN, AND DELETING GRPS AND ITEMS FROM THE OUTPUT CHAIN. TLE PROCESSING OR SLE PROCESSING IS CARRIED ON SIMULTANEOUSLY WITH CGLE PROCESSING ON A GRP BY GRP BASIS. ALTHOUGH THE UPDATE STAGE HAS NOT BEEN THOUROUGHLY TESTED IT SHOULD WORK FOR CHINA DATA MATERIAL SINCE THIS PRESUMABLY DOES NOT HAVE ANY UPDATES EXCEPT THE REPLACE VARIETY WHICH HAS BEEN THOROUGHLY TESTED. ONE PROBLEM WHOSE SOLUTION HAS NOT BEEN UNDERTAKEN IS THAT OF REKEYING IN THE OUTPUT CHAIN. USING THE FACT THAT NAMES ARE SUPPOSED TO BE UNIQUE IN ANY ONE CONTEXT, IN THIS CASE A SECTION WITH ALL ITS LADDER REFERENCES, SIMPLIFIES THE PROGRAMMING QUITE A BIT. THE SIMPLIFICATION LIES IN THAT UPDATING AN ITEM INVOLVES ONLY ALTERING, IN SOME WAY, THE OUTPUT CHAIN. THE REFERRING ITEM IN THE INPUT CHAIN IS THEREAFTER 'UNREFERENCEABLE', BUT THE ITEM IN THE OUTPUT CHAIN CAN STILL BE AFFECTED BY A LATER ITEM IN THE INPUT CHAIN. FOR EXAMPLE IF THERE IS AN ACTOR, JOE, AT THE FAR END OF THE OUTPUT CHAIN, AND THERE ARE 2 REFERENCES TO IN IN THE INPUT, THE FIRST ONE BEING AN ADDITION OF SOME SORT, AND THE SECOND BEING A DELETION, THEN THE ORIGINAL JOE AT THE FAR END OF THE OUTPUT CHAIN WOULD SURVIVE AND THE JOE JUST ADDED WOULD BE DELETED. DELETIONS SHOULD PRECEDE ADDITIONS, HOWEVER EVEN THIS WILL NOT NECESSARILY PREVENT ALL TRAGEDIES.

'SEE' LADDER REFERENCES ARE PROCESSED MUCH LIKE TLE'S IN INPUT/SETUP. IN THE UPDATE STAGE, HOWEVER, THE PROCESSING IS AS DESCRIBED IN THE JUNE 9 GENERAL SPECIFICATIONS. THIS HAPPENS ON AN INPUT GRP BY INPUT GRP BASIS, AND PRECEDES REGULAR PROCESSING AND RECHAINING TO THE OUTPUT CHAIN FOR THE GRP. A CGLE GRP WILL HAVE SEE TYPE PROCESSING DONE ON IT WITH RESPECT TO THE REFERRED-TO GRP BEFORE OTHER PROCESSING IS DONE. THIS BUG SHOULD NOT CAUSE MUCH TROUBLE, ESPECIALLY AT PRESENT, SINCE THE MATERIAL APPEARS TO HAVE SO FEW 'SEE' REFERENCES (ONLY 1 IN BATCH 1).

TIME ENTRIES ARE NOT HANDLED IN 'SEE' REFERENCES. THEY ARE TAKEN FROM THE CURRENT TIME ENTRY, IF ANY, REGARDLESS.

IN THE OUTPUT FORMATTING STAGE ACTORS THAT WERE READ IN BY A 'SEE' LADDER REFERENCE

AND NOT DELETED ARE FORMATTED OUT WITH A QUESTION MARK FOLLOWING THEM. AT PRESENT THIS IS TRUE OF CGLE ACTORS AS WELL. THIS IS DONE IN ORDER TO LOCATE CGLE'S MORE EASILY ON THE PRINTOUT SINCE CGLE'S HAVE NOT BEEN TESTED THOROUGHLY, AND

SUCCESSFUL ONES ARE SCARCE ON THE ERROR PRINTOUTS.

AFTER OUTPUT CHAIN SETUP, ANY CGLE'S LEFT ARE PUT THROUGH INPUT/SETUP PROCESSING AND CHAINED TO THE CAST GRPS THAT REFERRED TO THEM. THE LINK IS IN THE REFERRING GRP'S GCB IN THE GCG ARRAY, AND POINTS TO THE REFERRED-TO GRP'S GCB.

IN INPUT/SETUP REFERRED-TO CGLE GRPS ARE READ IN AND APPENDED TO THE S BUFFER AS THEIR REFERRING ITEMS ARE ENCOUNTERED. AFTER THIS THE GCG ELEMENT OF THE REFERRING GRP'S GCB IS SET TO CONTAIN THE LAST LOC OF THE REFERRED-TO GRP IN THE S BUFFER. AFTER THE TLE OR SLE SETUP STAGE, IF ANY, EACH GCB IN THE INPUT CHAIN IS EXAMINED TO SEE IF ITS GCG ELEMENT POINTS TO A REFERRED-TO CAST GRP IN THE S BUFFER. IF SO , THIS REFERRED-TO GRP IS ALSO SET UP, WITH THE GCG PTR IN THE REFERRING GRP NOW RESET TO POINT TO THE REFERRED-TO GRP'S GCB.

WHEN UPDATE PROCESSING IS DONE ON AN INPUT CHAIN GRP, IT IS AGAIN CHECKED TO SEE IF IT HAS A CGLE. IF SO, THEN THE REFERRED-TO GRP IS LOGICALLY SUBSTITUTED FOR THE OUTPUT CHAIN TEMPORARILY, AND THEN NORMAL UPDATE PROCESSING IS DONE ON IT. AFTER THE GRP IS PROCESSED, THE RESULT IS CHAINED TO THE ACTUAL OUTPUT CHAIN.

IF BLKSIZE PROBLEMS OCCUR FOR LADA IT WILL BE NECESSARY TO CHANGE BOTH THE DCB ON THE DD CARD AND THE S & Z BUFFERS IN LADDR. THIS, AS FAR AS I CAN SEE, IS AS SIMPLE AS IT SOUNDS, EXCEPT THAT CORE PROBLEMS MAY ARISE. THIS IS NEITHER A HIGH NOR A LOW PROBABILITY.

LADDR'S OPTIONS ARE IN COLS 1-32, AND THE OPTION CHAIN IS IN COLS 33-38. THE REMAINDER OF EACH CARD IS IGNORED.

RECOMMENDED OPTIONS FOR A FIRST RUN ARE 2, 6, 7, AND 27.

GENERAL DESCRIPTION OF FILE USAGE

DDNAME	USE
SYSIN	ALWAYS USED FOR OPTION CHAIN.
SYSPRINT	ALWAYS USED FOR PRINT FILE.
LADA	ALWAYS USED , INTERMEDIATE STORAGE, DA KEYED OUTPUT.
STRU	USED FOR DA KEYED INPUT WHEN OPTION 6 IS 0.
STRS	USED FOR SEQL UNKEYED INPUT WHEN OPTION 6 IS 1.
LADS	USED ONLY FOR SEQL UNKEYED OUTPUT WHEN OPTION 7 IS 1.

LADA IS USED FOR INTERMEDIATE STORAGE OF A SEASON'S PERFORMANCES, AND, IF OPTION 2 IS NOT SPECIFIED OR THE OUTPUT WANTED IS ONLY IN THE LAST SEASON PROCESSED, THEN LADA MAY ALSO SERVE AS THE OUTPUT FILE. THIS WOULD NECESSITATE CHANGING THE DISP PARAMETER TO 'KEEP' FOR THE LADA DD STATEMENT.

STRU IS THE MAIN (STRUCTURED DATA) INPUT FILE IF OPTION 6 IS NOT SPECIFIED, OTHERWISE STRS IS. IF USE OF STRU, LADS, OR STRS IS NOT SPECIFIED ON ANY OPTION CARD THEN ITS DD CARD NEED NOT BE PRESENT IN THE JCL DECK.

LADDER'S DATA OUTPUT IS IN THE SAME FORMAT AS THE STRU OR STRS FILE INPUT EXCEPT THAT LADDER ENTRIES AND PAGE ENTRIES ARE NO LONGER PRESENT. IN THE LADA FILE THE BLKS ARE LIKE THE KEYED STRU FILE BLKS. IN THE LADS FILE BLKS ARE UNKEYED, WITH THE KEY AND 2 BLANKS APPENDED TO THE FRONT OF THE BLK JUST AS WITH THE STRS FILE INPUT.

INTERNAL PROGRAMMING NOTES

****

FOR LADDER K AND L ARE SCRATCH VARIABLES USED ONLY IN SHORT SEQUENCES OF INSTRUCTIONS.

IGP1=0 IF NO PRE-LADDER GRPS, =1 IF THERE IS AT LEAST ONE PRE-LADDER GROUP.

IGP2 PTS TO LAST PRE-LADDER GRP, =0 IF NO PRE-LADDER GRPS.

IGP3 PTS TO 1ST LADDER GRP, =0 IF NO LADDER GRPS.

IGP4 PTS TO LAST LADDER GRP, =0 ONLY IF NO RA GRPS AT ALL EXIST.

THE GCB (GROUP CONTROL BLOCK) LINK FORWARD VARIABLE IS GF.

THE ICB (ITEM CONTROL BLOCK) LINK FORWARD FIELD IS ILK.

THE LENGTH OF EACH ACTOR OR ROLE CHAIN IS KEPT IN THE CHAIN'S GCB.

THE KEYS SUBROUTINE SEARCHES THE OUTPUT CHAIN FOR AN ITEM COREESPONDING TO ONE DESCRIBED IN THE CALL. THE CALLING SEQUENCE IS ..

1ST PARAMETER GRP IN WHICH KEY RESIDES (GCB NUMBER)
2ND PARAMETER OFFSET IN THE SUBGROUP OF THIS GROUP.
0= 1ST ON CHAIN AS DOES 1.

AN IMPLICIT PARAMETER IS THE ARRAY, SBITS, DEFINED ON SBITE. IN THIS ARRAY IF THE CORRESPONDING BIT IS A 1 THEN KEYING IS ATTEMPTED ON THAT PARTICULAR KIND OF ITEM. IF MORE THAN ONE OF THE BITS IS 1 THEN THE ORDER OF SEARCHING IS .. ROLE THEN ACTOR, THEN TIME (SECOND BIT, THIRD BIT, THEN FIRST BIT).

THE VALUES RETURNED ARE IN IRG, IROF, AND RBITS RESPECTIVELY. IRG IS 0 IF THE SEARCH WAS UNSUCCESSFUL. OTHERWISE IRG IS THE NUMBER OF THE GCB FOR THE GRP WHICH CONTAINS THE KEY, AND IRCF IS THE ITEM CONTROL BLOCK NUMBER (NOT OFFSET WITHIN SUBGRP, AS IN CALLING SEQUENCE, BUT ABSOLUTE). RBITS INDICATES WHICH KIND OF ITEM THE KEYING WAS ACHEIVED ON, BEING INTERPRETED THE SAME AS SBITS.

IN ADDITION, IGL INDICATES THE PREVIOUS GRP ON THE LADDER CHAIN IF ANY (=0 IF 1ST GRP), AND IIL INDICATES THE ABSOLUTE ITEM CONTROL BLOCK OF THE PRECEDING ITEM IN THE SUBGROUP, IF ANY (=0 IF NONE).

OTHER MORE EXTENSIVE, BUT LESS REIIABLE NOTES CAN BE FOUND IN THE FILING CABINET IN THE LAWRENCE ROOM.

```
/*
//ITEMREF  JOB 99999211,WCD,MSGLEVEL=(1,1)
//D EXEC LSPRINTS,OPT=CARDOCSN,MEM=DOCITEMS
//PRINT.IN DD DDNAME=INSYS
//PRINT.INSYS DD *
```

CONTROL OF THE SELECTION PROCESS AT ANY ONE POINT IN THE RUN LIES IN THE SELECTION CONTROL BLOCK (SCB), WHICH CONSISTS OF ONE DATE RANGE ENTRY AND 0 OR MORE SELECTION STATEMENT ENTRIES. A SELECTION STATEMENT ENTRY CONTAINS THE FOLLOWING ITEMS OF INFORMATION ..

A)	INCLUSIVENESS	(A DIGIT FROM 1 TO 4)
B)	SECTION TYPE	S
C)	THEATRE	M
D)	TIME	C
E)	TITLE	T
F)	SYNTACTIC ROLE	R
G)	SYNTICTIC ACTOR	A

EACH ITEM, EXCEPT INCLUSIVENESS IS A VARYING LENGTH CHARACTER STRING. MATCHING OF A SORT RECORD WITH ANY SELECTION ENTRY IN THE PROPER DATE RANGE CAUSES THE RECORD AND POSSIBLY THE SORT RECORDS OF THE GROUP, SECTION, OR PERFORMANCE IN WHICH IT IS FOUND TO BE PUT INTO THE OUTPUT FILE. WHETHER ONLY THE MATCHING RECORD ITSELF OR THE ENTIRE GROUP, SECTION, OR PERFORMANCE IS PUT OUT DEPENDS ON THE INCLUSIVENESS ITEM OF THE SELECTION ENTRY ON WHICH THE MATCH WAS MADE, 1=RECORD, 2=GROUP, 3=SECTION, 4=PERFORMANCE.

THE LETTER FOLLOWING EACH TYPE OF ITEM IN THE ABOVE TABLE IS THE ITEM TAG USED IN INPUT CONTROL STATEMENTS TO SPECIFY THE TYPE OF EACH ITEM.

SELECTION BY A SELECTION ENTRY IS DETERMINED IN THE FOLLOWING WAY .. FIRST THE DATE OF THE SORT RECORD MUST BE IN THE DATE RANGE OF THE SCB. SECOND, EACH ITEM IN THE SELECTION ENTRY OTHER THAN INCLUSIVENESS, THAT HAS A NONZERO LENGTH MUST MATCH THE CORRESPONDING ITEM IN THE SORT RECORD.

THE SELECTION ENTRIES WILL BE DEFINED BY CONTROL RECORDS IN THE FORM OF SELECTION CONTROL STATEMENTS IN THE FILE SYSIN. SINCE IT IS EXPECTED THAT THIS FILE WILL CONSIST OF CARDS, EACH LETTER READ IN THE DATA PORTION OF A SELECTION ENTRY ITEM WILL BE TRANSLATED TO A LOWER CASE LETTER IF IT IS ENTERED IN UPPER CASE UNLESS THE LETTER IS PRECEDED BY A SLASH, IN WHICH CASE THE SLASH WILL BE LEFT OUT AND THE CHARACTER FOLLOWING IT WILL NOT BE TRANSLATED. A CHARACTER IMMEDIATELY FOLLOWING AN INITAIL SLASH, INCLUDING ANOTHER SLASH OR A BLANK, IS TAKEN AS IS AND

IS COUNTED AS A NONBLANK CHARACTER FOR PURPOSES OF DELIMITING THE STATEMENT.

A SELECTION STATEMENT IS ONE THAT HAS AN INCLUSIVENESS TAG OF 1, 2, 3, 4, OR 6. AN ORDINARY SELECTION STATEMENT IS ONE THAT HAS AN INCLUSIVENESS TAG OF 1, 2, 3, OR 4.

EACH STATEMENT CONSISTS OF THE INCLUSIVENESS ITEM PLUS 0 OR MORE OTHER ITEMS, THE STATEMENT ENDING WITH 2 CONSECUTIVE BLANKS. EACH ITEM WILL START WITH A CHARACTER INDICATING WHAT SORT OF ITEM IT IS, TITLE, THEATRE, ROLE, ETC., FOLLOWED BY THE ITEM ITSELF, AND, EXCEPT FOR THE LAST ITEM IN A STATEMENT, ENDING WITH THE '&' SIGN. THE LAST ITEM WILL BE DELIMITED BY THE 2 CONSECUTIVE BLANKS THAT END THE STATEMENT. AN ENTIRE GROUP OF STATEMENTS THAT FORM AN SCB WILL BE DELIMITED BY A STATEMENT WITH AN INCLUSIVENESS TAG OF 7.

FOLLOWING THIS STATEMENT PROCESSING OF SORT RECORDS WILL PROCEED UNTIL A DATE GREATER THAN THE 2ND DATE IN THE DATE RANGE ENTRY OF THE SCB IS ENCOUNTERED. THUS SELECTION SPECFICATIONS SHOULD BE SET UP WITH DATE RANGES IN CHRONOLOGICAL ORDER.

EACH ORDINARY SELECTION STATEMENT CORRESPONDS TO ONE SCB ENTRY.

EACH REPEAT STATEMENT CORRESPONDS TO AS MANY SELECTION ENTRIES AS ARE DEFINED BY THE REPITITIVE SPECFICATIONS.

AN ORDINARY SE WILL BE FORMED FROM THE ITEMS IN THE SELECTION STATEMENT AS WELL AS THE LAST ITEM OF EACH OTHER TYPE USED IN A PREVIOUS SELECTION STATEMENT UNLESS SPECIFIED OTHERWISE. AN EXAMPLE SHOULD ILLUSTRATE THIS ..

```
5 1738 05 23 1738 06 14
1MDL&T/HAMLET, /PRINCE OF /DENMARK
4MCG
```

THE STATEMENT ON THE FIRST LINE, LIKE ALL STATEMENTS, STARTS WITH THE STATEMENT TAG, IN THIS CASE 5, WHICH INDICATES THAT IT IS A DATE RANGE STATEMENT. THE DATE RANGE ITSELF IS IN A FIXED LENGTH FORMAT AND HENCE NEEDS NO DELIMITERS AT IT ITS END.

THE START OF THE SECOND STATEMENT IS INDICATED BY THE FIRST NONBLANK CHARACTER FOLLOWING THE END OF THE DATE RANGE STATEMENT. THE 1 IS AN INCLUSIVENESS CHARACTER INDICATING THAT IF A MATCH IS FOUND ON THE SELECTION ENTRY RESULTING FROM THIS STATEMENT THEN ONLY THE SORT RECORD ON WHICH THE MATCH IS MADE NEED BE PUT OUT AS A RESULT OF THE MATCH. THE M FOLLOWING THE INCLUSIVENESS TAG INDICATES THAT THE ITEM FOLLOWING IS A THEATRE (THINK MOVIE). THE DL IS THE ITEM. THE '&' INDICATES THE END OF THE ITEM. CONTINUING ANALOGOUSLY THE 'T' INDICATES A TITLE, THE DOUBLE BLANKS FOLLOWING '/HAMLET, /PRINCE OF /DENMARK' INDICATE BOTH THE END OF '/HAMLET, /PRINCE OF /DENMARK' AND THE END OF THE STATEMENT.

AS A RESULT OF THE FIRST 2 STATEMENTS ANY SORT RECORD WITH A THEATRE ENTRY OF 'DL' AND A TITLE ENTRY OF'/HAMLET, /PRINCE OF /DENMARK' BETWEEN THE DATES OF MAY 23 1738 AND JUNE 14 1738 WILL BE PUT OUT. IN ADDITION THE STATEMENT INTERPRETER HAS REMEMBERED THE TWO ITEMS, THEATRE, AND TITLE, IN CASE OF LATER NEED. THIS USE IS ILLUSTRATED BY THE SELECTION ENTRY RESULTING FROM THE THIRD STATEMENT. THIS STATEMENT HAS ONLY ONE ITEM, A 'CG' THEATRE ENTRY, BUT SINCE IT HAS NOT BEEN TOLD OTHERWISE, THE STATEMENT INTERPRETER INTERPRETS THIS TO MEAN THAT THE OTHER KINDS OF ITEMS THAT HAVE BEEN DEFINED PREVIOUSLY, IN THIS CASE ONLY THE TITLE, '/HAMLET, /PRINCE OF /DENMARK', ARE TO REMAIN THE SAME. THAT IS, THE ENTRY CAUSES NOT ONLY MATCHING ON 'CG' BUT ON '/HAMLET, /PRINCE OF /DENMARK' AS WELL, EVEN THOUGH THE THIRD STATEMENT DOES NOT MENTION '/HAMLET, /PRINCE OF /DENAMRK'. THE NUMBER 4, THE INCLUSIVENESS INDICATOR FOR THE THIRD STATEMENT INDICATES THAT IF A MATCH IS FOUND FOR THIS SELECTION ENTRY THEN THE ENTIRE SET OF SORT RECORDS RESULTING FROM THAT PERFORMANCE ARE TO BE PUT OUT ,. 3 WOULD MEAN THE ANALOGOUS THING FOR THE SECTION, AND 2 FOR THE GROUP. THE WAY TO 'CLEAR' THE INTERPRETER'S MEMORY OF A PREVIOUSLY DEFINED ITEM IS TO REDEFINE IT (THE ITEM) AS A ZERO LENGTH CHARACTER STRING, FOR EXAMPLE ..

```
1T
```

WOULD CLEAR THE TITLE AS THOUGH IT HAD NOT BEEN MENTIONED PREVIOUSLY.

IT SHOULD BE REMEMBERED, HOWEVER, THAT THE ABOVE STATEMENT WOULD

RESULT IN A SELECTION ENTRY. CLEARING WORKS JUST AS WELL IN A STATEMENT THAT HAS OTHER ITEMS IN IT ..

1T&MHAY

WOULD CLEAR THE TITLE, AND PROVIDE FOR SELECTION OF SECTIONS OCCURRING AT 'HAY' ON THE SAME BASIS AS THE PREVIOUS DEFAULTS EXCEPT THAT ANY TITLE WOULD BE ACCEPTABLE.

A STATEMENT BEGINNING WITH AN INCLUSIVENESS TAG OF '6' IS A REPEAT STATEMENT. THE REPEAT STATEMENT MAY CONTAIN ANYTHING THAT AN ORDINARY SELECTION ENTRY STATEMENT HAS, BUT IN ADDITION IT CONTAINS, IMMEDIATELY FOLLOWING THE INCLUSIVENESS TAG, 2 NUMBERS THAT FORM ITS REPETITIVE SPECIFICATION. THESE NUMBERS DEFINE RESPECTIVELY THE FIRST AND LAST PREVIOUSLY DEFINED SELECTION ENTRIES THAT ARE TO BE USED IN THE REPETTION. THESE DEFINED SELECTION ENTRIES ARE REPEATED EXCEPT FOR CHANGES DEFINED IN THE USUAL WAY IN THE REMAINDER OF THE REPETITIVE STATEMENT. EXAMPLE, SUPPOSE THIS WERE THE 4TH STATEMENT.

6 1 2 AJONES&RHAMLET

TWO NEW SELECTION ENTRIES WOULD BE FORMED, THE FIRST REQUIRING DL, '/HAMLET, /PRINCE OF /DENMARK', JONES, AND HAMLET ,. THE SECOND REQUIRING CG, '/HAMLET, /PRINCE OF /DENMARK', JONES, AND HAMLET. THE INCLUSIVENESS TAG OF EACH SELECTION ENTRY THUS FORMED WILL BE THE SAME AS THAT OF THE ORGINAL ENTRY UPON WHICH IT WAS BASED.

THE MEMORY OF THE STATEMENT INTERPRETER IS CLEARED BY A REPEAT STATEMENT EXCEPT FOR THE ITEMS USED IN THE STATEMENT. THE CLEARING IS DONE BEFORE THE ITEMS IN THE STATEMENT ARE PROCESSED, THUS NO DEFAULTS ARE USED.

REPEAT STATEMENTS AND DATE RANGE STATEMENTS MUST BEGIN IN COLUMN 1 OF A CARD OR THE EQUIVALENT.

ITEMGET REFERENCE SHEET

INCLUSIVENESS AND STATEMENT TYPE NUMBERS ..

0 CLEAR ALL DEFAULTS FOR SCB
1 SELECT RECORDS
2 SELECT GROUPS
3 SELECT SECTIONS
4 SELECT PERFORMANCES
5 SET UP DATE RANGE STATEMENT
6 START OF REPEAT TYPE STATEMENT
7 INITIATE SELECTION STAGE, END OF SELECTION SETUP STAGE.
8 CLEAR ALL DEFAULTS AND SCB, EXCEPT FOR DATE RANGE.
9 STOP EXECUTION IMMEDIATELY.

ITEM TYPE CHARACTERS ..

S SECTION LETTER
M THEATRE (MOVIE)
C TIME ENTRY (CLOCK)
T TITLE
R ROLE
A ACTOR

ITEMS ARE DELIMITED BY THE '&' IF FOLLOWED BY ANOTHER ITEM IN THE SAME SE, OTHERWISE THEY ARE DELIMITED BY THE DOUBLE BLANK THAT DELIMITS THE END OF THE STATEMENT. STATEMENTS ARE DELIMITED EXCLUSIVELY BY THE DOUBLE BLANK. SCB'S ARE DELIMITED BY A TYPE 7 STATEMENT.

FORMAT FOR A REPEAT SPECIFICATION IS ..

COL 1	'6'
COL 2	IGNORED
COL 3-6	FIRST SE, RIGHT JUSTIFIED
COL 7-10	2ND SE, RIGHT JUSTIFIED
CO. 11-13	IGNORED
COL 14	ITEM TYPE CHAR FOR 1ST ITEM

FORMAT FOR DATE RANGE STATEMENT ..

```
COL 1       '5'
COL 2       IGNORED
COL 3-7     EARLIER YEAR
COL 8       IGNORED
COL 9-10    EARLIER MONTH NUMBER
COL 11      IGNORED
COL 12-13   EARLIER DAY OF MONTH
COL 14      IGNORED
COL 15-18   LATER YEAR
COL 19      IGNORED
COL 20-21   LATER MONTH NUMBER
COL 22      IGNORED
COL 23-24   LATER DAY OF MONTH
COL 25      SEARCH RESUMES FOR NEXT STATEMENT.

     IN ORDER TO SAVE CORE FOR MORE SE'S THE ITEMGET PROGRAM
(NAME = CTRL) IS CONSTRUCTED AS AN OVERLAY OF 3 PROCEDURES, THE
ROOT, THE CONTROL CARD INTERPRETER, AND THE SELECTOR.
     CARD INPUT (SYSIN FILE) IS TO THE ROOT SECTION FOR THE
OPTION CARD WHICH HAS 80 BINARY OPTION LOCATIONS.  ALL OTHER
SYSIN INPUT IS TO THE CONTROL CARD INTERPRETER.  THIS INTERPRETER
CONSIDERS ALL INPUT TO BE IN THE FORM OF STATEMENTS.  OFFICIALLY
A STATEMENT STARTS WITH A DIGIT FROM 0 TO 8, AND ENDS WITH 2 OR
MORE CONSECUTIVE BLANKS.  ACTUALLY THE INTERPRETER MAY BE SOMEWHAT
MORE LENIENT, BUT DON'T COUNT ON IT.
/*
//TRTDOC   JOB 99999211,WCD,MSGLEVEL=(0,0)
//LSPRINTS PROC LIB=SRCLIB,MEM=ERROR,OUTVOL=GWSJR1,DSSP=MOD,SSP=300,
//             OPT=DIRECTSN,CHAIN=SN
//PRINT  EXEC PGM=LSPRINT
//STEPLIB  DD UNIT=2311,VOL=SER=GWSJR1,DSN=UTLIB,DISP=OLD
//SYSPRINT DD SYSOUT=A,UCS=(&CHAIN)
//BAKUP    DD UNIT=2311,VOL=SER=&OUTVOL,DSN=&LIB.(&MEM),
//            DISP=(&DSSP,KEEP),SPACE=(TRK,(&SSP,50,10)),
//            DCB=(RECFM=FB,BLKSIZE=3600,LRECL=80)
//SYSIN    DD UNIT=2311,VOL=SER=GWSJR1,DSN=SYS1.PROCLIB(&OPT),
//            DISP=OLD
// PEND
//LSPRINT EXEC LSPRINTS,MEM=TRTDOC
//PRINT.IN DD DATA
                              *** TRT PROGRAM **
     THIS PROGRAM IS USED TO PRODUCE & DOCUMENT TRANSLATION
TABLES.  IT HAS 2 PARAMETERS, BOTH SPECIFIED ON THE EXEC CARD.
THESE ARE MAP= & INCNT=.
     MAP= SPECIFIES HOW MANY MAPS OF EACH TRANSLATION TABLE ARE
TO BE PRINTED.  NUMBERS LESS THAN 1 WILL PRODUCE NO MAPS.  INCNT=
SPECIFIES HOW MANY TRANSLATION TABLES ARE TO BE EXPECTED IN
THE INPUT.  AGAIN NUMBERS LESS THAN 1 WILL PRODUCE NO TRANSLATE
TABLES.
     TRT CHECKS FOR 2 ERRORS ..
1)   ILLEGAL CHARACTERS.  ANYTHING BESIDES COMMA, BLANK, HEX
     DIGIT, OR SEMICOLON UNLESS IN A COMMENT.
2)   OVERFLOW DUE TO TOO MANY QUOTES.  THIS HAPPENS WHEN THE SIZE
     OF THE OUTPUT ON ONE CARD BECOMES TOO BIG DUE TO THE LARGE
     NUMBER OF APOSTROPHE CHARACTERS (QUOTES).  EACH QUOTE IS
     PUNCHED AS 2 QUOTES ON THE CARD, EXCEPT THE FIRST AND LAST
     QUOTES, WHICH ARE DELIMITERS.

     ALTHOUGH CERTAIN OTHER FORMS OF INPUT ARE ACCEPTABLE, TRT
IS DESIGNED TO ACCEPT INPUT AS A LIST OF HEX NUMBERS IN PL/1
LIST DIRECTED FORMAT.  INPUT OF A TABLE MAY BE DELIMITED BY A SEMICOLON.
IF LESS THAN 256 CHARACTERS ARE SPECIFIED FOR A TABLE THEN THE
REMAINDER OF THE 256 BYTE TABLE PRODUCED IS UNDEFINED. THE INPUT
ROUTINE IS DESIGNED TO ACCEPT COLS 3-80  OF A CARD AS A COMMENT
IF THERE IS AN ASTERISK IN COL 3.  THERE MAY BE UP TO 4 OF THESE
COMMENT CARDS INTERSPERSED THROUGH THE TEXT.  THEY ARE PRINTED
```

```
C9,C9,40,40,40,40,40,40,40,40,C9,C9,C9,60,C9,C9,
C9,C9,C9,C9,C9,C9,C9,C9,C9,C9,C9,C9,C9,60,C9,C9,
C9,4B,4B,4B,4B,4B,4B,4B,4B,4B,C9,C9,C9,C9,C9,C9,
C9,4B,4B,4B,4B,4B,4B,4B,4B,4B,C9,C9,C9,C9,C9,C9,
C9,C9,4B,4B,4B,4B,4B,4B,4B,4B,C9,C9,C9,C9,C9,C9,
40,40,40,40,40,40,40,40,40,40,4E,C9,4E,C9,C9,C9,
/*
//CARDPRNT JOB 99999211,WCD,MSGLEVEL=1
//D EXEC LSPRINTS,OPT=CARDOCSN,MEM=DOCMAIL
//PRINT.IN DD DDNAME=INSYS
//PRINT.INSYS DD *
                LSP MAILING LIST PROGRAM
                ************************
   INPUT TO THIS PROGRAM IS IN THE FORM OF 80 BYTE RECORDS
(CARDS) THAT ARE DIVIDED INTO TWO 40 BYTE FIELDS, CORRESPONDING TO
ONE LINE EACH OF AN ADDRESS.  THE MAXIMUM NUMBER OF CARDS FOR
ONE ADDRESS IS 3, CONSTITUTING A LIMIT OF 6 LINES FOR AN ADDRESS.
   THE DELIMITER FOR SEPARATING ONE ADDRESS FROM ANOTHER IS THE '*'
WHICH APPEARS SOMEWHERE IN THE LAST CARD OF EACH ADDRESS.
   THE LAST ADDRESS MUST BE FOLLOWED BY A CARD HAVING AN ASTERISK
IN COLUMN ONE.
   THREE MODIFICATIONS ARE PERFORMED ON THE ADDRESS LIST BEFORE IT IS
PRINTED ..
1)  THE DELIMITING ASTERISK IS REMOVED.
2)  THE ADDRESSES ARE SORTED ON THE 40 BYTES OF THE FIRST LINE.
3)  IF THERE IS A '/' IN THE FIRST FIELD OF ANY ADDRESS THEN
    THE CHARACTERS PRECEDING THE SLASH ARE PLACED AFTER ALL THE
    OTHER NONBLANK CHARACTERS FOLLOWING THE '/' IN THE
    SAME FIELD, SEPARATED FROM
    THE NONBLANK CHARACTERS BY ONE BLANK.  ALSO THE SLASH IS
    REMOVED AND THE RESULTING CHARACTER STRING IS PLACED LEFT-ADJUSTED
    ON THE FIRST PRINTED LINE OF THE ADDRESS.
/*
//DOCFRMAT JOB 99999211,WCD
//D EXEC LSPRINTS,OPT=CARDOCSN,MEM=DOCFRMAT
//PRINT.IN DD DDNAME=INSYS
//PRINT.INSYS DD *
                        *** THE FORMAT PROGRAM ***
    THIS PROGRAM, CALLED FRMAT, IS USED TO PRINT SORT RECORDS.
IT CAN PRINT THE OUTPUT OF EITHER THE SORT/MERGE PROCESS OR
THE ITEMS STEP.  THE PRINTOUT IS FORMATTED SO AS TO PRINT MOST
SORT RECORDS ON A SINGLE LINE.  IF, HOWEVER, AN ITEM IS TOO LONG
FOR ITS FIELD O THE LINE THEN THE REMAINDER OF THE ITEMS FOLLOWING
IT WILL STILL BE PRINTED IN THEIR PROPER COLUMNS, BUT ON THE
NEXT LINE.  IF A SYNTACTIC ACTOR OVERFLOWS ITS FIELD THEN THE
LAST  PART OF THE ACTOR IS PRINTED AT THE BEGINNING OF THE NEXT
LINE.  MOST SUCH SYNTACTIC ACTORS ARE IN ERROR.
    FRMAT HAS 2 INPUT FILES, SRTD & SYSIN.  SRTD IS FOR SORT
RECORD INPUT.  SYSIN IS FOR THE OPTION CARD.  THERE ARE 32 OPTION
LOCCTIONS IN COLS 1-32.  ONLY OPTION 1 IS MEANINGFUL.  IT SHOULD
BE A 0, EXCEPT FOR PROGRAM DEBUGGING.  THE OTHER 31 OPTION
LOCATIONS MUST CONTAIN EITHER 0'S OR 1'S.  THE OUTPUT FILE
IS SYSPRINT.  IT IS FIXED LENGTH AND UNBLOCKED.  BLKSIZE IS 133.
/*
//DOCSAVED JOB 99999211,WCD
//D EXEC LSPRINTS,OPT=CARDOCSN,MEM=DOCSAVED
//PRINT.IN DD DDNAME=INSYS
//PRINT.INSYS DD *
                        *** SAVEDT PROGRAM ***
    THIS PROGRAM CAN BE USED FOR MTST INPUT.  MTST MATERIAL
IS NO LONGER BEING PRODUCED, SO THIS PROGRAM IS UNLIKELY TO
BE USED ANY MORE, AND IS THUS NOT VERY
BE USED ANY MORE, AND IS THUS NOT DOCUMENTED VERY WELL.  ITS
OUTPUT CAN BE USED BY THE STRUCTUR PROGRAM IF 2 THINGS ARE
DONE ..  1)  EITHER REMOVE ALL MTST TAPE ENTRIES OR PUT IN THE
MTST TAPE ENTRY PROCESSING STATEMENTS MENTIONED IN THE STRUCTUR
```

```
   LISTING AND RECOMPILE (IN CHICH CASE SCANNER T SECTIONS WILL
   NOT WORK TOO WELL).  2)  SET OPTION 5 IN STRCT TO 0.
/*
//EDTDOC   JOB 99999211,WCD,MSGLEVEL=(1,1)
//D EXEC LSPRINTS,OPT=CARDOCSN,MEM=DOCEDT
//PRINT.IN DD DDNAME=INSYS
//PRINT.INSYS DD *
      EDT IS AN INTERACTIVE EDITING PROGRAM DESIGNED TO MANIPULATE
   MTST ORIGINAL DATA FILES.  IT IS NOT WELL DOCUMENTED, COMMENTED,
   OR EVEN DEBUGGED NECESSARILY, BECAUSE IT IS NO LONGER USED DUE
   TO THE CHANGE TO THE SCANNER SYSTEM.  IT IS REPLACED BY ICIFIX, AND
   POSSIBLY BY RAXED.  EDT DOES WORK THOUGH, AND HAS BEEN
   EXTENSIVELY USED TO ENHANCE  THE LARGE MTST SAMPLE  A NUMBER OF
   TIMES THROUGH THE CYCLE.
         TREATMENT OF '1/2' CHARACTER IN EDT PROGRAM ..
      TREATMENT FOR FINISH COMMAND IS SAME AS FOR SUCCESSIVE READ, THEN
      WRITE COMMANDS.
      ON INPUT RESULTING FROM F OR R COMMANDS THE PROGRAM STARTS AT
   THE END OF THE INPUT BUFFER AND SEARCHES BACKWARD FOR THE 1ST '1/2'
   CHAR FOR 70 BYTES.  IF A '1/2' CHAR IS FOUND THEN THE PROGRAM REMOVES
   IT AND ALL THE CHARACTERS AFTER IT FROM THE BUFFER.  OTHERWISE A
   WARNING MESSAGE IS GIVEN.  THE CHECK FOR ILLEGAL CHARACTERS IN THE
   REMAINING DATA GIVES A WARNING MESSAGE ON THE CONSOLE FOR EACH ILLEGAL
   CHARACTER ENCOUNTERED UP TO A MAXIMUM COUNT DETERMINED BY PREVIOUS
   CONSOLE COMMAND.  THE DEFAULT IS 5.  THE ERROR MESSAGE GIVES THE
   CHARACTER LOCATION IN THE RAW INPUT BUFFER, THE CHARACTER'S APPEARANCE
   ON THE 1052, AND THE DECIMAL VALUE OF THE CHARACTER.
   THE CHARACTER IS LEFT UNALTERED, BUT IT MAY TRANSLATE WRONG ON OUTPUT
   OR CAUSE TROUBLE IN GENERAL.  IT SHOULD BE CORRECTED BY EDITING, IF POSSIBLE.
/*
//TAPELIST JOB 99999211,WCD,MSGLEVEL=(1,1)
//JOBLIB DD UNIT=2311,VOL=SER=GWSJR1,DISP=OLD,DSN=UTLIB
//G EXEC PGM=DOCPRINT
//SYSPRINT DD SYSOUT=A,UCS=(HN,FOLD)
//SYSIN DD *
            *** LONDON STAGE PROJECT TAPE USAGE LIST ***

   LSP01 NL, FREE
   LSP02 SL, PRINT FILE TAPE, DSN=SYSOUT
   LSP03 SL, FREE
   LSP04 NL, FREE
   LSP09 SL, FREE                          (FORMER REV 2 OF BATCH 1)
   LSP10 SL, ICISCAN SCRATCH OUTPUT, DSN IS LSPPU
   LSP11 SL, STRCT   SCRATCH OUTPUT, DSN IS STRS
   LSP12 SL, LADDER  SCRATCH OUTPUT, DSN IS LSPLD
   LSP13 SL, ITEMGET SCRATCH OUTPUT, DSN IS LSPIT, VOL NO. 1.
   LSP14 SL, ITEMGET SCRATCH OUTPUT, DSN IS LSPIT, VOL NO. 2.
   LSP15 SL, ITEMGET SCRATCH OUTPUT, DSN IS LSPIT, VOL NO. 3.
   LSP16 SL, ITEMGET SCRATCH OUTPUT, DSN IS LSPIT, VOL NO. 4.
   LSP17 SL, REV 3 OF BATCH 1 A94777C6.  3 MORE FIXES, 6C69, 7198, 9015.
   LSP18 SL, DEFAULT SORT OUTPUT.
   LSP19 NL, FREE
   LSP20 NL, FREE
   LSP21 NL, BACKUP OF GWSJR1 9/18/71
   LSP22 NL, GWSJR1 BACKUP 9/20/71 AM
   LSP23 SL, LONGTERM FILE TAPE.
         1 GENSPECS.JN0971, GENERAL SPECS SIGNED ON JUNE 9TH., 1971
         2 LSPMT.MY017110 MTST CORRECTED SAMPLE.
         3 UNALTERED REBLOCKED (3200) BATCH 1 A9R777C6 UNCHKD
   LSP24 SL, FREE
   LSP25 SL, DATATEXT AA1777C6 FROM REV 3 OF BATCH 1., CHKD.
   LSP26 SL, CORRTEXT ZA1777C6 FROM REV 3 OF BATCH 1., CHKD.
   LSP27 SL, STANDARD ICIFRONT OUTPUT, DSN VARIES.
   LSP28 NL, GWSJR1 BACKUP 10/2/71 AM.
   LSP29 NL, GWSJR1 BACKUP 10/9/71 AM
   LSP30 SL, LSPCR IS DSN, AA87789G, CHOPPED FROM LSP026 ZA1777C6
```

LSP31 SL, DSN IS LSPDT, AA87789G FROM AA1777C6 CN LSF25
LSP32 SL, REV 4 OF BATCH NO. 1, 10/11/71 AM ALL PAGE ENTRIES FIXED.
LSP33 SL, SAMPLE OF BATCH 2 AAD7839F REBLOCKED UNALTERED. CHKD.
LSP34 SL, FREE

LSP01 66832 74 10 04
LSP02 66832 74 10 03
LSP03 66832 74 10 2
LSP04 66797 19 10 2 200
LSP09 75304 01 000 11
LSP10 75304 01 000 20
LSP11 75304 01 000 34
LSP12 75304 01 000 44
LSP13 75304 03 000 6
LSP14 75304 03 000 10
LSP15 75304 04 000 9
LSP16 75304 04 000 28
LSP17 75304 04 000 36
LSP18 75304 04 000 37
LSP19 73122 70 000 35
LSP20 73122 70 000 29
LSP21 73122 70 000 27
LSP22 73122 29 000 28
LSP23 75939 42 001 39
LSP24 79053 01 001 41
LSP25 79053 01 001 39
LSP26 79053 01 001 15
LSP27 79053 01 002 39
LSP28 79053 01 001 45
LSP29 79053 01 001 13
LSP30 79053 01 003 47
LSP31 79053 01 001 43
LSP32 79053 01 002 43
LSP33 79053 01 002 47
LSP34 73122 89 000 17
LSP35 78958 36 000 41
LSP36 78938 32 000 31
LSP37 78938 32 000 9
LSP38 78938 32 000 18
LSP39 78938 34 000 41
LSP40 78958 36 000 15
LSP41 78938 34 000 20
LSP42 78938 32 000 41
LSP43 78938 34 000 10
LSP44 78938 34 000 30
LSP45 360/000988
LSP46 (NO APPARENT SERIAL NO.)
/*

THE SYSTEM WAS ORIGINALLY DESIGNED TO USE THE MTST FORMAT, BUT IN APRIL 1970 THE DECISION WAS MADE TO USE A DATA PREPARATION FIRM IN HONGKONG AND OPTICALLY SCAN THEIR TYPEWRITTEN OUTPUT. THIS METHOD WAS MUCH CHEAPER THAN THE MTST METHOD BUT IT WAS SLOWER, MORE ERROR PRONE, AND IT REQUIRED NEW PROGRAMMING. IN EMPLOYING THIS METHOD IT WAS THOUGHT WISER NOT TO HAVE THE GIRLS IN HONGKONG ATTEMPT TO DO SOME OF THE MORE SOPHISTICATED TRICKS INVOLVED IN MAKING A PRECISE LOGICAL REFLECTION OF THE SEMANTIC INFORMATION, UNDERSTANDABLE ONLY TO HUMANS, IN THE LONDON STAGE.

THE GENERAL SPECIFICATIONS OF JUNE 9, 1971 DESCRIBE THIS ORIGINAL SYSTEM OF SCANNER-ORIENTED PROGRAMS, WHICH WERE ALMOST IDENTICAL TO THE MTST PROGRAMS EXCEPT THAT SAVEDT (MTST) WAS REPLACED BY ICISCAN (SCANNER).

SOME TIME LATER THE DATA PREPARATION COMPANY INFORMED US OF ITS NEED FOR AN ERROR CORRECTION SYSTEM. APPARENTLY THEIR ERROR RATE, AS THEY DETERMINED IT, WAS NOT WITHIN THE AGREED LIMIT. ON JULY 21, 1971 THEY WERE SENT A COPY OF 'CORRECTION INSTRUCTIONS FOR THE LONDON STAGE PROJECT', DESCRIBING THE SYSTEM WHICH WE DEVELOPED.

```
    SOME TIME AFTER THE FIRST SCANNED SAMPLE WITH CORRECTION TEXT
ARRIVED ON AUGUST 20, 1871, HOWEVER, WORK ON THE CORRECTION
PROGRAM WAS SUSPENDED BECAUSE THE RATIO OF CORRECTION TEXT TO DATA
TEXT IN THE SAMPLE WAS ONLY 1 PERCENT.  THIS WAS SO SMALL THAT IF
THE RATIO HELD FOR ALL DATA THEN ONLY ABOUT 75 DOLLARS OF TYPING
WOULD BE PUT INTO CORRECTION TEXT.  THIS AMOUNT WOULD JUSTIFY
ONLY A FEW DAYS OF PROGRAMMING.  LATER BATCHES PROVED TO HAVE LARGER
PROPORTIONS OF CORRECTION TEXT, BUT OTHER WORK KEPT THE CORRECTION
SYSTEM FROM BEING COMPLETED UNTIL EARLY OCTOBER 1971, SHORTLY
BEFORE THE FIRST SAMPLE OF BATCH 2 ARRIVED.  (UNFORTUNATELY THIS
SAMPLE HAD NO CORRECTION TEXT IN IT).
    FORMATS FOR NONSTANDARD TIME ENTRIES ..
1)  END/ FIRST CAP.
2)  END/ RMN/ FIRST CAP.
3)  END OF ACT/ RMN/ FIRST CAP
4)  IN ACT/ RMN/ FIRST CAP.
1)  FOR 'END' OR 'END OF ACT' SET UP SAMLL 'A'.  FOR 'IN ACT' SMALL'D'.
2)  CHECK FOR RMN, IF FIND THEN ACCEPT AND PUT OUT AS TIME ENTRY.
                  ELSE TAKE FIRST CAP IF ANY & PUT OUT WITH TIME ENTRY.
                  ELSE TAKE FIRST CAP IF ANY & PUT OUT AS TIME ENTRY.
3)  IF NO CAP THEN TAKE AS ROLE OR ACTOR.
                     LSP SYSTEM GENERATION
1)  IPL NOT USING LSP'S PROCLIB ON GWSJR1.
2)  VARY GWSJR1 (TO BE) OFFLINE.
3)  IEHDASDR AS SINGLE JOB IN STREAM.  VTOC=2, EXTENT=3.
4)  VARY GWSJR1 ONLINE AGAIN.
5)  RUN IEBCOPY AS SINGLE JOB WITH NEW SPACE=
6)  RE-IPL USING SYS1.PROCLIB ON GWSJR1.
7)  RUN PLIST COMP USING LSPUTGEN AND OUTP=NEW, OSP=XXXX. ALSO
    OVERRIDING TEMP DATASET VOL PARMS TO ENSURE UTLIE WILL
    BE ALLOCATED AT BOTTOM OF GWSJR1.
7)  RUN ICIFIX WITH LSPUPGEN AND OSSP=NEW, SSP=     .
    ALSO OVERRIDE TEMP DATASET ALLOCS AGAIN, SAME REASON.
9)  RUN OTHER COMPILES, INC ITEMS.  SAVE COMPLETED DISK
    ON 2 SEPARATE TAPES.
10)  DELETE OBLIB, SRCLIB
11)  SAVE DISK AGAIN ON ONE OR BOTH OF THE TAPES.
    ICIFIX, ICISCAN, STRCT, LADDR, ITEMS, FRMAT, DOCPRINT,
    PLIST, LSPRINT, LSPMAIL.
/*
//EDT JOB 99999211,WCD,MSGLEVEL=1
//E EXEC LSPBACMP,OSSP=OLD,OOSP=OLD,SSP=10,OSP=30,OSD=EDT
//BAKUP.SYSUT1  DD *,DCB=BLKSIZE=80
    (SUBRG,STRG) ..
    EDIT .. PROC OPTIONS(MAIN) ,.
    /* THE NUMBERS REPRESENTED BY 'XX' IN THE COMMAND EXAMPLES ARE
       VARIABLE IN LENGTH.                                        */
    /* UPDATE COMMANDS ARE SMALL LETTERS.
        'F'  FINISH, READ AND WRITE CONSECUTIVE BLKS UNTIL END OF FILE.
        'L'  LIST, LIST ENTIRE CURRENT BLK IN MTST FORMAT WITH LINE NUMBERS.
        'DXX'  DELETE, DELETE LINE XX.
        'R'  READ, READ NEXT BLK INTO CORE.
        'W'  WRITE, WRITE OUT CURRENT BLK TO OUTPUT FILE.
        'S'  SWITCH, SWITCH PRINT ON BLK I/O TO OPPOSITE STATE.
                     THIS IS USED FOR BOTH REGULAR I/O, AND IN THE
                     UPDATE MODE TO CONTROL LISTING IN MTST FORMAT.
                     THE SWITCH IS INITIALLY IN THE 'OFF' STATE.
                     CONTROLS LISTING OF RAW INPUT FOR THE 'R' AND 'F' COMMANDS
                     CONTROLS RAW OUTPUT LISTING FOR 'W' AND 'F' COMMANDS
                     CONTROLS MTST FORMAT LISTING FOR 'U' COMMAND.
        'IXX YYYYY$' INSERT, INSERT STRING 'YYYYY' AFTER LINE XX.
                             LINE 0 DOES NOT EXIST, BUT YOU CAN PUT
                             SOMETHING AFTER IT ANYWAY.
        'CXX YYYYY$'  CHANGE, SAME AS DELETE FOLLOWED BY INSERT WITH
                             THE SAME LINE NUMBER.
        'U'  UPDATE, CLOSE CURRENT OUTPUT FILE, THEN REOPEN IT FOR
```

```
                    INPUT AND COPY BACK TO ORIGINAL INPUT FILE,
                    LISTING IN MTST FORMAT IF BIT1 SWITCH IS IN THE
                    ON POSITION, I.E., IF IT IS '1'B.

          ALL COMMANDS MAY FOLLOW ANY OTHER COMMAND.  THE END OF A
       COMMAND LIST IS INDICATED BY END OF INPUT STRING.
       HOWEVER THE 'F' AND 'U' COMMANDS RESULT IN A RETURN TO THE 'OK'
  MESSAGE AFTER COMPLETION.  COMMANDS FOLLOWING THE  F  &  U  COMMANDS ARE
  NOT EXECUTED.
     TO END THE PROGRAM TYPE A CAPITAL  R  AS A SPECIAL COMMAND.
     THERE IS A PROVISION FOR EASY CORRECTION OF TYPING ERRORS THAT YOU
  DETECT RIGHT AFTER MAKING.  IF YOU HAVE TYPED A WRONG CHARACTER THEN
  THAT CHARACTER CAN BE DELETED BY TYPING AN  AT  CHARACTER IMMEDIATELY
  AFTER THE INCORRECT CHARACTER.  IN FACT YOU CAN DELETE MORE PREVIOUS
  CHARACTERS BY TYPING MORE  AT SIGNS.  THUS 2 SUCCESSIVE  AT  SIGNS WILL
  DELETE THE TWO CHARACTERS PRECEDING THE  AT SIGNS.
     WHEN YOU WANT TO INSERT A QUOTE INTO THE TEXT YOU MUST TYPE IT AS 2
  SUCCESSIVE QUOTES.  A SINGLE QUOTE IS USED BY THE CONTROL PROGRAM TO
  DELIMIT THE END OF ALL INPUT, AND THE EDITING PROGRAM IS NOT OF THE
  QUOTE OR ANY CHARACTER FOLLOWING IT.
     NOTE .. TWO QUOTES FOLLOWED BY AN  AT  SIGN RESULTS IN A SINGLE QUOTE
  OF INPUT, I.E. YOU NEED 2  AT  SIGNS TO DELETE A DOUBLE QUOTE.
          THE INPUT STRING MUST BE 126 CHARACTERS OR LESS.  NO BLANKS ARE
          ALLOWED ANYWHERE EXCEPT AFTER THE 'IXX' OR THE 'CXX', WHERE ONE
          IS REQUIRED, AND IN THE STRING, 'YYYYY'.                      */
          DCL (A,B) CHAR(3625) VARYING,
          HEX ENTRY(CHAR(132) VAR) RETURNS(CHAR(264) VAR),
          SYSPRINT PRINT ENV(F(133,133)),
          P CHAR(3625) VAR,
          ISTACK1 INITIAL(05),
          IOPT(32) BIT(1),
          IO INITIAL(1),
          HEX00 CHAR(1) INITIAL(' '),
          HEX0D CHAR(1) INITIAL('?'),
          HEX25 CHAR(1) INITIAL('?'),
          HEX62 CHAR(1) INITIAL('?'),       /* THE '1/2' CHARACTER    */
          EBC CHAR(1) INITIAL('c'),
          EBL CHAR(1) INITIAL('l'),
              CCHAR(9) CHAR(1) INITIAL('f','l','d','r','w','s','i',
                                       'c','u'),
              PROCED(9) LABEL INITIAL(PUTTHRU,LISTALL,DELETE,RDELK,
              WRTBLK,SRCH,INSERT,CHANGE,UPDATER),
          T1DEF(8) CHAR(32) INITIAL(
               'z256??091378?|4?tnek??hbxudc??1?',
               '>.<i??swmvra??o?j=pq ?y-gf?,??;?',
               'Z¬%/??)(!+ε*?|$?TNEK??HBXUDC??L?',
               '?''?I??SWMVRA??O?J@PQ ?Y_GF?''??:?',
               '????????????????????????????????',
               '????????????????????????????????',
               '????????????????????????????????',
               '????????????????????????????????'),
          T1 CHAR(256) DEFINED T1DEF,
          T2DEF(8) CHAR(32) INITIAL(
               '??????????????????????????-????',
               '??????????????????????????:????',
               '??????????????¢??????????+.???',
               '????????????????????????????=??/??',
               '????????????????????????????????',
               '????????? ????????????????????',
               '?,?$!????????????????;??>?????????',
               '???&????? ????????????????????'),
          T2 CHAR(256) DEFINED T2DEF,
          J INITIAL(0) ,
          IBADCHAR INITIAL(0) ,
          BADCHAR CHAR(1) DEF IBADCHAR POS(2),
          LNS DEF LINESOUGHT,
```

```
        BIT1 BIT(1) DEF IOPT,
        C CHAR(126) VARYING,
        STR CHAR(126) VARYING,
        ICMND INITIAL(0),
        CMND CHAR(1) DEF ICMND POS(2),
        SMOD CHAR(1) ,.
        ON ENDFILE(IN) BEGIN ,.
        DISPLAY('END OF FILE') ,.   GOTO OK ,.   END ,.
        ON ENDFILE(SYSIN) ,.
        GET FILE(SYSIN) EDIT(IOPT) (B(1)) ,.
        ON ERROR SNAP CALL PUTDATA ,.
    PUTDATA .. PROC ,.
        PUT FILE(SYSPRINT) SKIP(1) LIST('ERROR ONCODE=',ONCODE) ,.
        PUT DATA ,.   END ,.
        IF IOPT(30) THEN
        DO I=1 TO 8 ,.
        DISPLAY(T1DEF(I) CAT '''   ''' CAT T2DEF(I) CAT '''') ,.
        END ,.
    OK ..
        C=REPEAT(HEX00,126) ,.   /* INITIALIZE C WITH HEX ZEROS FOR  */
                                 /* DELIMITING END OF INPUT ON 1052. */
        DISPLAY('OK') REPLY(C) ,.
        ICL=INDEX(C,HEX00) ,.
        IF IOPT(21) THEN PUT SKIP DATA(ICL,C) ,.
        IF ICL=0 THEN ICL=126 ,.   ELSE ICL=ICL-1 ,.
        IF IOPT(21) THEN PUT DATA(C,ICI) ,.
        C=SUBSTR(C,1,ICL) ,.
        IF IOPT(20) THEN DO ,.
        DISPLAY(ICL CAT ' ''' CAT SUBSTR(C,1,12) CAT '''' CAT
        HEX(SUBSTR(C,1,10))) ,.
        END ,.
        ON STRG ,.
    GETRUB .. N=INDEX(C,'@') ,.
        IF N GT 1 THEN DO ,.
           C=SUBSTR(C,1,N-2) CAT SUBSTR(C,N+1) ,.
           ICL=LENGTH(C) ,.   GOTO GETRUB ,.   END ,.
        N=0 ,.
    GETQUOTE ..
        L=N ,.
        N=INDEX(SUBSTR(C,L+1),'''''') ,.
        IF N GT 0 THEN DO ,.   N=L+N ,.
                               C=SUBSTR(C,1,N) CAT SUBSTR(C,N+2) ,.
                               GOTO GETQUOTE ,.   END ,.
        REVERT STRG ,.
        N=0 ,.
        IF IOPT(20) THEN PUT FILE(SYSPRINT) SKIP(1) EDIT
        ('FIXED STRING=''',C,'''') (A) ,.
    NXTCMND ..
        IF N GE ICL THEN GOTO OK ,.
        N=N+1 ,.
        LINESOUGHT=20000 ,.      /* SET LINE SOUGHT OUT OF SIGHT     */
        CMND=SUBSTR(C,N,1) ,.    /* GET COMMAND LETTER               */
        IF IOPT(9) THEN DO ,.
        PUT SKIP EDIT('CMND=',CMND,ICMND) (A) ,.
        DISPLAY(CMND CAT ICMND) ,.
        END ,.
        CALL FIX ,.
        IF CMND GE '0' THEN IF CMND LE '9' THEN GOTO STACKUP ,.
        IF CMND=',' THEN DO ,.
           ISTACK2=ISTACK1 ,.   N=N+1 ,.   GOTO STACKUP ,.   END ,.
        IF CMND='m' THEN DO ,.
           PUT FILE(SYSPRINT) SKIP(1) EDIT
           ((IX DO IX=1 TO 9),
           (SUBSTR(B,L,MIN(100,I-L+1)) DO L=1 TO I BY 100))
           (9 F(10),99 (SKIP,A)) ,.
           GOTO NXTCMND ,.   END ,.
```

```
      IF CMND='b' THEN DO ,.
         A=B ,.   GOTO NXTCMND ,.   END ,.
      IF CMND='a' THEN DO ,.
         B=A ,.   GOTO NXTCMND ,.   END ,.
      IF CMND='L' THEN DO ,.
MORELIST ..
      CMND='1' ,.
      CALL INPUT ,.   CALL MTLIST ,.   GOTO MORELIST ,.   END ,.
      IF CMND='p' THEN DO ,.
      PUT SKIP LIST('BEGINNING PUT',ISTACK2,ISTACK1) ,.
         P=SUBSTR(B,1,ISTACK2-1) CAT SUBSTR(E,ISTACK1+1) ,.
         B=SUBSTR(B,ISTACK2,ISTACK1-ISTACK2+1) ,.
         CALL OUTPUT ,.   B=P ,.   GOTO NXTCMND ,.   END ,.
      IF CMND='t' THEN DO ,.
         CALL GETLINE ,.
      PUT SKIP LIST('DOING TYPE',LINC,SUBSTR(B,K,INTST-K)) ,.
         DISPLAY(LINO CAT SUBSTR(B,K,INTST-K)) ,.
         GOTO NXTCMND ,.   END ,.
      DO IX=1 TO 9 ,.
      IF CCHAR(IX)=CMND THEN GOTO PROCED(IX) ,.
      END ,.
      IF CMND='R' THEN RETURN ,.
      IF CMND=' ' THEN GOTO OK ,.
      DISPLAY('UNDEFINED COMMAND ' CAT '''' CAT CMND CAT '''') ,.
      GOTO OK ,.
STACKUP .. N=N-1 ,.
      CALL GETLINE ,.
      ISTACK1=LNS ,.
      PUT SKIP DATA(ISTACK1,ISTACK2) ,.
      N=N-1 ,.
      GOTO NXTCMND ,.
CHANGE ..
INSERT ..
      CALL GETLINE ,.
      IX=INDEX(SUBSTR(C,N+1),'$') ,.
      IF IX=0 THEN DO ,.
      DISPLAY('UNABLE TO FIND ''$'' DELIMITER') ,.
      PUT SKIP(3) DATA(IX,C,N) ,.
      GOTO OK ,.
      END ,.
      IX=IX+N ,.
      STR=SUBSTR(C,N,IX-N) CAT '??' ,.
      N=IX ,.          /* SET PTR TO NEXT COMMAND FOR CONTINUOUS CMND*/
      IX=L ,.          /* SET PTR TO CAUSE INSERTION ONLY.          */
      IF CMND=EBC THEN DO ,.   /* MASSAGING FOR CHANGE RATHER THAN */
                               /* INSERT INVOLVES PRINTING AN EXTRA*/
                               /* MESSAGE, AND THEN RESETTING A    */
                               /* PTR TO RESULT IN DELETION.       */
      PUT FILE(SYSPRINT) SKIP(1) EDIT
      ('LINE',LINO,', ''',SUBSTR(B,K,L-K)  ,''' DELETED BY CHANGE')
      (A,F(3),3 A) ,.
         IX=K ,.     /* RESET PTR TO CAUSE DELETION TOO.          */
         END ,.
      B=SUBSTR(B,1,IX-1) CAT STR CAT SUBSTR(E,L) ,.
      CALL FIX ,.
      PUT FILE(SYSPRINT) SKIP(1) EDIT
      ('''',STR,''' INSERTED AS LINE',LINC      ) (3 A,F(3)) ,.
      GOTO NXTCMND ,.
DELETE ..
      CALL GETLINE ,.
      STR=SUBSTR(B,K,L-K+1) ,.
      B=SUBSTR(B,1,K-1) CAT SUBSTR(B,L) ,.
      CALL FIX ,.
      PUT FILE(SYSPRINT) SKIP(1) EDIT('LINE',LINO      ,', ''',
      STR,''' DELETED') (A,F(3),3 A) ,.
      N=N-1 ,.
```

```
        GOTO NXTCMND ,.
    SRCH ..
        CALL GETLINE ,.
        IOPT(LNS)=NOT IOPT(LNS) ,.
         DISPLAY('OPT' CAT LNS CAT ' =' CAT IOPT(INS)) ,.
        N=N-1 ,.
        GOTO NXTCMND ,.
    PUTTHRU ..
        PUT SKIP LIST('********** FINISH COMMAND GIVEN') ,.
    PUTPUT ..
        CALL INPUT ,.
        CALL OUTPUT ,.
        GOTO PUTPUT ,.
    GETLINE .. PROC ,. /* ASSUMES DECIMAL NUMBER CHAR STRING STARTS  */
                       /* IN C AT OFFSET=(N+1) FROM START OF C       */
        CALL FIX ,.
        IRC=VERIFY(SUBSTR(C,N+1),'0123456789') ,.
        IF IRC LE 1 THEN DO ,.
        DISPLAY('SUSPICIOUS NUMERICAL PARAMETER') ,.
        PUT SKIP(3) DATA(IRC,C,N) ,.
        GOTO OK ,.
        END ,.
        IRC=IRC+N ,.    /* SETS IRC TO PT TO 1ST CHAR AFTER NUMBER
                        /* STRING'S END.                              */
        GET STRING(SUBSTR(C,N+1,IRC-N-1)) EDIT(LINESOUGHT)
        (F(IRC-N-1)) ,.
        N=IRC ,.        /* NOW N PTS TO 1ST CHAR AFTER NUMBER STRING  */
        SMOD=SUBSTR(C,N,1) ,.
        IF SMOD=' ' THEN N=N+1 ,.
        CALL MTLIST ,. /* FIND LINE WHOSE NUMBER IS HELD IN          */
                       /* THE VARIABLE LINESOUGHT.                   */
        END ,.
    MTLIST .. PROC ,.  /* FINDS A LINE AND SETS UP PTRS TO THE       */
                       /* BEGINNING AND END OF IT.  ALSO LISTS LINES */
                       /* AS IT IS SEARCHING IF THE COMMAND LETTER   */
                       /* IN THE VARIABLE, CMND, IS 'L'.             */
             /* ON RETURN K PTS TO 1ST CHAR OF LINE.  L PTS TO 1ST
                CHAR OF NEXT LINE, IF ANY.  IF NO NEXT LINE THEN L
                PTS TO I+1 WHERE I IS THE CURRENT LENGTH OF THE B
                BUFFER.  BOTH K & L =1 IF LINE SPECIFIED IS LINE 0.
                IF THE LINE NUMBER SPECIFIED IS GREATER THAN THE
                THE NUMBER OF THE ACTUAL LAST LINE THEN THE LAST LINE
                IS DELIVERED INSTEAD.                                 */
        IF CMND=EBL THEN
        PUT FILE(SYSPRINT) SKIP(2) EDIT
        ('********** LISTING REQUESTED OF INPUT BLK NO.',J,
        ',   OUTPUT BLK NO. WILL BE',IO,',   LENGTH=',I)
        (2 (A,F(4)),A,F(5)) ,.
        LINO=0 ,.
        L=1 ,.
        K=1 ,.
        ON STRG SNAP PUT DATA(K,I,L) ,.
    NEXTLINE .. IF LINO=LINESOUGHT THEN RETURN ,.
        LINO=LINO+1 ,.
        K=L ,.
        L=INDEX(SUBSTR(B,K,I-K+1),'?') ,.  /* 12-9-8-5 EBCDIC CR 0D  */
        IF L=0 THEN L=I ,.  ELSE L=L+K-1 ,.
        IF CMND='1' THEN PUT FILE(SYSPRINT) SKIP(1) EDIT
        (LINO,SUBSTR(B,K,L-K)) (F(3),X(4),A) ,.
        INTST=L ,.
    FINDSTRT ..
        IF L=I THEN RETURN ,.
        SMOD=SUBSTR(B,L,1) ,.
        IF SMOD NE '?' THEN IF SMOD NE '?' THEN GOTO NEXTLINE ,.
        L=L+1 ,.
        GOTO FINDSTRT ,.
```

```
        END ,.
    FIX .. PROC ,.
        I=LENGTH(B) ,.
    CUTBACK ..
        IF I=0 THEN GOTO APPEND ,.
        SMOD=SUBSTR(B,I,1) ,.
        IF SMOD=HEX0D OR SMOD=HEX25 THEN DO ,.
        I=I-1 ,.   B=SUBSTR(B,1,I) ,.   GOTO CUTBACK ,.   END ,.
    CUTFORWARD ..
        IF I=0 THEN GOTO APPEND ,.
        SMOD=SUBSTR(B,1,1) ,.
        IF SMOD=HEX0D OR SMOD=HEX25 THEN DO ,.
        I=I-1 ,.   B=SUBSTR(B,2) ,.   GOTO CUTFORWARD ,.   END ,.
    APPEND ..
        B=B CAT HEX0D CAT HEX25 ,.   I=I+2 ,.   END FIX ,.
    INPUT .. PROC ,.
        READ FILE(IN) INTO(A) ,.
        IF IOPT(20) THEN
        PUT SKIP LIST('AFTER INPUT READ') ,.
        I=LENGTH(A) ,.
        B=TRANSLATE(A,T1) ,.
        J=J+1 ,.
        IF BIT1 THEN
        PUT SKIP LIST(B) ,.
        DO K12=I TO  MAX(I-70,1) BY -1 ,.
        IF SUBSTR(B,K12,1)=HEX62 THEN GOTO FOUND12 ,.   END ,.
        DISPLAY('NO ''1/2'' CHAR WITHIN 70 BYTES OF ECB') ,.
        GOTO LENFIXED ,.
       FOUND12 ..
        I=K12-1 ,.
      B=SUBSTR(B,1,I) ,.
     LENFIXED ..
        A=B ,.
        DO K12=1 TO I ,.
        IF SUBSTR(A,K12,1)=HEX62 THEN SUBSTR(A,K12,1)='?' ,.   END ,.
        K12=0 ,.
        IERRCNT=0 ,.
    RESEARCH ..
        L=INDEX(SUBSTR(A,K12+1),'?') ,.
        IF L=0 THEN RETURN ,.
        IERRCNT=IERRCNT+1 ,.
        K12=L+K12 ,.
        BADCHAR=SUBSTR(B,K12,1) ,.
        DISPLAY('BAD CHAR ' CAT BADCHAR CAT ' AT' CAT K12
        CAT HEX(BADCHAR)) ,.
        IF K12=I THEN RETURN ,.
        IF IERRCNT GT ISTACK1 THEN RETURN ,.
        GOTO RESEARCH ,.
        END ,.
    OUTPUT .. PROC ,.
        B=B CAT HEX62 ,.          /* REINSERT '1/2' CHAR ON OUTPUT    */
        IF IOPT(20) THEN
        PUT SKIP LIST('BEGINNING OUTPUT PROCESSING.'),.
        A='A IS THIS SENTENCE JUST BEFORE OUTPUT TRANSLATION.' ,.
        A=TRANSLATE(B,T2) ,.
        IF BIT1 THEN
        DO ,.
        PUT SKIP LIST(B) ,.
        PUT SKIP LIST(TRANSLATE(A,T1)) ,.
        END ,.
        WRITE FILE(TOUT) FROM(A) ,.
        IO=IO+1 ,.
        END ,.
    LISTALL .. CALL MTLIST ,.
        PUT SKIP(32) ,.
        GOTO NXTCMND ,.
```

```
        RDBLK .. CALL INPUT ,.
            IF IOPT(1) THEN
            PUT FILE(SYSPRINT) SKIP(1) EDIT
            ('BLK',J,' READ, LENGTH=',I) (A,F(4),A,F(5)) ,.
            GOTO NXTCMND ,.
        WRTBLK .. CALL OUTPUT ,.
            GOTO NXTCMND ,.
        UPDATER ..
            CLOSE FILE(TOUT) ,.
            CLOSE FILE(IN) ,.
            ON ENDFILE(TOUT) GOTO ENDPROG ,.
        PUTBACK ..
            PUT FILE(SYSPRINT) SKIP(2) LIST
            ('********** UPDATE COMMAND GIVEN') ,.
            DO J=1 TO 2000 ,.
            IO=J ,.
            READ FILE(TOUT) INTO(A) ,.
            IF BIT1 THEN DO ,.
            B=TRANSLATE(A,T1) ,.   CALL FIX ,.
            CMND=EBL ,.   CALL MTLIST ,.   END ,.
            IF IOPT(2) THEN
            WRITE FILE(IN) FROM(A) ,.
            END ,.
            CLOSE FILE(IN) ,.   CLOSE FILE(TOUT) ,.
            GOTO OK ,.
        HEX .. PROC(CH) RETURNS(CHAR(264) VAR) ,.
            DCL
            CH CHAR(133) VARYING,
            CHX CHAR(266),
            DCHX(266) CHAR(1) DEF CHX,
            IC3 INITIAL(0),
            ICH3 CHAR(1) DEF IC3 POS(2),
            HTAB(0 .. 15) CHAR(1) INITIAL('0','1','2','3','4','5','6',
               '7','8','9','A','B','C','D','E','F') STATIC ,.
            IC1=LENGTH(CH) ,.
            DO IC2=1 TO IC1 ,.
            ICH3=SUBSTR(CH,IC2,1) ,.
            DCHX(IC2+IC2-1)=HTAB(IC3/16) ,.
            DCHX(IC2+IC2)=HTAB(MOD(IC3,16)) ,.
            END ,.
            RETURN(SUBSTR(CHX,1,IC1*2)) ,.
            END ,.
        ENDPROG .. END ,.
    //SAVEDT JOB 99999211,WCD,MSGLEVEL=1
    //E EXEC LSPUPGEN,MEM=SAVEDT
    //BAKUP.SYSUT1  DD *,DCB=BLKSIZE=80
        SAVEDIT .. PROC OPTIONS(MAIN) ,.
            PUT EDIT('PROGRAM STARTED') (A) ,.
        G .. FORMAT(A,F(4)) ,.
        H .. FORMAT(A,F(5)) ,.
            DCL
            N INITIAL(0),
            IOPT(32) BIT(1),
            NOTELIST(0 .. 36) CHAR(3) VARYING,
            IPAGES(0 .. 36),
            ERRFILE(0 .. 36) CHAR(4),
            SYSPRINT PRINT ENV(F(133,133)),
            ERRMSG ENTRY(CHAR(120) VARYING,FIXED BIN(15,0),FIXED BIN(15,0),
               CHAR(1) VARYING,BIT(1)),
           (A,B,P) CHAR(3625) VARYING,
            C CHAR(120) VARYING,
            CHPAGE CHAR(4) INITIAL('8888'),
            CH14 CHAR(20),
            C4 CHAR(4) ,
            CONGO LABEL(GETPAG, NOTAPE),
            INDXRTRN LABEL(NEXTSEC,NEXTBRKTS) ,
```

```
        BIT1 BIT(1) ,
        SMOD CHAR(1) VARYING,    /* CURRENT SECTYPE INDICATOR CHAR   */
        EBC CHAR(1) INITIAL('c'),     /* SMALL EBCDIC 'C' */
        EBSML CHAR(26) INITIAL('abcdefghijklmnopqrstuvwxyz'),
        EBSMQ CHAR(27) INITIAL('abcdefghijklmnopqrstuvwxyz'''),
        EBLBRKT CHAR(1) INITIAL('['),
        EBRBRKT CHAR(1) INITIAL(']'),
        P0_9_6H26 CHAR(1) INITIAL('?'), /* EOF GARBAGE CODES IN MT */
        CH2 CHAR(2),
        CH3 CHAR(3),
        C1 CHAR(1) DEFINED CH3 POSITION(3),
        C2 CHAR(1) DEFINED CH3,
        M12 CHAR(1) INITIAL('?'), /*11.0.9.2, H'62' FOR MT & EBCDIC */
        MTPRDCR CHAR(2) INITIAL('?|'),
        MTQSTCR CHAR(2) INITIAL('??'),
        MTEXCR CHAR(2) INITIAL('??'),
        MTLF CHAR(1) INITIAL('?'),    /* 11.9.5 PUNCHES H'15' */
        MT3BLNK CHAR(3) INITIAL('???'),
        MTASCS  CHAR(17) INITIAL((17) '?' ),
        LRECRD CHAR(18),
        T1DEF(8) CHAR(32) INITIAL(
           'z256 ?091378?|4?tnek? hbxudc??1?',
           '].[i??swmvra??o?j=pq ?y-gf?,??;?',
           'Z¬%/ ?)(!+&*?|$?TNEK? HBXUDC??L?',
           '?''?I??SWMVRA??O?J@PQ ?Y_GF?''??:?',
           '????????????????????????????????',
           '????????????????????????????????',
           '????????????????????????????????',
           '????????????????????????????????'),
        T1 CHAR(256) DEFINED T1DEF,        /* MTST TO EBCDIC TR TABL */
        ISKIP INITIAL(0),
        J INITIAL(0),.
        ON ENDFILE(IN) GOTO ENDPROG ,.
        ON CONVERSION GOTO CONGO ,.        /* LABEL VARIABLE         */
        ON ENDFILE(SYSIN) ,.
        GET FILE(SYSIN) EDIT(IOPT) (B(1)) ,.
        NP=0 ,.
    INPUT .. READ FILE(IN) INTO(A) ,.
        J=J+1 ,.       /* J NOW = THE NUMBER OF GOOD INPUT BLKS      */
        I=LENGTH(A) ,.
        IF I LE 20 THEN DO ,.
           PUT FILE(SYSPRINT) SKIP(4) EDIT
           ('BLK',J,' IS SHORT, LENGTH =',I) (2 (R(G)))
        ('''',TRANSLATE(A,T1),'''') (SKIP,A,A,A) ,.
           J=J-1 ,.
           GOTO INPUT ,.
           END ,.
        ELSE WRITE FILE(TOUT) FROM(A) ,.
        IPAGES=NP ,.
        B=TRANSLATE(A,T1) ,.
        PUT FILE(SYSPRINT) SKIP(3) EDIT
        ('BLK',J,' HAS A LENGTH OF',I,' *** ''',
        SUBSTR(B,1,MIN(I,70)),''' ***')
        (R(G),R(H),3 A) ,.
        K=2 ,.
    PERIODS .. M=K ,.
        K=INDEX(SUBSTR(A,K+1),MTIF) ,.     /* SEARCH FOR A FEED CODE */
        IF K=0 THEN GOTO TABS ,. /* IF NOT FIND THEN ALL DONE        */
        K=M+K ,.                   /*   NOW K POINTS TO THE FEED     */
        CH2=SUBSTR(A,K-2,2) ,.
        IF CH2 NE MTPRDCR    AND CH2 NE MTEXCR AND CH2 NE MTQSTCR THEN
             DO ,.                          /* REMOVE A FEED CHAR     */
             K=K-1 ,.
             A=SUBSTR(A,1,K) CAT SUBSTR(A,K+2) ,.
             END ,.
          GOTO PERIODS ,.       /* DELETE REMAINING FEEDS IF ANY     */
```

```
TABS ..
    ILX=INDEX(B, '|t') ,.     /* CHECK FOR NEW MTST TAPE ENTRY     */
    IF ILX=0 THEN GOTO NOTAPE ,.
    IRX=INDEX(SUBSTR(B,ILX+2,5),' ') ,.          /* GET NEXT BLNK*/
    IF IRX LT 2 THEN GOTO NOTAPE ,.    /* WRONG LENGTH, GIVE UP  */
    C=SUBSTR(B,ILX+2,IRX-1) ,.         /* SET UP FOR CONVERSION  */
    ION=N ,.                    /* SAVE LAST MTST TAPE NUMBER     */
    CONGO=NOTAPE ,./* DEFINE WHERE TO GO ON CONVERSION INTERRUPT */
    GET STRING(C) EDIT(N) (F(IRX-1)) ,.          /* N RESERVED   */
    MTNO=0 ,.                   /* RESET BLK CNT FOR MTST TAPE    */
    IF IOPT( 2) THEN
    IF ILX GT 9 THEN PUT FILE(SYSPRINT) SKIP(1) EDIT
    ('POSSIBLE ERROR - MT TAPE ENTRY STARTS AT',ILX) (A,F(4)) ,.
    IF IOPT(15) THEN
    IF N NE ION+1 AND ION NE 0 THEN DO ,.
       PUT FILE(SYSPRINT) SKIP(2) EDIT
       ('POSSIBLE ERROR ***** NONCONSECUTIVE MTST ENTRIES,',
       ION,', ',N) (2 R(G)) ,.
       CALL ERRMSG('MTOO',ILX,2,SMOD,'0'B) ,.
       END ,.
NOTAPE ..
    MTNO=MTNO+1 ,.              /* INCREMENT MTST TAPE BLK CNT    */
    PUT FILE(SYSPRINT) SKIP(1) EDIT
    ('REF. CODE',MTNO,' OF MTST TAPE NO.',N) (2 R(G)) ,.
    IF IOPT( 1) THEN PUT FILE(SYSPRINT) SKIP(2) EDIT(B) (A) ,.
    L=INDEX(B,'?') ,.
    IF L NE 0 THEN PUT FILE(SYSPRINT) SKIP(2) EDIT
    ('ILLEGAL(0.8.2) CHAR FOUND IN BLK',J,
    ' AT POSITION',L) (2 (R(G))) ,.
    I=LENGTH(A) ,.
    DO M=I TO MAX(1,I-70) BY -1 ,.     /* SRCH BAKWRDS FOR '1/2' */
       IF SUBSTR(A,M,1)=M12 THEN DO ,.
          K=INDEX(A,M12) ,.
    IF K NE M THEN DO ,.
    PUT FILE(SYSPRINT) SKIP(1) EDIT
          ('POSSIBLE ERROR-  ''1/2'' CHAR IN BLK',J,' AT',M)
          ( 2 (R(H))) ,.
    PUT DATA(K,M,I) ,.
    END ,.
          GOTO PRINTBIN ,.
          END ,.
          END ,.
    IF IOPT( 4) THEN
    PUT FILE(SYSPRINT) SKIP(1) EDIT
    ('ERROR-  NO ''1/2'' CHAR WITHIN 70 BYTES OF EOB') (A) ,.
    M=I ,.                                  /*   USE WHOLE BLK    */
PRINTBIN ..
    A=MT3BLNK CAT SUBSTR(A,1,M-1) CAT MT3ELNK ,.
    B=TRANSLATE(A,T1) ,.        /* TRANSLATE MTST TO EBCDIC       */
    K=INDEX(B,P0_9_6H26) ,.  /* CHECK FOR ILLEGAL CODES IN TEXT   */
    IF IOPT( 5) THEN
    IF K NE 0 THEN PUT FILE(SYSPRINT) SKIP(1) EDIT
    ('BLK',J,'HAS AN ILLEGAL(0.9.6) CODE IN LOCATION',K)
    (A,F(4),A,F(4)) ,.
    B=TRANSLATE(A,T1) ,.
    A=REPEAT(' ',3625) ,.
    ERRFILE='' ,.
    NOTELIST='' ,.
    I=LENGTH(B) ,.
    ISR=1 ,.          /* INITIAL OLD RIGHT SECTION DELIMITER      */
    IRX=1 ,.          /* INITIAL INDEX PTR TO START SCANNING      */
                      /* END OF BLK INPUT PROCESSING AND CHECKS   */
NEXTSEC ..
    IF ISR NE 1 THEN
    IF IOPT( 6) THEN
    PUT FILE(SYSPRINT) SKIP(1) EDIT
```

```
            ('''',SUBSTR(B,ISR+2,MIN(99,I-ISR-1)),''' AT',ISR+2)
        (3 A,F(5)) ,.
        ISL=ISR ,.      /* SAVE OLD RIGHT SEC-DELIM AS NEW LEFT ONE   */
    NEXTSECPART ..
        IF ISR=I-2 THEN             /* IF LAST SEC ALREADY PROCESSED    */
             DO ,.                  /* DO PAGE AND OUTPUT SETUP PROCES  */
        IF NOT IOPT(7) THEN GOTO SKIPOPT7 ,.
        PUT FILE(SYSPRINT) SKIP(2) EDIT((K DO K=10 TO 90 BY 10))
        (COL(20),9 (F(2),X(8))) ,.
    SKIPOPT7 ..
        ILX=1 ,.
        CONGO=GETPAG ,.
    GETPAG .. IRX=INDEX(SUBSTR(B,ILX+1),' p'),.    /* 12.11.7 H'97'  */
        IF IRX=0 THEN GOTO GOTPAG ,.        /* ALL PAGE ENTRIES DONE  */
        IRX=IRX+ILX+1 ,.            /* IRX NOW POINTS TO THE 'P' OF A   */
                                    /* PROSPECTIVE PAGE ENTRY           */
        ILX=IRX ,.                  /* SAVE PTR TO START NEXT PAGE SRCH */
        IRB=INDEX(SUBSTR(B,IRX+1,5),' ') ,.
        IF IRB LT 2 THEN GOTO GETPAG ,.     /* NO NUMBER FOUND          */
        C=SUBSTR(B,IRX+1,IRB-1) ,.          /* SET UP CONVERSION TRY    */
        NOP=NP ,.                   /* SAVE PAGE NUMBER FOR ERROR CHECK */
        GET STRING(C) EDIT(NP) (F(IRB-1)) ,.          /* ATTEMPT CONV   */
        L=(IRX+1)/100 ,.            /* CONVERSION ATTEMPT SUCCEEDED, SO */
        NOTELIST(L)=NOTELIST(L) CAT '*' ,. /* NOTE PAGE ENTRY IN MRG   */
        DO K=L TO 36 ,.
        IPAGES(K)=NP ,.
        END ,.
        IF IOPT(14) THEN IF NP LT 1 OR (NP LT NOP) THEN
        CALL ERRMSG('FUNP',IRX+1,2,SMOD,'1'B) ,.
        SUBSTR(A,IRX,1)='P' ,.
        GOTO GETPAG ,.
    GOTPAG ..
        IF IOPT( 7) THEN
        PUT FILE(SYSPRINT) SKIP(1) EDIT
         ((ERRFILE(L/100),L/100,SUBSTR(B,L,100),
        IPAGES(L/100),NOTELIST(L/100),
        SUBSTR(A,L,100),( K   DO K=1 TO 9)
        DO L=1 TO I BY 100))
       (SKIP(1),A(4),F(3),X(3),A(100),F(6),X(1),A(3),
        SKIP(1),COL(11),A(100),SKIP(1),COL(11), 9 (X(9),F(1)))  ,.
        P='' ,.
        IRX=1 ,.                    /*   PURE & INDX EDITING & OUTPUT   */
    EDITOUT ..
        IF IRX=I THEN GOTO PUTOUT ,.        /* FINISHED, WRITE OUT      */
        ILX=IRX ,.      /* NOW POINTS TO 1ST BYTE OF NEW AREA           */
        SMOD=SUBSTR(A,ILX,1) ,.             /* GET AREA TYPE            */
        IF SMOD='P' THEN DO ,.  /* IF IT'S A PAGE CHANGE ENTRY          */
        IRX=INDEX(SUBSTR(B,ILX+1,5),' ')+ILX ,.      /* FIND #'S END */
        C=SUBSTR(B,ILX+1,IRX-ILX-1) ,.      /* SET UP FOR CONVERSION    */
        IOP=NP ,.                   /* SAVE PAGE NO. FOR ERR CHK        */
        GET STRING(C) EDIT(NP) (F(IRX-ILX-1)) ,.     /* ATTEMPT CONV */
        PUT STRING(CHPAGE) EDIT(NP) (F(4)) ,.
        P=P CAT SUBSTR(B,ILX-1,IRX-ILX+2) ,.
        IRX=IRX+1 ,.
        GOTO EDITOUT ,.
        END ,.
        IRX=VERIFY(SUBSTR(A,ILX+1),SMOD) ,.          /* SET RELATIVE */
                    /* POINTER TO 1ST BYTE OF NEXT AREA, IF ANY        */
        IF IRX=0 THEN IRX=I ,.  /* LAST AREA FOUND                      */
        ELSE IRX=IRX+ILX ,.     /* NOT LAST AREA                        */
        IF SMOD=' ' THEN DO ,.  /* BLANK (UNDERLINING STRUCTRD DATA */
        P=P CAT SUBSTR(B,ILX,IRX-ILX) ,.
        END ,.
        ELSE IF SMOD='I' THEN DO ,.
        C2=SUBSTR(B,IRX-1,1) ,.
        IF C2 NE ' ' THEN DO ,.  IRX=IRX-1 ,.  SMOD=C2 ,.  END ,.
```

```
      C=SMOD CAT SUBSTR(B,ILX+1,IRX-ILX-2) ,.
      IF SMOD NE 'I' THEN IRX=IRX+1 ,.
      IF LENGTH(C) LT 20 THEN DO ,.      /* PAD FOR SORTING.       */
      CH14=C ,.
      C=CH14 ,.
      END ,.
      IF IOPT(17) THEN
      PUT FILE(SYSPRINT) SKIP(1) LIST(C) ,.
      WRITE FILE(INDX) FROM(C) ,.
      END ,.
      GOTO EDITOUT ,.
PUTOUT ..
      IF IOPT( 8) THEN
      PUT FILE(SYSPRINT) SKIP(1) DATA(P) ,.
      WRITE FILE(PURE) FROM(P) ,.
      GOTO INPUT ,.
      END ,.
      IX=ISR ,.                  /* SAVE INDEX TO CALCULATE NEW ISR  */
      ISR=INDEX(SUBSTR(B,ISR+1),' |') ,. /* FIND NEXT BLNK-PRFX      */
      IF ISR=0 THEN DO ,. /* SET UP LAST SECTION AND PROCESS IT      */
      ISR=I-2 ,.
      GOTO DOLASTSEC ,.
      END ,.
      ISR=ISR+IX ,.  /* SET PTR TO ACTUAL LOC OF RIGHT SEC-DELIM     */
      IF VERIFY(SUBSTR(B,ISR+2,1),EBSML) THEN GOTO NEXTSECPART ,.
          /* TRY AGAIN IF THIS IS NOT THE END OF THE SECTION         */
DOLASTSEC ..
      SMOD=SUBSTR(B,ISL+2,1) ,.      /* OBSERVE TYPE OF THE SECTION */
                    /* ABOVE STMT MARKS SECTION START IN MAP MRGN */
      NOTELIST(ISR/100)=NOTELIST(ISR/100) CAT SUBSTR(B,ISR+2,1) ,.
      IF SMOD=EBC THEN     /* IF IT'S A COMMENTS SECTION THEN . . . */
           DO ,.
      ILB=ISL+2 ,.
      IRB=ISR-2 ,.
      DO L=ISL TO ISR-1 ,. SUBSTR(A,L,1)='-' ,.  END ,.
      INDXRTRN=NEXTSEC ,.
      GOTO NEXTIND ,.
           END ,.
      IRB=ISL+2 ,.
NEXTBRKTS ..
      ILB=INDEX(SUBSTR(B,IRB+1,ISR-IRB-1), EBLBRKT) ,.
      IF ILB=0 THEN GOTO NEXTSEC ,.
      ILB=ILB+IRB ,.
      IRB=INDEX(SUBSTR(B,ILB+1,ISR-ILB-1),EBRBRKT) ,.
      IF IRB=0 THEN DO ,.
      IF IOPT(11) THEN
      PUT SKIP(1) FILE(SYSPRINT) EDIT
      ('ERROR-  NO RIGHT BRACKET, LEFT IS AT',ILB) (R(H)) ,.
      CALL ERRMSG('NORB',ILB,1,SMOD,'1'B) ,.
      IRB=ISR-ILB ,.       /* EXTEND 'BRACKET' TO END OF SECTION     */
                           /* INDEXES SHOULD GO OK, SORRY IF ANY     */
                           /* STRUCTURED DATA IS LOST                */
      END ,.
      IRB=IRB+ILB ,.
      IF SUBSTR(B,ILB-1,1)='|' THEN ILB=ILB-1 ,.
      DO L=ILB TO IRB ,. SUBSTR(A,L,1)='X' ,.   END ,.
      INDXRTRN=NEXTBRKTS ,.
NEXTIND ..
      ILX=INDEX(SUBSTR(B,IRX+1,IRB-IRX-1),'|') ,.
      IF ILX=0 THEN GOTO INDXRTRN ,.     /* EXIT-RETURN FROM INDXN */
      ILX=ILX+IRX ,.
      CH3=SUBSTR(B,ILX-1,3) ,.           /*   SET UP C1 AND C2     */
      IF VERIFY(C1,EBSML)=1 THEN BIT1='1'B ,. ELSE BIT1='0'B ,.
      IF C2= ' ' AND NOT BIT1 OR C1=EBLBRKT OR C1=EBRBRKT THEN
      DO ,.          /* SO THIS WASN'T THE START OF AN INDEX AFTER */
      IRX=ILX ,.     /* ALL, BUT RESET THE SEARCH POINTER AND      */
```

```
        GOTO NEXTIND ,.          /* KEEP SEARCHING FOR START OF INDX */
        END ,.
        IF ILX LE ILB THEN DO ,.
        PUT FILE(SYSPRINT) SKIP(1) EDIT
        ('INDEX OUTSIDE OF BRACKETS STARTING AT',ILX) (R(H)) ,.
        CALL ERRMSG('IXOB',ILX,1,SMOD,'1'B) ,.
        END ,.
        IRX=INDEX(SUBSTR(B,ILX+1,IRB-ILX-1),'|') ,.
        IF IRX=0 THEN DO ,.
        IRX=INDEX(SUBSTR(B,ILX+1,IRB-ILX-1),'/') ,.
        IF IRX=0 THEN DO ,.
        PUT FILE(SYSPRINT) SKIP(1) EDIT
        ('ERROR - NO MATCHING INDEX CLOSER, ILX=',ILX,', IRB=',IRB)
        ( 2 (R(H))) ,.
        CALL ERRMSG('NORX',ILX,1,SMOD,'1'B) ,.
        IRX=ILX ,.
        GOTO INDXRTRN ,.
        END ,.
        IRX=IRX+ ILX ,.
        GOTO VERISMALL ,.
        END ,.
        IRX=IRX+ ILX ,.
        CH3=SUBSTR(B,ILX-1,3) ,.          /*   SET UP C1 AND C2      */
        IF VERIFY(C1,EBSML)=1 THEN BIT1='1'B ,. ELSE BIT1='0'B ,.
        IF C2=' ' AND NOT BIT1 OR C1=EBRBRKT  OR C1=EBLBRKT THEN
        DO ,.
        PUT FILE(SYSPRINT) SKIP(1) EDIT
        ('CAUGHT INDEX CROSSING SECTION OR BRACKET AT',IRX,
        ', IRB='          ,IRB) (A,F(4),A,F(4))
        (', INDEX BEGINS AT',ILX) (A,F(4)) ,.
        CALL ERRMSG('ICSB',IRX,1,SMOD,'1'B) ,.
        GOTO INDXRTRN ,.
        END ,.
    VERISMALL ..
        IF VERIFY(SUBSTR(B,IRX+1,1),EBSMQ) THEN      /* INDEX DID    */
                                /* END WITH A SMALL LETTER OR QUOTE */
        DO ,.
        PUT FILE(SYSPRINT) SKIP(1) EDIT
        ('BAD END OF INDEX VERIFIED AT',IRX, ', INDEX STARTED AT',ILX)
        ( 2 (R(H))) ,.
        CALL ERRMSG('BNDX',IRX,1,SMOD,'1'B) ,.
        END ,.
        DO L=ILX TO IRX+1 ,. SUBSTR(A,L,1)='I' ,. END ,.
        IF SUBSTR(B,IRX,1)='/' THEN GOTO NEXTIND ,.
        SUBSTR(B,IRX,1)=SUBSTR(B,IRX+1,1) ,.
        SUBSTR(B,IRX+1,1)=' ' ,.          /* FOR NEATNESS          */
        GOTO NEXTIND ,.
    ERRMSG .. PROCEDURE(ABBRV,LOC,ITYPE,SECTYPE,PRINTOPT) ,.
        DCL ABBRV CHAR(120) VARYING,
        LOC FIXED BIN(15,0),
        PRINTOPT BIT(1),
        ITYPE FIXED BIN(15,0),
        MSG(2) CHAR(8) VARYING INITIAL('','POSSIBLE'),
        SECTYPE CHAR(1) VARYING ,.
        IF IOPT(12) THEN IF PRINTOPT THEN
        PUT FILE(SYSPRINT) SKIP(2) EDIT
        (MSG(ITYPE),' ERROR **** ',ABBRV,' IN ',SECTYPE,' LOC=',LOC)
        (SKIP(2),6 A,F(5)) ,.
        ERRFILE(LOC/100)=ABBRV ,.
        END ERRMSG ,.
        ENDPROG .. END SAVEDIT ,.
/*
//LSPMAIL  JOB 99999211,WCD,MSGLEVEL=(1,1)
// EXEC PL1LFG,PARM='NOPRINT'
//GO.SYSLIB DD
//       DD DSN=SYS1.SORTLIB,DISP=SHR
```

```
//GO.SYSLIN DD UNIT=2311,VOL=SER=GWSJR1,DISP=OLD,DSN=LSPMAIL
//GO.SORTLIB DD DSN=SYS1.SORTLIB,DISP=SHR
//GO.SYSPNCH DD SYSOUT=B,DCB=BLKSIZE=80
//GO.SORTWK01 DD UNIT=2311,SPACE=(TRK,20),DSN=&&WK1,DISP=NEW
//GO.SORTWK02 DD UNIT=2311,SPACE=(TRK,20),DSN=&&WK2,DISP=NEW
//GO.SORTWK03 DD UNIT=2311,SPACE=(TRK,20),DSN=&&WK3,DISP=NEW
//GO.SYSIN DD *
01111111111111111111111111111111111111111111111111111111111111111111111111110001
00111111111111111111111111111111111111111111111111111111111111111111111110001
ABOU SAIF/MRS LEILA                                 LAWRENCE*
ABRAHAM/MR DAVID B.                                 1105 WEST SUMMER STREET
APPLETON, WISCONSIN  54911*
AMERICAN TYPE FOUNDERS CO. INCORPORATED             200 ELMORA AVENUE
ELIZABETH , NEW JERSEY, 07207*
ARITHMOI/                                           PROFESSOR RICHARD E. WHITAKER
CENTRAL COLLEGE                                     PELLA, IOWA 50219*
ARMENTI/DR AMEDIO                                   M.I.T., LINCOLN LABORATORY
LEXINGTON, MASSACHUSETT  02173*
ARMSTRONG/PROFESSOR WILLIAM                         WESTFIELD COLLEGE
UNIVERSITY OF LONDON                                LONDON, ENGLAND*
ARNOTT/PROFESSOR J. F.                              UNIVERSITY OF GLASGOW
GLASGOW, SCOTLAND*
ASHLEY/PROFESSOR L. R. N.                           DEPARTMENT OF ENGLISH
BROOKLYN COLLEGE                                    CITY UNIVERSITY OF NEW YORK
BROOKLYN, NEW YORK  11210*
ASPREY/PROFESSOR WINIFRED                           DEPARTMENT OF MATHEMATICS
VASSAR COLLEGE                                      POUGHKEEPSIE, NEW YORK*
AUBURN/MR MARK S.                                   5532 SOUTH DORCHESTER
CHICAGO, ILLINOIS  60637*
AUBURN/MR. MARK S.                                  59 WEST SCHREYER
COLUMBUS, OHIO 43214*
AUTOMATIC DATA PROCESSING INCORPORATED/             1330 PEACHTREE STREET
ATLANTA, GEORGIA 30309*
AUTOMATIC DATA PROCESSING INCORPORATED/             405 ROUTE 3
CLIFTON, NEW JERSEY 07015*
AVERY/MRS EMMETT                                    108 NORTH FOURTH STREET
CHENEY, WASHINGTON  99004*
BACHHUBER/MR JOHN                                   INSTITUTE OF PAPER CHEMISTRY*
BALDWIN/MISS O.                                     MISS T. WILSON
23 MANILLA ROAD                                     BRISTOL  B58 4EE
ENGLAND*
BARKER, HON. SECRETARY/MISS KATHLEEN                SOCIETY FOR THEATRE RESEARCH
14 WORONZOW ROAD                                    LONDON, N. W. 8, ENGLAND*
BEDFORD/PROFESSOR EMMETT G.                         DEPARTMENT OF ENGLISH
UNIVERSITY OF WISCONSIN-PARKSIDE                    KENOSHA, WISCONSIN  53140*
BENDER/PROFESSOR TODD                               DEPARTMENT OF ENGLISH
UNIVERSITY OF WISCONSIN                             MADISON, WISCONSIN  53706*
BENEFICIAL INSURANCE GROUP/                         3700 WILSHIRE BLVD.
LOS ANGELES, CALIFORNIA 90005*
BERANEK/MR LEO                                      LEDGEWOOD ROAD
WINCHESTER, MASSACHUSETTS  01890*
BEST/MR RICHARD                                     DIGITAL EQUIPMENT COMPANY
MAYNARD, MASSACHUSETTS  01754*
BLACKLOCK/MR P.E.N.                                 COMPUTER SERVICES CENTRE LTD
STATION HOUSE                                       STONEBRIDGE PARK
WEMBLEY, MIDDLESEX                                  ENGLAND*
BOJARSKI/PROFESSOR E. A.                            MCMURRAY COLLEGE
ABILENE, TEXAS*
BONMARIAGE, RESEARCH ASSISTANT/MR J.                BELGIAN ARCHIVES FOR THE SOCIAL SCIENCES
VAN EVENSTRAAT, 2A, ROOM 04-45                      3000 LOUVAIN, BELGIUM*
BORMAN/DR LORRAINE                                  VOGELBACK COMPUTING CENTER
NORTHWESTERN UNIVERSITY                             EVANSTON, ILLINOIS*
BOTSKO/MR EDWARD S.                                 MARKETING REPRESENTATIVE
DISSLY SYSTEMS CORPORATION                          EASTON, PENNSYLVANIA  18042*
BOWLES/DR. EDMUND                                   GEM REGION HEADQUARTERS
INTERNATIONAL BUSINESS MACHINES                     10401 FERNWOOD ROAD
```

BETHESDA, MARYLAND 20034*	
BOWLES/MR EDMUND	CONSULTANT FOR HUMANITIES
INTERNATIONAL BUSINESS MACHINES	1040 FERNWOOD ROAD
BETHESDA, MARYLAND 20034*	
BOWMAN/PROFESSOR NED	DEPARTMENT OF DRAMA
UNIVERSITY OF PITTSBURGH	PITTSBURGH, PENNSYLVANIA*
BOYER/MRS MILDRED	RESOURCE SPECIALIST, SOUTHWEST
EDUCATIONAL DEVELOPMENT LABORATORY	800 BRAZOS STREET
AUSTIN, TEXAS 78701*	
BROWN/MR CURTIS	INSTITUTE OF PAPER CHEMISTRY*
BROWN/MR WALTER	LAWRENCE*
BROWN, ASSISTANT DIRECTOR/MRS M. K.	COMPUTER APPLICATIONS CENTER
SKIDMORE COLLEGE	SARATOGA SPRINGS, NEW YORK 12866*
BRUSTEIN/MR ROBERT	UNIVERSITY THEATRE
222 YORK STREET	NEW HAVEN, CONN.*
BUCHANAN/MR WILLIAM E.	APPLETON WIRE WORKS CORPORATION
714 EAST HANCOCK STREET	APPLETON, WISCONSIN 54911*
BURNIM/PROFESSOR KALMAN A.	DRAMA DEPARTMENT
TUFTS UNIVERSITY	MEDFORD, MASSACHUSETTS 02155*
CAPITAL CITIES, LTD. /	LONDON, ENGLAND*
CASH/PROFESSOR ARTHUR	DEPARTMENT OF ENGLISH
NEW YORK STATE UNIVERSITY COLLEGE	NEW PALTZ, NEW YORK 12561*
CENTURY RESEARCH	25039 NARBONNE
LOMITA, CALIFORNIA 90717*	
CHAN/MR JOHN	47 MACARTHUR ROAD
WELLESLEY, MASSACHUSETTS*	
CHRISMAN/MRS DONALD	21 FRANKLIN STREET
NORTHAMPTON, MASSACHUSETTS 01060*	
CHURCH/MR JOHN	INSTITUTE OF PAPER CHEMISTRY*
CLIFFORD/PROFESSOR JAMES L.	THE JONSONIAN NEWSLETTER
PHILOSOPHY HALL	COLUMBIA UNIVERSITY
NEW YORK, NEW YORK 10027*	
COLEMAN, CHAIRMAN/PROF WILLIAM S. E.	DEPARTMENT OF THEATRE ARTS AND SPEECH
DRAKE UNIVERSITY	DES MOINES, IOWA 50311*
COMMERCIAL UNION ASSURANCE CO LTD/	431 GODSTONE
WHYTELEAF, SURREY, ENGLAND*	
CONNELL/MR PHILIP R.	DIRECTOR OF INFORMATION SYSTEMS
ADVANCED COMPUTER SYSTEMS, INC.	3131 SOUTH DIXIE DRIVE
DAYTON, OHIO 45439*	
CONOLLY/PROFESSOR L. W.	DEPARTMENT OF ENGLISH
THE UNIVERSITY OF ALBERTA	EDMONTON 7, CANADA*
CULBERTSON/MR DON S.	EXECUTIVE SECRETARY
AMERICAN LIBRARY ASSOCIATION	50 EAST HURON STREET
CHICAGO, ILLINOIS 60611*	
DALAND/MR WILL	LAWRENCE UNIVERSITY
APPLETON, WISCONSIN 54911*	
DALE/PROFFESSOR A. G.	DEPARTMENT OF COMPUTER SCIENCES
UNIVERSITY OF TEXAS AT AUSTIN	AUSTIN, TEXAS 78712*
DEHART/DR FLORENCE E.	DEPARTMENT OF LIBRARIANSHIP
KANSAS STATE TEACHERS COLLEGE	EMPORIA, KANSAS 66801*
DEHART/MISS FLORENCE E.	SCHOOL OF LIBRARY
AND INFORMATION SCIENCE	UNIVERSITY OF WESTERN ONTARIO
LONDON 72 ONTARIO*	
DISSLY SYSTEMS CORPORATION/	CHURCH AND BANK STREETS
EASTON, PENNSYLVANIA, 18042*	
DOLLAR/PROFESSOR CHARLES M.	DEPARTMENT OF HISTORY
OKLAHOMA STATE UNIVERSITY	STILLWATER, OKLAHOMA 74074*
DONOW/PROFESSOR HERBERT S.	DEPARTMENT OF ENGLISH
SOUTHERN ILLINOIS UNIVERSITY	CARBONDALE, ILLINOIS 62901*
DOWDY/MR GERALD	MARKETING GROUP
INFORMATION CONTROL INC.	ONE GATEWAY CENTER, 5TH AT STATE
KANSAS CITY, KANSAS 66101*	
DOWNER/PROFESSOR ALAN	DEPARTMENT OF ENGLISH
PRINCETON UNIVERSITY	PRINCETON, NEW JERSEY 08540*
DUO COMPUTER CORPORATION/	1925 BELLMORE AVENUE
BELLMORE, NEW YORK 11710*	

EDUCATION/CENTER FOR HISTORY OF — UNIVERSITY OF TEXAS
AUSTIN, TEXAS 78712*

EMERSON/MR WILLIAM R. — NATIONAL ENDOWMENT FOR THE HUMANITIES,
ROOM 2329, GENERAL SERVICES BUILDING — 1800 F STREET N.W.
WASHINGTON, D.C.*

EVERSMAN/MR BILL — DELTA DATA SYSTEMS
9501 WEST DEVON AVENUE — ROSEMONT, ILLINOIS 60018*

FIFTH AT STATE — KANSAS CITY , KANSAS 66101*

FISHER/MR JOHN HURT — EXECUTIVE SECRETARY
MODERN LANGUAGE ASSOCIATION — 62 FIFTH AVENUE
NEW YORK, NEW YORK 10011*

FLOOD/PROFESSOR BARBARA — GRADUATE SCHOOL OF LIBRARY SCIENCE
DREXEL UNIVERSITY — 33RD STREET AND LANCASTER AVENUE
PHILADELPHIA, PENNSYLVANIA 19104*

FORD, CHAIRMAN/PROFESSOR GEORGE — DEPARTMENT OF ENGLISH
UNIVERSITY OF ROCHESTER — ROCHESTER, NEW YORK 14627*

FORMSCAN INCORPORATED/ — 16620 ORANGE AVENUE
PARAMOUNT, CALIFORNIA 90723*

FRENCH, SYSTEMS ENGINEER/HIRAM T. — IMLAC CORPORATION
296 NEWTON STREET — WALTHAM, MASSACHUSETTS 02154*

FREUND/MR HUGH — SCAN DATA CORPORATION
254 WEST 31ST STREET — NEW YORK, NEW YORK 10001*

FRIEDMAN/MRS MURIEL — 4300 MARINE DRIVE
CHICAGO, ILLINOIS 60613*

GOLDEN/MR RICHARD — ACLS CONSULTANT ON INFORMATION RETRIEVAL
345 EAST 46TH STREET — NEW YORK, NEW YORK 10017*

GOODWIN/MR JIM — BOLT, BERANEK AND NEWMAN
CAMBRIDGE, MASSACHUSETTS*

GROW/MR BYRON — INSTITUTE OF PAPER CHEMISTRY*

HALL/SIR JULIAN — THE GARRICK CLUB
GARRICK STREET — LONDON, ENGLAND*

HAMILTON/MR THOMAS — SEA, INC.
4210 WEST IRVING PARK ROAD — CHICAGO, ILLINOIS 60641*

HARDT, MARKETING REPRESENTATIVE/MR J. — INTERNATIONAL BUISNESS MACHINES CORP.
424 SOUTH MONROE AVENUE — GREEN BAY, WISCONSIN 54301*

HAYWARD/PRESIDENT SUMNER — ASSOCIATED COLLEGES OF THE MIDWEST
60 WEST WALTON STREET — CHICAGO, ILLINOIS 60610*

HEADRICK/MR THOMAS — VICE PRESIDENT FOR ACADEMIC AFFAIRS
LAWRENCE*

HEINEMANN/MISS MARCIA — 10A LONSDALE SQUARE
LONDON N. 1 — ENGLAND*

HELLER/PROFESSOR JACK — COMPUTER CENTER
STATE UNIVERSITY OF NEW YORK — AT STONY BROOK
LONG ISLAND, NEW YORK*

HERNDON/MR WILLIAM — SCAN-DATA CORPORATION
DETROIT DATA CENTER — SUITE 101
32500 DOLLY MADISON DRIVE NORTH — MADISON HEIGHTS, MICHIGAN 48071*

HIGHFILL/PROFFESSOR PHILIP — DEPARTMENT OF ENGLISH
GEORGE WASHINGTON UNIVERSITY — WASHINGTON, D. C. 20006*

HO/MR HARRY — HONG KONG DATA PROCESSING COMPANY, LTD.
THAI KONG BUILDING, 21ST FLOOR — 482 HENNESSEY ROAD
HONG KONG*

HODGE, EDITOR/MISS FRANCES — EDUCATIONAL THEATRE JOURNAL
AMERICAN EDUCATIONAL THEATRE ASSOCIATION — 726 JACKSON PLACE, N. W.
WASHINGTON, D. C. 20566*

HOGAN/MR CHARLES BEECHER — RACEBROOK ROAD
WOODBRIDGE, CONNECTICUT*

HOGAN/MR CHARLES BEECHER — 85 NORTH RACEBROOK ROAD
WOODBRIDGE,CONNECTICUT 06525*

HOLLAND/PROFESSOR NORMAN N. — DEPARTMENT OF ENGLISH
STATE UNIVERSITY OF NEW YORK AT BUFFALO — BUFFALO, NEW YORK 14214*

HOOK/EMERITUS PROFESSOR LUCYLE — C/O DEPARTMENT OF ENGLISH
BARNARD COLLEGE — COLUMBIA UNIVERSITY
NEW YORK, NEW YORK 10027*

HOPPENSPERGER/MR JOE — LAWRENCE*

HOPKINSON/MR LEE — LAWRENCE*

HOROWITZ/PROFESSOR FLOYD R.
UNIVERSITY OF KANSAS
DEPARTMENT OF ENGLISH
LAWRENCE, KANSAS 66044*

HOWARD/MR SHERWIN
APPLETON, WISCONSIN 54911*
SAGE COTTAGE , LAWRENCE UNIVERSITY

HUGHES/PROFESSOR LEO
UNIVERSITY OF TEXAS
DEPARTMENT OF ENGLISH
AUSTIN, TEXAS 78712*

HUI/MR PHILIP T.Y.
21-F THAI KONG BUILDING
HONG KONG*
HONG KONG DATA PROCESSING CO. LTD.
482 HENNESSY ROAD

HUNTER/PROFESSOR GEORGE
UNIVERSITY OF WARWICK
DEPARTMENT OF ENGLISH
COVENTRY, ENGLAND*

INFORMATION CONTROL INCORPORATED/
1 GATEWAY CENTER

NAGLER/MR. A.M.
222 YORK STREET
UNIVERSITY THEATRE
NEW HAVEN, CONN.*

INGLE, QUALITY CONTROL ENGINEER/MR SUD
FOND DU LAC, WISCONSIN 54935*
KIEKHAEFER MERCURY

JOHNSON/MRS JEAN
FIFTH AT STATE
TALLMAN-FOBBINS, INCORPORATED
KANSAS CITY , KANSAS 66101*

KNIGHT, DIV. VICE PRES./MR DOUGLAS M.
30 ROCKEFELLER PLAZA
EDUCATIONAL DEVELOPMENT, RCA
NEW YORK, NEW YORK 10020*

KNOWLER, DIRECTOR/MR D. L.
CORPORATION S
DALLAS, TEXAS 75207*
GRAPHIC ARTS OPTIMATION SERVICES
2600 STEMMONS FREEWAY, SUITE 176

KOCH, CHAIRMAN/PROFESSOR PHILIP
LANGUAGES AND LITERATURES
PITTSBURGH, PENNSYLVANIA 15213*
DEPARTMENT OF FRENCH AND ITALIAN
UNIVERSITY OF PITTSBURGH

KOSKINEN/MR DONALD
MENASHA, WISCONSIN 54952*
GEORGE BANTA COMPANY, INC.

LANGHANS/PROFESSOR EDWARD A.
UNIVERSITY OF HAWAII
DEPARTMENT OF DRAMA
HONOLULU, HAWAII 96822*

LAWLOR, CHAIRMAN/PROFESSOR JOHN
UNIVERSITY OF KEELE
DEPARTMENT OF ENGLISH
STAFFORDSHIRE, ENGLAND*

LEFF/DR. LEONARD
DE KALB, ILLINOIS 60115*
620 LOCUST STREET

LEVINE/PROFESSOR JONATHAN
UNIVERSITY OF PITTSBURGH
DEPARTMENT OF HISTORY
PITTSBURGH, PENNSYLVANIA 65213*

LINCOLN/MR STODDARD
NEW YORK, NEW YORK 10024*
127 NORTH 87TH STREET

LISCANO/MR GILBERT
COMPUSCAN
TETERBORO NEW JERSEY 07608*
SALES PRODUCT MANAGER
900 HUYLER STREET

LITTLEWOODS POOLS LTD./
LIVERPOOL, 11
WALTON HALL AVENUE
ENGLAND*

LITTO/PROFESSOR FREDERICK M.
UNIVERSITY OF KANSAS
DEPARTMENT OF SPEECH AND DRAMA
LAWRENCE, KANSAS 66044*

LOFTIS/PROFESSOR JOHN
STANFORD UNIVERSITY
DEPARTMENT OF ENGLISH
STANFORD, CALIFORNIA 94305*

LORD/MR CHARLES
HALES CORNERS, WISCONSIN 53130*
5551 SOUTH BONNIE LANE

LYKOS/PROFESSOR PETER
ILLINOIS INSTITUTE OF TECHNOLOGY
INFORMATION SCIENCE CENTER
CHICAGO, ILLINOIS 60616*

MAIHOFER/MR RICHARD F.
SCAN-DATA CORPORATION
NORRISTOWN, PENNSYLVANIA 19401*
WESTERN REGIONAL MANAGER
800 EAST MAIN STREET

MANN/PROFESSOR DAVID
MIAMI UNIVERSITY
DEPARTMENT OF ENGLISH
OXFORD, OHIO 45056*

MARDER, EDITOR/PROFESSOR LOUIS
UNIVERSITY OF ILLINOIS
CHICAGO, ILLINOIS 60680*
THE SHAKESPEARE NEWSLETTER
AT CHICAGO CIRCLE

MARTIN/PROFESSOR PETER
FLORIDA ATLANTIC UNIVERSITY
ENGLISH DEPARTMENT
BOCA RATON, FLORIDA 33432*

MARTZ/MRS LOUIS
YALE UNIVERSITY
331B STERLING MEMORIAL LIBRARY
NEW HAVEN, CONN.*

MAYER, III/PROFESSOR DAVID
LONDON, S. W. 1
3, HEADFORT PLACE
ENGLAND*

MC CLENAHAN/MR JAMES
INSTITUTE OF PAPER CHEMISTRY*

MCINTYRE/MR JACK — SCAN-DATA CORPORATION
800 EAST MAIN STREET — NORRISTOWN, PENNSYLVANIA 19401*
MCMULLEN/MR. FRANK — UNIVERSITY THEATRE
222 YORK STREET — NEW HAVEN, CONN.*
MERCHANT/MR. DONALD E. — MEAD DATA CENTRAL, INC.
1254 JEFFERSON DAVIS HWY. — ARLINGTON, VA. 22202*
MILIC/PROFESSOR LOUIS — DEPARTMENT OF ENGLISH
CLEVELAND STATE UNIVERSITY — CLEVELAND, OHIO 44115*
MILLER/ MR. J. PHILIP — COMPUTING FACILITIES, BOX 1152
WASHINGTON UNIVERSITY — SAINT LOUIS, MO. 63130*
MOLNAR/DR. ANDREW — NATIONAL SCIENCE FOUNDATION
WASHINGTON D.C. 20550*
MONTERIO/MR FRED — HAZELTINE CORPORATION
LITTLE NECK, NEW YORK 11362*
MORGAN/MR RICHARD S. — INTERNATIONAL DATA HIGHWAYS LTD.
141 GT. CHARLES STREET — BIRMINGHAM B3 3LG,ENGLAND*
NANCE/DR RICHARD E. — COMPUTER SCIENCE CENTER
SOUTHERN METHODIST UNIVERSITY — DALLAS, TEXAS*
NEFF/PROFESSOR AND MRS EMERY — WESTMORELAND, NEW HAMPSHIRE 03467*
NICOLL/PROFESSOR ALLARDYCE — UNIVERSITY OF BIRMINGHAM
BIRMINGHAM, ENGLAND*
NICOLL/PROFESSOR ALLARDYCE — WINDS ACRE
COLWALL, NEAR MALVERN — WORCESTER, ENGLAND*
OAKLEY/MRS GARY — COMPUTER RESOURCES LTD.
307 LESMILL ROAD — DON MILLS, ONTARIO, CANADA*
POLLIN/PROFESSOR ALICE — DEPARTMENT OF SPANISH
NEW YORK UNIVERSITY — THE BRONX, NEW YORK 10468*
POLLIN/PROFESSOR BURTON — DEPARTMENT OF ENGLISH
BRONX COMMUNITY COLLEGE — THE BRONX, NEW YORK 10468*
PRICE/PROFESSOR CECIL — DEPARTMENT OF ENGLISH
UNIVERSITY COLLEGE OF SWANSEA — SINGLETON PARK,
SWANSEA, WALES*
RABEN, EDITOR/DR JOSEPH — COMPUTERS AND THE HUMANITIES
QUEENS COLLEGE — THE CITY UNIVERSITY OF NEW YORK
FLUSHING, NEW YORK 11367*
RAMDASS/MR P. RAJA — PROGRAMMING CONSULTANT
COMPUTER SCIENCES LABORATORY — 1020 WEST JEFFERSON BOULEVARD
LOS ANGELES, CALIFORNIA 90007*
REAGOR/MISS SIMONE — DIVISION OF RESEARCH GRANTS
NATIONAL ENDOWMENT FOR THE HUMANITIES — 1800 F STREET, N.W.
WASHINGTON, D.C.*
REMLE/MR ARTHUR — 219 EAST WISCONSIN AVENUE
NEENAH, WISCONSIN 54956*
RICHARDS/PROFESSOR KENNETH — DEPARTMENT OF DRAMA
UNIVERSITY OF MANCHESTER — MANCHESTER M13 9PL, ENGLAND*
RIEGER/MR FREDERICK — INDUSTRIAL NUCLEONICS CORPORATION
650 ACKERMAN ROAD — COLUMBUS OHIO*
ROACH/MR W. E. — VICE PRESIDENT, SALES
AMERICAN-COMPUTYPE, INC. — 2819 N. HAMLINE AVENUE
ST. PAUL, MINNESOTA 55113*
ROBINSON/MR THEODORE W., JUNIOR — MARKETING ADMINISTRATOR
INFOTON INCORPORATED — SECOND AVENUE
BURLINGTON, MASSACHUSETTS, 01803*
ROBINSON/PROFESSOR J. W. — DEPARTMENT OF ENGLISH
UNIVERSITY OF NEBRASKA — LINCOLN, NEBRASKA 68508*
ROSENFELD/MISS SYBIL — 103 RALPH COURT
QUEENS WAY, LONDON W2 — ENGLAND*
SACON/MR LEONARD D. — IMLAC CORPORATION
296 NEWTON STREET — WALTHAM, MASSACHUSETTS 02154*
SAWIN/PROFESSOR LEWIS, ASSOCIATE DEAN — UNIVERSITY OF COLORADO
BOULDER COLORADO 80302*
SAWYER/PROFESSOR PAUL — DEPARTMENT OF ENGLISH
BRADLEY UNIVERSITY — PEORIA, ILLINOIS 61606*
SCHMIDT/MR CHARLES — INTERNATIONAL BUSINESS MACHINES
NORTH MEADE STREET — APPLETON, WISCONSIN 54911*
SCHMIDT, SALES REPRESENTATIVE/MR CHARLES — INTERNATIONAL BUSINESS MACHINES CORP.

2631 NORTH MEADE STREET
APPLETON, WISCONSIN 54911*

SCHNEIDER/PROFESSOR ELISABETH
DEPARTMENT OF ENGLISH
UNIVERSITY OF CALIFORNIA
SANTA BARBARA, CALIFORNIA 93106*

SCHODER/MR BARRY
SALES MANAGER, ELECTROGRAPHICS
R.R.DONNELLY AND SONS, CO
2223 SOUTH MARTIN LUTHER KING DRIVE
CHICAGO, ILLINOIS 60616*

SCHUETZ/MRS CHRISTIE
ADVERTISING AND PROMOTION MANAGER
THE UNIVERSITY OF ILLINOIS PRESS
URBANA , ILLINOIS 61801*

SCOUTEN/PROFESSOR ARTHUR H.
DEPARTMENT OF ENGLISH
UNIVERSITY OF PENNSYLVANIA
PHILADELPHI, PENNSYLVANIA 19104*

SEALTS/PROFESSOR MERTON M.
DEPARTMENT OF ENGLISH
UNIVERSITY OF WISCONSIN
MADISON, WISCONSIN 53706*

SEDELOW/PROFESSOR SALLY
DEPARTMENT OF LINGUISTICS
UNIVERSITY OF KANSAS
LAWRENCE, KANSAS 66044*

SHIU/MR DAVID
CHINA DATA SYSTEMS
P. O. BOX 7
FDR STATION
NEW YORK, NEW YORK 10022*

STACK/MR WILL
INTERNATIONAL BUSINESS MACHINES
NORTH MEADE STREET
APPLETON, WISCONSIN 54911*

STAFFELBACH/MISS EMMA
CHANTICLEER PRESS, INC.
424 MADISON AVENUE
NEW YORK 17, NEW YORK*

STERNBERG, DIRECTOR/MR VERNON
SOUTHERN ILLINOIS UNIVERSITY PRESS
CARBONDALE, ILLINOIS 62901*

STIFF/MR RON
PROJECT MANAGER
INFORMATION SCIENCE CENTER
ILLINOIS INSTITUTE OF TECHNOLOGY
CHICAGO, ILLINOIS 60616*

STIFF/MR RON, PROJECT MANAGER
INFORMATION SCIENCE CENTER
ILLINOIS INSTITUTE OF TECHNOLOGY
CHICAGO, ILLINOIS 60616*

STONE/PROFESSOR PHILIP J.
DEPARTMENT OF SOCIAL RELATIONS
HARVARD UNIVERSITY
CAMBRIDGE, MASSACHUSETTS*

STONE, JR./PROFESSOR GEORGE WINCHESTER
DEAN, NEW YORK UNIVERSITY LIBRARIES
10 WASHINGTON PLACE, ROOM 401
NEW YORK, NEW YORK 10003*

STRATMAN/CARL J., C.S.V.
DEPARTMENT OF ENGLISH
LEWIS TOWERS, LOYOLA UNIVERSITY
820 NORTH MICHIGAN AVENUE
CHICAGO, ILLINOIS 60611*

STRATMAN, C.S.V./CARL J.
DEPARTMENT OF ENGLISH
LEWIS TOWERS, LOYOLA UNIVERSITY
820 NORTH MICHIGAN AVENUE
CHICAGO , ILLINOIS 60611*

TALLMAN-ROBBINS AND CO. /
909 EAST CAPITAL
SPRINGFIELD, ILLINOIS*

TASCH, EDITOR/MR PETER A.
THE SCRIBLERIAN
TEMPLE UNIVERSITY
PHILADELPHIA, PENNSYLVANIA 19122*

TAYLOR, ACCOUNTS MANAGER/MR TOM A.
CORPORATION S
2600 STEMMONS FREEWAY, SUITE 204
DALLAS, TEXAS 75207*

TECHNIDATA GMBH
BESSER MER STRASSE 91
BERLIN , WEST GERMANY*

TETER/MR GORDON
INDUSTRIAL NUCLEONICS CORPORATION
650 ACKERMAN ROAD
COLUMBUS OHIO*

THE EDITOR
DRAMA SURVEY
THE BOLINGBROKE SOCIETY
BOX 4098
MINNEAPOLIS, MINNESOTA 55414*

THOMPSON/PROFESSOR CRAIG
DEPARTMENT OF ENGLISH
UNIVERSITY OF PENNSYLVANIA
PHILADELPHIA, PENNSYLVANIA 19401*

TRETTIN/MR TOM
LAWRENCE*

TURNER/MR C.G.
SALES MANAGER
THE MONOTYPE CORPORATION LIMITED
SALFORDS,REDHILL
SURREY, ENGLAND*

TURNER/MR DONALD G.
GEORGE BANTA COMPANY, INC.
MENASHA, WISCONSIN 54952*

VEITH/PROFESSOR DAVID M.
DEPARTMENT OF ENGLISH
SOUTHERN ILLINOIS UNIVERSITY
CARBONDALE, ILLINOIS 62901*

VENEZKY/PROFFESSOR RICHARD L.
CENTER FOR COGNITIVE LEARNING
THE UNIVERSITY OF WISCONSIN
1404 REGENT STREET
MADISON, WISCONSIN 53706*

VERNON/PROFESSOR PAUL
DEPARTMENT OF ENGLISH
UNIVERSITY OF LONDON, KINGS COLLEGE
STRAND , W. C. 2.

```
LONDON, ENGLAND*
WACHAL/PROFFESSOR ROBERT S.                DEPARTMENT OF LINGUISTICS
UNIVERSITY OF IOWA                         IOWA CITY, IOWA 52240*
WAITE/PROFESSOR STEPHEN                    KIEWIT CENTER
DARTMOUTH COLLEGE                          HANOVER, NEW HAMPSHIRE 03775*
WALKER/PROFESSOR EDWARD A.                 DEPARTMENT OF FRENCH
VICTORIA COLLEGE                           UNIVERSITY OF TORONTO
TORONTO 5, CANADA*
WALLACE/MISS IDA                           1 DUPONT CIRCLE
WASHINGTON, D. C.*
WEINER, CHAIRMAN/DR PETER                 COMPUTER SCIENCES DEPARTMENT
408A DENHAM LABRATORY                      10 HILLHOUSE AVENUE
NEW HAVEN, CONNECTICUT*
WELLS, DIRECTOR/MR JAMES M.                THE NEWBERRY LIBRARY
60 WEST WALTON STREET                      CHICAGO, ILLINCIS  60610*
WICKHAM/PROFESSOR GLYNNE                   DEPARTMENT OF DRAMA
UNIVERSITY OF BRISTOL                      BRISTOL, ENGLAND*
WILLIAMS/PROFESSOR AUBREY                  DEPARTMENT OF ENGLISH
UNIVERSITY OF FLORIDA                      GAINSVILLE, FLORIDA  32601*
WILLIAMS/PROFESSOR WILLIAM P.              DEPARTMENT OF ENGLISH
NORTHERN ILLINOIS UNIVERSITY               DEKALB, ILLINCIS  60115*
WILSON/PROFESSOR JOHN HAROLD               DEPARTMENT OF ENGLISH
OHIO STATE UNIVERSITY                      COLUMBUS, OHIO  43210*
WRIGHT II/MR STEWART                       DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASSACHUSETTS  01754*
WROLSTAD/MR MARWIN O.                      LAWRENCE*
YANG/DR SHOU-CHUAN                         DATA AND COMPUTATION CENTER
UNIVERSITY OF WISCONSIN                    MADISON, WISCONSIN*
YOST/MR CHARLES                            MARKETING GROUP
INFORMATION CONTROL INCORPORATED           1 GATEWAY CENTER
FIFTH AT STATE                            -2-52 2 3938# 215SAS  66101*
YOUNG/MR F. STANSBURY                      1611 SOUTH ALICIA DRIVE
APPLETON, WISCONSIN  54911*
ZAMZOW, BRANCH MANAGER/MR CRAIG D.         DIGITAL EQUIPMENT CORPORATION, SUITE 107
2825 NORTH MAYFAIR ROAD                    MILWAUKEE, WISCONSIN  53222*
ZIMBARO/PROFESSOR ROSE                     DEPARTMENT OF ENGLISH
STATE UNIVERSITY OF NEW YORK               AT STONY BROOK
LONG ISLAND, NEW YORK  11790*
OJOSSEM/PROFESSOR HERBERT                  LAWRENCE*
OMITH/PRESIDENT THOMAS                     LAWRENCE UNIVERSITY*
**
/*
         CENPROCS MODEL=65,STORAGE=G,FEATURE=PROTECT
MPX      CHANNEL ADDRESS=0,TYPE=MULTIPLEXOR
SEL1     CHANNEL ADDRESS=1,TYPE=SELECTOR
RMTE     IOCONTRL UNIT=2701,ADDRESS=03
TP0      IODEVICE UNIT=TWX,ADDRESS=030,FEATURE=(AUTOANSR,AUTOCALL),    C
               ADAPTER=TELE2
TP2      IODEVICE UNIT=TWX,ADDRESS=031,FEATURE=(AUTOANSR,AUTOCALL),    C
               ADAPTER=TELE2
TUBES    IOCONTRL UNIT=2848,ADDRESS=02,MODEL=3,FEATURE=NODESCUR
TUBE0    IODEVICE UNIT=2260,ADDRESS=020,MODEL=1,FEATURE=ALKYB2260
TUBE1    IODEVICE UNIT=2260,ADDRESS=021,MODEL=1,FEATURE=ALKYB2260
TUBE2    IODEVICE UNIT=2260,ADDRESS=022,MODEL=1,FEATURE=ALKYB2260
TUBE3    IODEVICE UNIT=2260,ADDRESS=023,MODEL=1,FEATURE=ALKYB2260
TUBE4    IODEVICE UNIT=2260,ADDRESS=024,MODEL=1,FEATURE=ALKYB2260
TUBE5    IODEVICE UNIT=2260,ADDRESS=025,MODEL=1,FEATURE=ALKYB2260
DISKS    IOCONTRL UNIT=2841,ADDRESS=19
DISK1    IODEVICE UNIT=2311,ADDRESS=190
DISK2    IODEVICE UNIT=2311,ADDRESS=191
DISK3    IODEVICE UNIT=2311,ADDRESS=192
RDRPHPT  IOCONTRL UNIT=2821,ADDRESS=00,MODEL=1
PRTR     IODEVICE UNIT=1403,ADDRESS=00E,MODEL=2,FEATURE=UNVCHSET
PNCH     IODEVICE UNIT=2540P,ADDRESS=00D,MODEL=1
RDR      IODEVICE UNIT=2540R,ADDRESS=00C,MODEL=1
TAPES    IOCONTRL UNIT=2803,ADDRESS=18,MODEL=1
```