```
00001000  00000 0   $CONTROL USLINIT,MAP,CODE
00002000  00000 0   $TITLE "MERGE/3000, HP32214B.00.00"
00003000  00000 0   BEGIN
00004000  00000 1
00005000  00000 1
00006000  00000 1
00007000  00000 1
00008000  00000 1
00009000  00000 1   DEFINE VERSION= "HP32214B.00.00 MERGE/3000  "   #;
00010000  00000 1
00011000  00000 1
00012000  00000 1   EQUATE MERGEEXITPARAM=13;      <<EXIT PARAMETER FOR MERGE PROC>>
00013000  00000 1
00014000  00000 1   INTEGER X=X,S0=S-0,S1=S-1;      <<STANDARD KLUDGES>>
00015000  00000 1   LOGICAL DEBUGGING:=FALSE;       <<TRUE IF ENTRY POINT "BUGGER" IS USED>>
00016000  00000 1   ENTRY BUGGER;
00017000  00000 1   DOUBLE POINTER NUMRECSPTR;      <<POINTS TO NUMRECS LOCAL TO MERGE
00018000  00000 1                                    PROCEDURE, SO TRAP CAN GET TO IT>>
00019000  00000 1   EQUATE NUMSTATS=10;             <<# OF WORDS OF STATISTICS>>
00020000  00000 1
00021000  00000 1
00022000  00000 1
00023000  00000 1   INTRINSIC DATELINE,PRINT,ASCII,FREAD,FWRITE,FGETINFO,PRINT'FILE'INFO,
00024000  00000 1             ZSIZE,PROCTIME,TIMER,DEBUG,DASCII,RESETCONTROL,XCONTRAP,
00025000  00000 1             TERMINATE,FOPEN,FRELATE,FCLOSE,DLSIZE,DBINARY,BINARY,INEXT',
00026000  00000 1             FCHECK,FRENAME;
```

```
00028000  00000 1      DEFINE FWD=OPTION FORWARD#;
00029000  00000 1    PROCEDURE MERGETITLE; FWD;
00030000  00000 1    PROCEDURE MERGEERRORMESS(N,MESS,L);
00031000  00000 1        VALUE N;
00032000  00000 1        INTEGER N,L;
00033000  00000 1        BYTE ARRAY MESS;
00034000  00000 1        FWD;
00035000  00000 1    PROCEDURE MERGE(NUMINPUTFILES,INPUTFILES,OUTPUTFILE,KEYSONLY,NUMKEYS,
00036000  00000 1                    KEYS,PREPROCESSOR,POSTPROCESSOR,ERRORPROC,KEYCOMPARE,
00037000  00000 1                    STATISTICS,FAILURE);
00038000  00000 1        VALUE NUMINPUTFILES,OUTPUTFILE,KEYSONLY,NUMKEYS;
00039000  00000 1        INTEGER NUMINPUTFILES,OUTPUTFILE,NUMKEYS;
00040000  00000 1        INTEGER ARRAY INPUTFILES,KEYS,STATISTICS;
00041000  00000 1        LOGICAL KEYSONLY,FAILURE;
00042000  00000 1        PROCEDURE PREPROCESSOR,POSTPROCESSOR,ERRORPROC;
00043000  00000 1        LOGICAL PROCEDURE KEYCOMPARE;
00044000  00000 1        FWD,VARIABLE;
00045000  00000 1    INTEGER PROCEDURE GETERRORMESS(BUF,N);
00046000  00000 1        VALUE N;
00047000  00000 1        INTEGER N;
00048000  00000 1        BYTE ARRAY BUF;
00049000  00000 1        FWD;
00050000  00000 1    PROCEDURE TRAP; FWD;
00051000  00000 1    PROCEDURE MERGEMAIN; FWD;
```

```
00053000  00000 1   $CONTROL SEGMENT=MERGELIB
00054000  00000 1   PROCEDURE MERGETITLE;
00055000  00000 1   <<PRINT THE VERSION AND DATELINE>>
00056000  00000 1
00057000  00000 1   BEGIN
00058000  00000 2      ARRAY W(0:35);
00059000  00000 2      BYTE ARRAY B(*)=W;
00060000  00000 2      BYTE POINTER B';
00061000  00000 2      MOVE B:=VERSION,2; @B':=TOS;
00062000  00034 2      DATELINE(B');
00063000  00036 2      PRINT(W,INTEGER(LOGICAL(@B)-LOGICAL(@B'))-27,"0");
00064000  00045 2   END <<MERGETITLE>>;
```

IDENTIFIER	CLASS	TYPE	ADDRESS
B	ARRAY	BYTE	Q +002
B'	POINTER	BYTE	Q +003
W	ARRAY	LOGICAL	Q +001

```
00000  035004 171700 051401 035043 041401 010201 051402 173402    00010  170003 010201 140017 044120 031462 031061 032102 027060
00020  030056 030060 020115 042522 043505 027463 030060 030040    00030  020040 021033 020042 051403 041403 000000 041401 041402
00040  041403 006100 023033 021060 000000 031400
```

```
00065000  00000 1   PROCEDURE MERGEERRORMESS(N,MESS,L);
00066000  00000 1   VALUE N;
00067000  00000 1   BYTE ARRAY MESS;
00068000  00000 1   INTEGER N,L;
00069000  00000 1
00070000  00000 1   <<MERGEERRORMESS INSERTS INTO THE BYTE ARRAY MESS A STRING
00071000  00000 1     REPRESENTING ERROR NUMBER N. THE NUMBER OF CHARACTERS IS
00072000  00000 1     RETURNED AS A RESULT. IF N IS NOT A RECOGNIZED MESSAGE,
00073000  00000 1     THE STRING "ERROR NUMBER N" IS RETURNED. THE LONGEST
00074000  00000 1     MESSAGE WILL BE LIMITED TO 72 CHARACTERS.>>
00075000  00000 1
00076000  00000 1   BEGIN
00077000  00000 2   <<THE ARRAY 'MESSAGES' CONTAINS ALL THE MESSAGES, EACH ONE IS
00078000  00000 2     REPRESENTED AS A BYTE CONTAINING THE MESSAGE LENGTH, FOLLOWED
00079000  00000 2     BY THE MESSAGE ITSELF.>>
00080000  00000 2
00081000  00000 2   BYTE ARRAY MESSAGES(0:0)=PB:=
00082000  00000 2
00083000  00000 2   << 1>> 36,"NO NUMINPUTFILES PARAMETER SPECIFIED"
00084000  00022 2   << 2>>,21,"ILLEGAL NUMINPUTFILES"
00085000  00035 2   << 3>>,33,"NO INPUTFILES PARAMETER SPECIFIED"
00086000  00056 2   << 4>>,56,"NEITHER OUTPUTFILE NOR POSTPROCESSOR PARAMETER SPECIFIED"
00087000  00113 2   << 5>>,56,"IF KEYCOMPARE IS SPECIFIED, KEYS AND NUMKEYS MUST NOT BE"
00088000  00147 2   << 6>>,56,"IF KEYCOMPARE IS NOT SPECIFIED, KEYS AND NUMKEYS MUST BE"
00089000  00204 2   << 7>>,15,"ILLEGAL NUMKEYS"
00090000  00214 2   << 8>>,49,"KEYFIELD IS NOT WITHIN RECORD LENGTH OF EACH FILE"
00091000  00245 2   << 9>>,33,"ILLEGAL ASCENDING/DESCENDING CODE"
00092000  00266 2   <<10>>,16,"ILLEGAL KEY CODE"
00093000  00276 2   <<11>>,30,"FAILURE ON FGETINFO(INPUTFILE)"
00094000  00316 2   <<12>>,25,"FREAD ERROR ON INPUT FILE"
00095000  00333 2   <<13>>,27,"FWRITE ERROR ON OUTPUT FILE"
```

HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

```
00096000  00351 2   <<14>>,44,"INPUT RECORD DOES NOT INCLUDE ALL KEY FIELDS"
00097000  00377 2   <<15>>,47,"IF KEYCOMPARE IS SPECIFIED, KEYSONLY MAY NOT BE"
00098000  00427 2   <<16>>,24,"INSUFFICIENT STACK SPACE"
00099000  00444 2   ;
00100000  00444 2   INTEGER X=X;
00101000  00444 2   BYTE I;            <<USED TO GET BYTES FROM MESSAGES>>
00102000  00444 2   EQUATE MAXERROR=16; <<HIGHEST ERROR NUMBER>>
00103000  00444 2
00104000  00444 2      MOVE MESS:="MERGELIB: ",2;
00105000  00460 2      IF NOT (1<=N<=MAXERROR) THEN
00106000  00464 2      BEGIN <<PRODUCE "ERROR NUMBER N">>
00107000  00464 3         MOVE *:="ERROR NUMBER ";
00108000  00500 3         L:=ASCII(N,10,MESS(23))+23;
00109000  00510 3      END
00110000  00510 2   ELSE
00111000  00511 2      BEGIN <<N IS LEGITIMATE, SCAN MESSAGES N TIMES,>>
00112000  00511 3         TOS:=@I; TOS:=@MESSAGES;
00113000  00515 3         X:=N;
00114000  00516 3         WHILE (X:=X-1)>0 DO
00115000  00520 3         BEGIN
00116000  00520 4            MOVE *:=* PB,(1),1;  <<READ MESSAGE LENGTH>>
00117000  00522 4            TOS:=TOS+LOGICAL(I); <<SKIP MESSAGE>>
00118000  00523 4            ASSEMBLE(DECB);      <<RESET I'S POINTER>>
00119000  00525 4         END;
00120000  00527 3      <<NOW TOS POINTS TO THE MESSAGE>>
00121000  00527 3         MOVE *:=* PB,(1),1; <<READ THE COUNT>>
00122000  00531 3         DELB;
00123000  00531 3         MOVE *:=* PB,(L:=I);
00124000  00536 3         L:=L+10;
00125000  00541 3      END;
00126000  00541 2   END <<MERGEERRORMESS>>;
```

IDENTIFIER	CLASS	TYPE	ADDRESS	
I	SIMP. VAR.	BYTE	Q +001	
L	SIMP. VAR.	INTEGER	Q -004	
MAXERROR	EQUATE			VALUE = %20
MESS	ARRAY	BYTE	Q -005	
MESSAGES	ARRAY	BYTE	PB+000	
N	SIMP. VAR.	INTEGER	Q -006	
X	SIMP. VAR.	INTEGER	XREG	

```
00000  022116 047440 047125 046511 047120 052524 043111 046105   00010  051440 050101 051101 046505 052105 051040 051520 042503
00020  044506 044505 042025 044514 046105 043501 046040 047125   00030  046511 047120 052524 043111 046105 051441 047117 020111
00040  047120 052524 043111 046105 051440 050101 051101 046505   00050  052105 051040 051520 042503 044506 044505 042070 047105
00060  044524 044105 051040 047525 052120 052524 043111 046105   00070  020116 047522 020120 047523 052120 051117 041505 051523
00100  047522 020120 040522 040515 042524 042522 020123 050105   00110  041511 043111 042504 034111 043040 045505 054503 047515
00120  050101 051105 020111 051440 051520 042503 044506 044505   00130  042054 020113 042531 051440 040516 042040 047125 046513
00140  042531 051440 046525 051524 020116 047524 020102 042470   00150  044506 020113 042531 041517 046520 040522 042440 044523
00160  020116 047524 020123 050105 041511 043111 042504 026040   00170  045505 054523 020101 047104 020116 052515 045505 054523
00200  020115 052523 052040 041105 007511 046114 042507 040514   00210  020116 052515 045505 054523 030513 042531 043111 042514
00220  042040 044523 020116 047524 020127 044524 044111 047040   00230  051105 041517 051104 020114 042516 043524 044040 047506
00240  020105 040503 044040 043111 046105 020511 046114 042507   00250  040514 020101 051503 042516 042111 047107 027504 042523
00260  041505 047104 044516 043440 041517 042105 010111 046114   00270  042507 040514 020113 042531 020103 047504 042436 043101
```

```
00300  044514 052522 042440 047516 020106 043505 052111 047106    00310  047450 044516 050125 052106 044514 042451 014506 051105
00320  040504 020105 051122 047522 020117 047040 044516 050125    00330  052040 043111 046105 015506 053522 044524 042440 042522
00340  051117 051040 047516 020117 052524 050125 052040 043111    00350  046105 026111 047120 052524 020122 042503 047522 042040
00360  042117 042523 020116 047524 020111 047103 046125 042105    00370  020101 046114 020113 042531 020106 044505 046104 051457
00400  044506 020113 042531 041517 046520 040522 042440 044523    00410  020123 050105 041511 043111 042504 026040 045505 054523
00420  047516 046131 020115 040531 020116 047524 020102 042430    00430  044516 051525 043106 044503 044505 047124 020123 052101
00440  041513 020123 050101 041505 035001 173605 170003 010201    00450  140006 046505 051107 042514 044502 035040 021012 020042
00460  021001 021020 131606 012626 170003 010201 140010 042522    00470  051117 051040 047125 046502 042522 020040 021015 020043
00500  000600 041606 021012 021427 177605 000000 022427 053604    00510  140031 171401 010201 172013 010201 131606 000500 141310
00520  021001 020041 150401 006000 007400 140407 177252 021001    00530  020041 000100 150401 004500 053604 020043 043604 022412
00540  053604 031403
```

```
00128000  00000 1   PROCEDURE MERGE(NUMINPUTFILES,INPUTFILES,OUTPUTFILE,KEYSONLY,NUMKEYS,
00129000  00000 1                   KEYS,PREPROCESSOR,POSTPROCESSOR,ERRORPROC,KEYCOMPARE,
00130000  00000 1                   STATISTICS,FAILURE);
00131000  00000 1   VALUE NUMINPUTFILES,OUTPUTFILE,KEYSONLY,NUMKEYS;
00132000  00000 1   INTEGER NUMINPUTFILES,OUTPUTFILE,NUMKEYS;
00133000  00000 1   INTEGER ARRAY INPUTFILES,KEYS,STATISTICS;
00134000  00000 1   LOGICAL KEYSONLY,FAILURE;
00135000  00000 1   PROCEDURE PREPROCESSOR,POSTPROCESSOR,ERRORPROC;
00136000  00000 1   LOGICAL PROCEDURE KEYCOMPARE;
00137000  00000 1   OPTION VARIABLE;
00138000  00000 1
00139000  00000 1   <<MERGE READS DATA FROM ONE OR MORE SORTED INPUT FILES AND MERGES THEM,
00140000  00000 1     PRODUCING A SINGLE OUTPUT FILE.  THE PARAMETERS ARE DEFINED AS
00141000  00000 1     FOLLOWS...
00142000  00000 1
00143000  00000 1   NUMINPUTFILES IS THE NUMBER OF FILES TO BE MERGED. THIS PARAMETER IS NOT
00144000  00000 1   OPTIONAL.
00145000  00000 1
00146000  00000 1   INPUTFILES IS AN ARRAY CONTAINING THE MPE SYSTEM FILE NUMBERS OF THE
00147000  00000 1   FILES TO BE MERGED. THIS PARAMETER IS NOT OPTIONAL.
00148000  00000 1
00149000  00000 1   OUTPUTFILE IS THE MPE SYSTEM FILE NUMBER OF THE FILE TO WHICH THE
00150000  00000 1   MERGED RECORDS ARE TO BE WRITTEN.  IF NOT SPECIFIED, MERGED RECORDS ARE
00151000  00000 1   SIMPLY DISCARDED.  IN THAT CASE, THE POSTPROCESSOR PARAMETER MUST BE
00152000  00000 1   SPECIFIED.
00153000  00000 1
00154000  00000 1   KEYSONLY, IF TRUE, INDICATES THAT ONLY THE KEYS ARE TO BE OUTPUT, IF
00155000  00000 1   FALSE, OR IF UNSPECIFIED, THE ENTIRE RECORDS ARE OUTPUT.
00156000  00000 1
00157000  00000 1   NUMKEYS AND KEYS DEFINE THE RULES FOR COMPARING TWO RECORDS, EITHER THEY
00158000  00000 1   OR THE KEYCOMPARE PARAMETER MUST BE SPECIFIED, BUT NOT BOTH. NUMKEYS
00159000  00000 1   GIVES THE NUMBER OF KEYS TO BE USED IN THE COMPARE.  KEYS IS AN ARRAY
00160000  00000 1   WHICH CONTAINS 3 WORDS FOR EACH KEY FIELD.  THESE ARE DEFINED AS
00161000  00000 1   FOLLOWS...
00162000  00000 1      WORD 0 = POSITION WITHIN RECORD OF 1ST CHARACTER OF KEY (1ST BYTE=1)
00163000  00000 1      WORD 1 = NUMBER OF BYTES IN KEY
00164000  00000 1      WORD 2, BITS (0:8) = 0 FOR ASCENDING KEY, 1 FOR DESCENDING
00165000  00000 1      WORD 2, BITS (8:8) GIVES TYPE OF DATA
00166000  00000 1               0 = LOGICAL
00167000  00000 1               1 = 2'S COMPLEMENT
00168000  00000 1               2 = REAL
00169000  00000 1               3 = PACKED DECIMAL
00170000  00000 1               4 = NUMERIC DISPLAY
00171000  00000 1               5 = PACKED, EVEN NUMBER OF DIGITS
00172000  00000 1
00173000  00000 1   PREPROCESSOR IS A PROCEDURE WHICH, IF SPECIFIED, IS CALLED ONCE FOR EACH
00174000  00000 1   RECORD WHEN IT IS READ FROM ONE OF THE INPUT FILES.  THE PROCEDURE
00175000  00000 1   SHOULD BE DECLARED AS FOLLOWS...
00176000  00000 1         PROCEDURE PREPROCESSOR(FILE,RECORD,LENGTH);
00177000  00000 1         INTEGER FILE,LENGTH;
00178000  00000 1         BYTE ARRAY RECORD;
00179000  00000 1            ETC...
00180000  00000 1
00181000  00000 1         FILE IS AN INDEX TO THE ARRAY INPUTFILES INDICATING WHICH FILE THE
00182000  00000 1         RECORD IS FROM. RECORD IS THE RECORD, LENGTH IS THE NUMBER OF
00183000  00000 1         BYTES IN THE RECORD.
00184000  00000 1
```

```
00185000  00000 1   POSTPROCESSOR IS A PROCEDURE WHICH, IF SPECIFIED, IS CALLED ONCE FOR
00186000  00000 1   EACH RECORD BEFORE IT IS WRITTEN TO THE OUTPUT FILE. THE PROCEDURE
00187000  00000 1   SHOULD BE DECLARED AS FOLLOWS...
00188000  00000 1         PROCEDURE POSTPROCESSOR(RECORD,LENGTH);
00189000  00000 1         BYTE ARRAY RECORD;
00190000  00000 1         INTEGER LENGTH;
00191000  00000 1            ETC...
00192000  00000 1
00193000  00000 1         RECORD AND LENGTH ARE AS IN PREPROCESSOR.
00194000  00000 1
00195000  00000 1   ERRORPROC IS A PROCEDURE WHICH, IF SPECIFIED, IS CALLED WHENEVER A FATAL
00196000  00000 1   ERROR OCCURS IN THE MERGE PROGRAM.  IT SHOULD BE DECLARED AS FOLLOWS...
00197000  00000 1         PROCEDURE ERRORPROC(ERRORCODE);
00198000  00000 1         INTEGER ERRORCODE;
00199000  00000 1            ETC...
00200000  00000 1
00201000  00000 1         ERRORCODE IS THE NUMBER OF THE ERROR. IT MAY BE USED TO CALL
00202000  00000 1         MERGEERRORMESS. IF ERRORPROC IS NOT SPECIFIED, A MESSAGE WILL BE
00203000  00000 1         PRINTED AUTOMATICALLY.
00204000  00000 1
00205000  00000 1   KEYCOMPARE, IF SPECIFIED, IS CALLED WHENEVER TWO RECORDS MUST BE
00206000  00000 1   COMPARED.  THIS MAY BE USED INSTEAD OF NUMKEYS AND KEYS.  IT SHOULD BE
00207000  00000 1   DECLARED AS FOLLOWS;
00208000  00000 1         LOGICAL PROCEDURE KEYCOMPARE(REC1,LEN1,REC2,LEN2);
00209000  00000 1         BYTE ARRAY REC1,REC2;
00210000  00000 1         INTEGER LEN1,LEN2;
00211000  00000 1            ETC...
00212000  00000 1
00213000  00000 1         REC1 AND REC2 ARE POINTERS TO THE TWO RECORDS. LEN1 AND LEN2 ARE
00214000  00000 1         THEIR LENGTHS IN BYTES.  THE PROCEDURE SHOULD RETURN 'TRUE' IF
00215000  00000 1         REC1 SHOULD PRECEDE REC2, 'FALSE' OTHERWISE. 'TRUE' SHOULD ALSO BE
00216000  00000 1         RETURNED IN CASE OF TIES, TO INSURE THAT EQUAL KEY RECORDS ARE
00217000  00000 1         ORDERED BY THEIR POSITIONS IN 'INPUTFILES'.
00218000  00000 1
00219000  00000 1   STATISTICS, IF SPECIFIED, IS FILLED WITH THE FOLLOWING DATA;
00220000  00000 1
00221000  00000 1         WORD 0       NUMBER OF INPUT FILES
00222000  00000 1              1-2     NUMBER OF RECORDS MERGED
00223000  00000 1              3       SPACE AVAILABLE IN WORDS
00224000  00000 1              4-5     NUMBER OF COMPARES
00225000  00000 1              6-7     CPU TIME (MS)
00226000  00000 1              8-9     ELAPSED TIME (MS)
00227000  00000 1
00228000  00000 1   FAILURE, IF SPECIFIED, IS SET TO TRUE IF AN ERROR OCCURS, FALSE IF NOT.
00229000  00000 1
00230000  00000 1   ERROR CONDITIONS...
00231000  00000 1       CCE = NO ERROR OCCURRED
00232000  00000 1       CCL = AN ERROR OCCURRED
00233000  00000 1
00234000  00000 1   >>
00235000  00000 1
00236000  00000 1   BEGIN
00237000  00000 2   ARRAY
00238000  00000 2      MERGESTATISTICS(*)=Q;      <<CONTAINS THE DATA FOR STATISTICS>>
00239000  00000 2   INTEGER NUMINFILES;   << = NUMINPUTFILES>>
00240000  00000 2   DOUBLE NUMRECS!=0D;
00241000  00000 2   INTEGER SPACEAVAIL;
```

```
00242000  00000 2   DOUBLE NUMCOMPARES:=0D;
00243000  00000 2   DOUBLE CPUTIME;
00244000  00000 2   DOUBLE ELAPSEDTIME;
00245000  00000 2   <<END OF STATISTICS>>
00246000  00000 2
00247000  00000 2   INTEGER
00248000  00000 2      ORIGINALZ,              <<ORIGINAL VALUE OF USER'S Z REGISTER>>
00249000  00000 2      MINLENGTH:=0,           <<MINIMUM VALID LENGTH OF A RECORD>>
00250000  00000 2      TOTALKEYS:=0,           <<TOTAL LENGTH OF KEY FIELDS>>
00251000  00000 2      T,                      <<MISC. TEMP>>
00252000  00000 2      I,L,R1SIGN,R2SIGN,      <<USED BY KEYCOMPARE>>
00253000  00000 2      W,                      <<VALUE OF WINNER, INDEXES THRU TREE>>
00254000  00000 2      W1,W2,                  <<USED BY MATCH>>
00255000  00000 2      X=X,S0=S-0,Q0=Q-0,      <<COMMON KLUDGES>>
00256000  00000 2      STAT=Q-1,              <<GET TO STATUS WORD>>
00257000  00000 2      MAXRECLEN:=0,           <<MAXIMUM RECLEN OF ALL INPUT FILES>>
00258000  00000 2      RECLEN,                 <<TEMP FOR FGETINFO(INPUTFILE)>>
00259000  00000 2      WORDSPERRECORD,         <<WDS OCCUPIED BY EACH RECORD>>
00260000  00000 2      SPACEPERFILE;           <<SPACEAVAIL / NUMINPUTFILES>>
00261000  00000 2   LOGICAL
00262000  00000 2      KEYCOMPARE':=FALSE,         <<TRUE IF KEYCOMPARE SPECIFIED>>
00263000  00000 2      OPTIONS=Q-4,            <<CONTAINS PARAM SPECIFIER BITS>>
00264000  00000 2      FAILED:=FALSE;          <<SET TRUE IF AN ERROR OCCURS>>
00265000  00000 2   EQUATE
00266000  00000 2      NUMINPUTFILEBIT=4,      <<BITS OF OPTIONS>>
00267000  00000 2      INPUTFILESBIT=NUMINPUTFILEBIT+1,
00268000  00000 2      OUTPUTFILEBIT=INPUTFILESBIT+1,
00269000  00000 2      KEYSONLYBIT=OUTPUTFILEBIT+1,
00270000  00000 2      NUMKEYSBIT=KEYSONLYBIT+1,
00271000  00000 2      KEYSBIT=NUMKEYSBIT+1,
00272000  00000 2      PREPROCESSORBIT=KEYSBIT+1,
00273000  00000 2      POSTPROCESSBIT=PREPROCESSORBIT+1,
00274000  00000 2      ERRORPROCBIT=POSTPROCESSBIT+1,
00275000  00000 2      KEYCOMPAREBIT=ERRORPROCBIT+1,
00276000  00000 2      STATISTICSBIT=KEYCOMPAREBIT+1,
00277000  00000 2      FAILUREBIT=STATISTICSBIT+1;
00278000  00000 2   EQUATE F=%(16)F,              <<HEX F>>
00279000  00000 2      PACKEDCODE=3,             <<KEYCODE FOR PACKED DECIMAL>>
00280000  00000 2      PACKEDMINUS=%15,          <<- IN PACKED DECIMAL>>
00281000  00000 2      DELTA="A"-"1",            <<MAPS + TO ABS IN NUMERIC DISPLAY>>
00282000  00000 2      PLUSZERO=%173,            <<+0 IN NUMERIC DISPLAY>>
00283000  00000 2      MINUSZERO=%175;           <<-0 IN NUMERIC DISPLAY>>
00284000  00000 2   DEFINE
00285000  00000 2      LEFT=(8:4)#,              <<LEFT HALF OF A BYTE>>
00286000  00000 2      RIGHT=(12:4)#;            <<RIGHT HALF OF A BYTE>>
00287000  00000 2   EQUATE
00288000  00000 2      STACKFUDGE=1000,         <<AMOUNT LEFT UNUSED TWEEN S&Z>>
00289000  00000 2      CCE=2,CCL=1;             <<CONDITION CODE SETTINGS>>
00290000  00000 2   ARRAY
00291000  00000 2      MESS(0:35);                <<FOR PRINTING MESSAGES>>
00292000  00000 2   INTEGER POINTER
00293000  00000 2      KEYPOSITION=KEYS,        <<SYNONYM FOR 1ST WD OF ENTRY IN KEYS>>
00294000  00000 2      KEYLENGTH,               <<USED TO ACCESS 2ND WD OF ENTRY IN KEYS>>
00295000  00000 2      KEYCODE,                 <<USED TO ACCESS 3RD WD OF ENTRY IN KEYS>>
00296000  00000 2      C1,C2,                   <<USED BY MATCH>>
00297000  00000 2      OUTPUTBUFFER,           <<USED FOR OUTPUTTING WHEN KEYSONLY IS SET>>
00298000  00000 2      BUFADR,                  <<POINTS TO AVAILABLE SPACE>>
```

```
00299000  00000 2       BUFINX,                 <<ARRAY OF POINTERS, 1 PER INPUTFILE>>
00300000  00000 2       WINNER,                 <<TREE OF WINNERS, >>
00301000  00000 2       CHAMP;                  <<POINTS TO BEST RECORD IN TREE>>
00302000  00000 2   BYTE POINTER
00303000  00000 2       B'OUTPUTBUFFER,         <<BYTE VERSION OF OUTPUTBUFFER>>
00304000  00000 2       BCHAMP,                 << "       "      "   CHAMP>>
00305000  00000 2       R1,R2;                  <<USED BY KEYCOMPARE>>
00306000  00000 2   DEFINE
00307000  00000 2       CONDCODE=STAT,(6:2)#;
```

```
00309000  00000 2   SUBROUTINE MATCH(NODE);
00310000  00000 2   VALUE NODE;
00311000  00000 2   INTEGER NODE;
00312000  00000 2
00313000  00000 2   <<MATCH COMPARES THE ELEMENTS WHICH ARE THE SONS OF NODE IN THE WINNER
00314000  00000 2   TREE AND WRITES THE NUMBER OF THE WINNER IN LOCATION WINNER(NODE).
00315000  00000 2   THE WINNER TREE IS A BINARY TREE WITH 2*NUMINFILES-1 NODES, NUMBERED
00316000  00000 2   FROM 1. THE FATHER OF NODE I IS NODE I/2. THE VALUE OF WINNER(I)
00317000  00000 2   IS I-NUMINFILES, IF I>=NUMINFILES.  OTHERWISE, IT IS EITHER THAT OF
00318000  00000 2   WINNER(2I) OR WINNER(2I+1), WHICHEVER REPRESENTS A FILE WHOSE CURRENT
00319000  00000 2   RECORD PRECEDES THE OTHER.>>
00320000  00000 2
00321000  00000 2   BEGIN
00322000  00000 3      W1:=WINNER(NODE+NODE);     <<FIRST COMPETITOR>>
00323000  00005 3      W2:=WINNER(X:=X+1);        <<SECOND COMPETITOR>>
00324000  00010 3      @C2:=BUFINX(W2);           <<PTS TO W2'S ENTRY>>
00325000  00013 3      IF C2<0 THEN GO TO WIN1;   <<NO CONTEST, FILE W2 IS DONE>>
00326000  00016 3      @C1:=BUFINX(W1);           <<PTS TO W1'S ENTRY>>
00327000  00021 3      IF C1<0 THEN GO TO WIN2;   <<NO CONTEST, FILE W1 IS DONE>>
00328000  00024 3      NUMCOMPARES:=NUMCOMPARES+1D; <<DO AN ACTUAL COMPARE>>
00329000  00030 3      @R1:=@C1&LSL(1)+1;         <<SET UP BYTE POINTERS TO THE RECORDS>>
00330000  00034 3      @R2:=@C2&LSL(1)+1;
00331000  00040 3      IF KEYCOMPARE; THEN
00332000  00042 3      BEGIN <<CALL USER PROCEDURE>>
00333000  00042 4         TOS:=KEYCOMPARE(R1(1),C1,R2(X),C2);
00334000  00052 4         GO TO FINI;
00335000  00060 4      END;
00336000  00060 3   <<PERFORM COMPARE USING TABLES>>
00337000  00060 3      I:=0;        <<INDEX TO KEYS>>
00338000  00062 3      DO
00339000  00062 3      BEGIN <<PROCESS A KEY>>
00340000  00062 4         L:=KEYLENGTH(I);           <<SET L>>
00341000  00065 4      <<MOVE R1,R2 TO START OF FIELD>>
00342000  00065 4         @R1:=@R1(KEYPOSITION(X));
00343000  00071 4         @R2:=@R2(X);
00344000  00073 4         X:=KEYCODE(I),(8:8);       <<TYPE OF KEY>>
00345000  00076 4         ASSEMBLE(BR *+1,X;         <<HOMEMADE SWITCH>>
00346000  00100 4            BR LOG; BR TCM; BR REALL;
00347000  00103 4            BR PACDEC; BR NUMDISP; BR PACDEC);
00348000  00106 4
00349000  00106 4   TCM:   <<TWOS COMPLEMENT>>
00350000  00106 4         IF R1&LSL(8)<>R2&LSL(8) THEN GO NEQ;
00351000  00114 4         IF R1(1)<>R2(X),(L-1) THEN GO NEQ; GO EQ;
00352000  00127 4
00353000  00127 4   REALL;  <<REAL OR LONG>>
00354000  00127 4         IF R1<128 AND R2<128 THEN
00355000  00135 4         BEGIN <<BOTH POSITIVE. DO LOGICAL COMPARE>>
00356000  00135 5   LOG:      IF R1<>R2,(L) THEN GO NEQ; GO EQ;
00357000  00145 5         END;
00358000  00145 4      <<IF EITHER IS NEGATIVE, LOGICAL COMPARE REVERSE>>
00359000  00145 4         IF R2<>R1,(L) THEN GO NEQ; GO EQ;
00360000  00155 4
00361000  00155 4   PACDEC: <<PACKED DECIMAL. SIGN MAGNITUDE WITH THE SIGN IN THE EXTREME
00362000  00155 4              RIGHTMOST 4 BIT FIELD, MINUS IS %15, PLUS ANYTHING ELSE.
00363000  00155 4              IF CODE IS 3, LEFTMOST 4BITS IS HIGH ORDER DIGIT, OTHERWISE
00364000  00155 4              LEFTMOST 4 BITS IS JUNK AND NEXT 4 BITS IS HIGH DIGIT>>
00365000  00155 4         R1SIGN:=X;          <<SAVE THE CODE>>
```

```
00366000  00157 4          IF R1(L-1).RIGHT=PACKEDMINUS THEN
00367000  00165 4          BEGIN <<R1 NEGATIVE>>
00368000  00165 5             IF R2(X).RIGHT=PACKEDMINUS THEN
00369000  00171 5             BEGIN <<BOTH NEGATIVE. COMPARE REVERSE>>
00370000  00171 6                IF R1SIGN=PACKEDCODE THEN
00371000  00174 6                   IF R2<>R1,(L) THEN GO TO NEQ ELSE GO TO EQ;
00372000  00206 6                IF L>1 AND (R2.RIGHT<>R1.RIGHT OR R2(1)<>R1(X),(L-1))
00373000  00226 6                   THEN GO TO NEQ; GO TO EQ;
00374000  00232 6             END <<BOTH NEGATIVE>>;
00375000  00232 5          END <<R1 NEGATIVE>>
00376000  00232 4       ELSE
00377000  00233 4          IF R2(X).RIGHT<>PACKEDMINUS THEN
00378000  00237 4          BEGIN <<BOTH POSITIVE>>
00379000  00237 5             IF R1SIGN=PACKEDCODE THEN
00380000  00242 5                IF R1<>R2,(X) OR R1(X).LEFT<>R2(X).LEFT THEN GO TO NEQ ELSE
00381000  00260 5                                                          GO TO EQ;
00382000  00261 5             IF L>1 AND (R1.RIGHT<>R2.RIGHT OR R1(1)<>R2(X),(L-2) OR
00383000  00301 5                         R1(L-1).LEFT<>R2(X).LEFT) THEN GO TO NEQ; GO TO EQ;
00384000  00313 5          END <<BOTH POSITIVE>>;
00385000  00313 4       <<OPPOSITE SIGNS. CHECK FOR BOTH ZERO>>
00386000  00313 4          IF R1SIGN=PACKEDCODE THEN
00387000  00316 4          BEGIN <<NORMAL CASE>>
00388000  00316 5             IF R1(X).LEFT=0 AND R2(X).LEFT=0 THEN
00389000  00326 5              IF L=1 OR R1=R2,(X) AND R1=0 AND R1=R1(1),(L-2) THEN GO TO EQ;
00390000  00351 5          END
00391000  00351 4       ELSE
00392000  00354 4          BEGIN <<FUNNY PACKED CASE>>
00393000  00354 5             IF L=1 THEN GO TO EQ;
00394000  00357 5             IF R1(X).LEFT=0 AND R2(X).LEFT=0 AND R1.RIGHT=0 AND R2.RIGHT=0
00395000  00375 5                AND R1(1)=R2(X),(L-2) THEN
00396000  00406 5                IF L=2 OR R1(X)=0 AND R1(X)=R1(2),(L-3) THEN GO TO EQ;
00397000  00424 5          END <<ZERO CHECKING>>;
00398000  00424 4       <<OPPOSITE SIGNS--NOT BOTH ZERO, SO THE NEGATIVE ONE WINS>>
00399000  00424 4          TOS:=R1(L-1).RIGHT=PACKEDMINUS; GO TO CHECKORDER;
00400000  00435 4
00401000  00435 4    NUMDISP; <<NUMERIC DISPLAY. THE DIGITS OCCUR 1 PER BYTE, IN ASCII. THE
00402000  00435 4               LAST DIGIT, HOWEVER, MAY INCLUDE EITHER A POSITIVE OR NEGA-
00403000  00435 4               TIVE SIGN.  A POSITIVE SIGN MAPS 0 INTO %173 AND 1-9 INTO
00404000  00435 4               "A"-"I". A NEGATIVE SIGN MAPS 0 INTO %175 AND 1-9 INTO
00405000  00435 4               "J"-"R".  IF THERE IS NO SIGN, 0-9 ARE REPRESENTED AS
00406000  00435 4               "0"-"9".  NO SIGN HAS THE SAME VALUE AS PLUS.>>
00407000  00435 4          R1SIGN:=R1(L-1);
00408000  00441 4          R2SIGN:=R2(X);
00409000  00443 4          IF R1SIGN>="J" AND R1SIGN<>PLUSZERO THEN
00410000  00451 4          BEGIN <<R1 IS NEGATIVE>>
00411000  00451 5             IF R2SIGN>="J" AND R2SIGN<>PLUSZERO THEN
00412000  00457 5             BEGIN <<BOTH NEGATIVE. COMPARE REVERSE.>>
00413000  00457 6                IF R2<>R1,(X) THEN GO TO NEQ;
00414000  00464 6                IF (IF R2SIGN=MINUSZERO THEN 0 ELSE R2SIGN)
00415000  00475 6                 <>(IF R1SIGN=MINUSZERO THEN 0 ELSE R1SIGN) THEN GO TO NEQ;
00416000  00505 6                GO TO EQ;
00417000  00507 6             END;
00418000  00507 5          <<R1 NEGATIVE, R2 POSITIVE. CHECK FOR BOTH ZERO>>
00419000  00507 5             IF R1SIGN=MINUSZERO AND (R2SIGN=PLUSZERO OR R2SIGN="0") THEN
00420000  00520 5             IF L=1 OR R1="0" AND R1=R1(1),(L-2) AND R1=R2,(L-1)
00421000  00540 5                THEN GO TO EQ;
00422000  00544 5             TOS:=1; GO TO CHECKORDER;     <<R1 WINS>>
```

```
00423000  00546 5          END;
00424000  00546 4       <<R1 POSITIVE>>
00425000  00546 4          IF R2SIGN<"J" OR R2SIGN=PLUSZERO THEN
00426000  00554 4          BEGIN <<BOTH POSITIVE>>
00427000  00554 5             IF R1<>R2,(X) THEN GO TO NEQ;
00428000  00561 5          <<COMPARE SIGNS>>
00429000  00561 5             IF (IF R1SIGN<"A" THEN R1SIGN ELSE IF R1SIGN=PLUSZERO THEN "0"
00430000  00573 5                                                  ELSE R1SIGN-DELTA)
00431000  00577 5              <>(IF R2SIGN<"A" THEN R2SIGN ELSE IF R2SIGN=PLUSZERO THEN "0"
00432000  00607 5                                                  ELSE R2SIGN-DELTA)
00433000  00613 5                THEN GO TO NEQ; GO TO EQ;
00434000  00617 5          END;
00435000  00617 4       <<R1 POSITIVE, R2 NEGATIVE, CHECK FOR BOTH ZERO,>>
00436000  00617 4          IF R2SIGN=MINUSZERO AND (R1SIGN=PLUSZERO OR R1SIGN="0") THEN
00437000  00630 4          IF L=1 OR R2="0" AND R2=R2(1),(L-2) AND R1=R2,(L-1)
00438000  00650 4                THEN GO TO EQ;
00439000  00654 4          TOS:=0; GO TO CHECKORDER;          <<R2 WINS>>
00440000  00656 4
00441000  00656 4
00442000  00656 4    EQ; <<KEY FIELDS ARE EQUAL>>
00443000  00656 4       <<ADJUST RECORD POINTERS TO START OF RECORD>>
00444000  00656 4          @R1:=@R1(-KEYPOSITION(I));
00445000  00663 4          @R2:=@R2(X);
00446000  00665 4       END UNTIL (I:=I+3)=NUMKEYS;
00447000  00673 3    <<ALL KEY FIELDS ARE EQUAL>>
00448000  00673 3
00449000  00673 3       IF @C1<@C2 THEN GO TO WIN1 ELSE GO TO WIN2;
00450000  00700 3
00451000  00700 3    NEQ; <<KEYFIELDS ARE UNEQUAL, AND CC IS SET ACCORDING TO R1:R2>>
00452000  00700 3
00453000  00700 3       TOS:=0;           <<DOESNT AFFECT CC>>
00454000  00700 3       IF < THEN TOS:=TOS+1;
00455000  00703 3
00456000  00703 3    CHECKORDER;  <<REVERSE DECISION IF DESCENDING FIELD>>
00457000  00703 3
00458000  00703 3       IF KEYCODE(I)>255 THEN TOS:=NOT TOS;
00459000  00710 3
00460000  00710 3    FINI;
00461000  00710 3       IF TOS THEN
00462000  00711 3    WIN1; WINNER(NODE):=W1 ELSE
00463000  00715 3    WIN2; WINNER(NODE):=W2;
00464000  00720 3
00465000  00720 3    END <<MATCH>>;
```

IDENTIFIER	CLASS	TYPE	ADDRESS
CHECKORDER	LABEL		PB+703
EQ	LABEL		PB+656
FINI	LABEL		PB+710
LOG	LABEL		PB+135
NEQ	LABEL		PB+700
NODE	SIMP, VAR,	INTEGER	S -001
NUMDISP	LABEL		PB+435
PACDEC	LABEL		PB+155
REALL	LABEL		PB+127
TCM	LABEL		PB+106

```
WIN1            LABEL           PB+711
WIN2            LABEL           PB+715
```

```
00467000  00721 2   SUBROUTINE ERROR(N);
00468000  00721 2   VALUE N;
00469000  00721 2   INTEGER N;
00470000  00721 2
00471000  00721 2   <<PROCESS ERROR NUMBER N>>
00472000  00721 2
00473000  00721 2   BEGIN
00474000  00721 3      IF OPTIONS.(ERRORPROCBIT:1) THEN ERRORPROC(N) ELSE
00475000  00730 3      BEGIN <<PRINT ERROR MESSAGE>>
00476000  00730 4         MERGEERRORMESS(N,MESS,L);
00477000  00735 4         PRINT(MESS,-L,0);
00478000  00741 4      END;
00479000  00741 3      FAILED:=TRUE;            <<SET DEATH FLAG>>
00480000  00743 3      GO TO FINI;
00481000  00745 3   END <<ERROR>>;
```

IDENTIFIER	CLASS	TYPE	ADDRESS
N	SIMP. VAR.	INTEGER	S -001

```
00482000  00751 2   SUBROUTINE IOERROR(FILE,N);
00483000  00751 2   VALUE FILE,N;
00484000  00751 2   INTEGER FILE,N;
00485000  00751 2
00486000  00751 2   <<PROCESS IO ERROR N ON FILE>>
00487000  00751 2
00488000  00751 2   BEGIN
00489000  00751 3      PRINT'FILE'INFO(FILE);
00490000  00753 3      ERROR(N);
00491000  00756 3   END <<IOERROR>>;
```

IDENTIFIER	CLASS	TYPE	ADDRESS
FILE	SIMP. VAR.	INTEGER	S -002
N	SIMP. VAR.	INTEGER	S -001

```
00493000  00757 2   SUBROUTINE FILL(FILE);
00494000  00757 2   VALUE FILE;
00495000  00757 2   INTEGER FILE;
00496000  00757 2
00497000  00757 2   <<READ RECORDS FROM INPUTFILE(FILE) UNTIL THE BUFFER IS FILLED. THE
00498000  00757 2   BUFFER BEGIN AT BUFADR(FILE*SPACEPERFILE). >>
00499000  00757 2
00500000  00757 2   BEGIN
00501000  00757 3      @C1:=BUFINX(FILE):=FILE*SPACEPERFILE+@BUFADR; <<C1 PTS TO NEXT REC>>
00502000  00766 3      @C2:=@C1+SPACEPERFILE;                  <<BOUNDARY OF SPACE FOR FILE>>
00503000  00771 3      DO
00504000  00771 3      BEGIN <<READ A RECORD>>
00505000  00771 4         C1:=FREAD(INPUTFILES(FILE),C1(1),-MAXRECLEN);
00506000  01002 4         IF <> THEN
00507000  01003 4         BEGIN <<ERROR OR EOF>>
00508000  01003 5            IF > THEN
00509000  01004 5            BEGIN <<EOF. MARK WITH -2>>
00510000  01004 6               C1:=-2;
00511000  01006 6               RETURN;
00512000  01007 6            END;
00513000  01007 5            IOERROR(INPUTFILES(FILE),12);
00514000  01014 5         END;
00515000  01014 4         IF C1<MINLENGTH THEN IOERROR(INPUTFILES(FILE),14);
00516000  01024 4         IF OPTIONS.(PREPROCESSORBIT:1) THEN
00517000  01027 4         BEGIN <<CALL USER'S PREPROCESSOR PROCEDURE>>
00518000  01027 5            TOS:=@FILE;
00519000  01030 5            TOS:=(@C1+1)&LSL(1);
00520000  01033 5            PREPROCESSOR(*,*,C1);
00521000  01036 5         END;
00522000  01036 4         @C1:=(C1+3)&LSR(1)+@C1;
00523000  01043 4      END UNTIL @C2-@C1<=WORDSPERRECORD;  <<CHECK FOR OUT OF SPACE>>
00524000  01047 3      C1:=-1;                          <<MARK END OF BUFFER>>
00525000  01051 3   END <<FILL>>;
```

IDENTIFIER	CLASS	TYPE	ADDRESS
FILE	SIMP. VAR.	INTEGER	S -001

```
00527000  01053 2   <<SAVE USER'S Z REGISTER>>
00528000  01053 2
00529000  01053 2      PUSH(Z)!  ORIGINALZ!=TOS!
00530000  01073 2
00531000  01073 2   <<START TIMING>>
00532000  01073 2
00533000  01073 2      IF OPTIONS.(STATISTICSBIT:1) THEN
00534000  01076 2      BEGIN
00535000  01076 3         CPUTIME!=PROCTIME!
00536000  01101 3         ELAPSEDTIME!=TIMER!
00537000  01104 3      END!
00538000  01104 2
00539000  01104 2   <<CHECK INPUTFILES>>
00540000  01104 2
00541000  01104 2      IF NOT OPTIONS.(NUMINPUTFILEBIT:1) THEN ERROR(1)!
00542000  01112 2      IF (NUMINFILES!=NUMINPUTFILES)<=0 THEN ERROR(2)!
00543000  01122 2      IF NOT OPTIONS.(INPUTFILESBIT:1) THEN ERROR(3)!
00544000  01130 2
00545000  01130 2   <<CHECK OUTPUTFILE>>
00546000  01130 2
00547000  01130 2      IF NOT OPTIONS.(OUTPUTFILEBIT:1) AND NOT OPTIONS.(POSTPROCESSBIT:1)
00548000  01133 2         THEN ERROR(4)!
00549000  01141 2
00550000  01141 2   <<CHECK KEYSONLY>>
00551000  01141 2
00552000  01141 2      IF NOT OPTIONS.(KEYSONLYBIT:1) THEN KEYSONLY:=FALSE!
00553000  01146 2
00554000  01146 2   <<CHECK KEYS>>
00555000  01146 2
00556000  01146 2      IF OPTIONS.(KEYCOMPAREBIT:1) AND @KEYCOMPARE<>0 THEN
00557000  01154 2      BEGIN <<KEYCOMPARE SPECIFIED, DISALLOW KEYS, NUMKEYS, AND KEYSONLY>>
00558000  01154 3         IF KEYSONLY THEN ERROR(15)!
00559000  01161 3         IF INTEGER(OPTIONS).(NUMKEYSBIT:2)<>%(2)00 THEN ERROR(5)!
00560000  01170 3         KEYCOMPARE!:=TRUE!       <<INDICATE THAT IT'S SPECIFIED>>
00561000  01172 3      END
00562000  01172 2   ELSE
00563000  01173 2      BEGIN <<KEYCOMPARE NOT SPECIFIED>>
00564000  01173 3         IF INTEGER(OPTIONS).(NUMKEYSBIT:2)<>%(2)11 THEN ERROR(6)!
00565000  01202 3         IF NOT (1<=NUMKEYS<=10000) THEN ERROR(7)!
00566000  01212 3         NUMKEYS!:=NUMKEYS*3!
00567000  01215 3         @KEYLENGTH!:=@KEYS+1!
00568000  01220 3         @KEYCODE!:=@KEYS+2!
00569000  01223 3         X!:=0!
00570000  01223 3         DO
00571000  01223 3         BEGIN <<CHECK INDIVIDUAL KEY>>
00572000  01224 4            IF KEYPOSITION(X)<=0 OR KEYLENGTH(X)<=0 OR
00573000  01232 4               32767-KEYPOSITION(X)<KEYLENGTH(X) THEN ERROR(8)!
00574000  01242 4            IF (T!:=KEYPOSITION(X)+KEYLENGTH(X)-1)>MINLENGTH
00575000  01246 4               THEN MINLENGTH!:=T!
00576000  01252 4            IF KEYCODE(X).(0:7)<>0 THEN ERROR(9)!
00577000  01261 4            IF KEYCODE(X).(8:8)>5 THEN ERROR(10)!
00578000  01270 4            TOTALKEYS!:=TOTALKEYS+KEYLENGTH(X)!
00579000  01273 4         END UNTIL (X!:=X+3)=NUMKEYS!
00580000  01277 3      END <<KEYFIELD PROCESSING>>!
00581000  01277 2
00582000  01277 2   <<DETERMINE MAXIMUM RECORD LENGTH OF INPUT FILES>>
00583000  01277 2
```

```
00584000  01277 2       FOR X:=0 UNTIL NUMINPUTFILES-1 DO
00585000  01303 2       BEGIN
00586000  01306 3          FGETINFO(INPUTFILES(X),,,,RECLEN);
00587000  01314 3          IF <> THEN IOERROR(INPUTFILES(X),11);
00588000  01323 3          RECLEN:=IF RECLEN<0 THEN -RECLEN ELSE RECLEN&LSL(1);
00589000  01334 3          IF RECLEN<MINLENGTH THEN ERROR(8);
00590000  01343 3          IF RECLEN>MAXRECLEN THEN MAXRECLEN:=RECLEN;
00591000  01350 3       END;
00592000  01351 2       WORDSPERRECORD:=(MAXRECLEN+3)&LSR(1);
00593000  01355 2
00594000  01355 2       IF OPTIONS.(OUTPUTFILEBIT:1) AND KEYSONLY THEN
00595000  01362 2       BEGIN <<ALLOCATE OUTPUTBUFFER>>
00596000  01362 3          TOS:=(TOTALKEYS+1)&LSR(1);
00597000  01365 3          @OUTPUTBUFFER:=@S0;
00598000  01367 3          @@OUTPUTBUFFER:=@S0&LSL(1);
00599000  01372 3          ASSEMBLE(ADDS 0);
00600000  01373 3       END;
00601000  01373 2
00602000  01373 2    <<ALLOCATE TABLES>>
00603000  01373 2
00604000  01373 2       @WINNER:=@S0; TOS:=NUMINPUTFILES&LSL(1)-1; ASSEMBLE(ADDS 0);
00605000  01401 2       TOS:=NUMINPUTFILES; @BUFINX:=@S0; ASSEMBLE(ADDS 0);
00606000  01405 2
00607000  01405 2    <<ALLOCATE BUFFER SPACE FOR INPUT FILES>>
00608000  01405 2
00609000  01405 2       SPACEAVAIL:=ZSIZE(32767)-STACKFUDGE-@S0;
00610000  01415 2       SPACEPERFILE:=SPACEAVAIL/NUMINPUTFILES;
00611000  01421 2       IF SPACEPERFILE<=WORDSPERRECORD THEN ERROR(16);
00612000  01432 2
00613000  01432 2    <<INITIALIZE TABLES>>
00614000  01432 2
00615000  01432 2       TOS:=NUMINPUTFILES*SPACEPERFILE; @BUFADR:=@S0; ASSEMBLE(ADDS 0);
00616000  01437 2       FOR T:=0 UNTIL NUMINPUTFILES-1 DO
00617000  01445 2       BEGIN <<FILL BUFFER I>>
00618000  01447 3          FILL(T);
00619000  01453 3          WINNER(T+NUMINPUTFILES):=T;
00620000  01460 3       END;
00621000  01461 2
00622000  01461 2    <<INITIALIZE THE SELECTION TREE>>
00623000  01461 2
00624000  01461 2       FOR T:=NUMINPUTFILES-1 STEP -1 UNTIL 1 DO MATCH(T);
00625000  01500 2
00626000  01500 2    <<NOW START OUTPUTTING THE RESULTS>>
00627000  01500 2
00628000  01500 2    OUTPUTLOOP:
00629000  01500 2
00630000  01500 2       @CHAMP:=BUFINX(W:=WINNER(1));
00631000  01507 2       IF CHAMP>=0 THEN
00632000  01512 2       BEGIN <<VALID RECORD>>
00633000  01512 3          NUMRECS:=NUMRECS+1D;
00634000  01516 3          IF OPTIONS.(POSTPROCESSBIT:1) THEN
00635000  01521 3          BEGIN <<CALL USER'S POSTPROCESSOR PROCEDURE>>
00636000  01521 4             TOS:=(@CHAMP+1)&LSL(1);
00637000  01524 4             POSTPROCESSOR(*,CHAMP);
00638000  01527 4          END;
00639000  01527 3          IF OPTIONS.(OUTPUTFILEBIT:1) THEN
00640000  01532 3          BEGIN <<SHIP TO OUTPUTFILE>>
```

```
00641000  01532 4            IF KEYSONLY THEN
00642000  01534 4            BEGIN <<EXTRACT THE KEYS>>
00643000  01534 5               @BCHAMP:=@CHAMP&LSL(1)+1;
00644000  01540 5               I:=0;                        <<FIELD COUNTER>>
00645000  01542 5               TOS:=@B'OUTPUTBUFFER;
00646000  01543 5               DO MOVE *:=BCHAMP(KEYPOSITION(I)),(KEYLENGTH(I)),2
00647000  01551 5                     UNTIL (I:=I+3)=NUMKEYS;
00648000  01560 5               DEL;
00649000  01560 5               FWRITE(OUTPUTFILE,OUTPUTBUFFER,-TOTALKEYS,0);
00650000  01566 5            END <<KEYSONLY>>
00651000  01566 4         ELSE FWRITE(OUTPUTFILE,CHAMP(1),-CHAMP,0);
00652000  01577 4            IF <> THEN IOERROR(OUTPUTFILE,13);
00653000  01605 4         END <<SHIP TO OUTPUTFILE>>;
00654000  01605 3
00655000  01605 3   <<ADJUST POINTER FOR FILE WINNER(1)>>
00656000  01605 3
00657000  01605 3         BUFINX(W):=@CHAMP:=@CHAMP+(CHAMP+3)&LSR(1);
00658000  01615 3         IF CHAMP=-1 THEN FILL(W);
00659000  01624 3
00660000  01624 3   <<SORT NEW RECORD INTO PLACE>>
00661000  01624 3
00662000  01624 3         W:=W+NUMINPUTFILES;
00663000  01627 3         WHILE (W:=W&LSR(1))>0 DO MATCH(W);
00664000  01642 3         GO TO OUTPUTLOOP;
00665000  01643 3      END <<VALID RECORD>>;
00666000  01643 2
00667000  01643 2   <<GATHER FINAL STATISTICS>>
00668000  01643 2
00669000  01643 2      IF OPTIONS.(STATISTICSBIT:1) THEN
00670000  01646 2      BEGIN
00671000  01646 3         CPUTIME:=PROCTIME-CPUTIME;
00672000  01653 3         ELAPSEDTIME:=TIMER-ELAPSEDTIME;
00673000  01660 3         MOVE STATISTICS:=MERGESTATISTICS,(NUMSTATS);
00674000  01664 3      END;
00675000  01664 2
00676000  01664 2   <<CUT BACK STACK>>
00677000  01664 2
00678000  01664 2   FINI;
00679000  01664 2      IF OPTIONS.(FAILUREBIT:1) THEN FAILURE:=FAILED;
00680000  01671 2      CONDCODE:=IF FAILED THEN CCL ELSE CCE;
00681000  01701 2
00682000  01701 2      X:=ORIGINALZ;
00683000  01702 2      TOS:=@Q0; SET(S);
00684000  01704 2      ZSIZE(X);
00685000  01706 2   END <<MERGE>>;
```

IDENTIFIER	CLASS	TYPE	ADDRESS	
B'OUTPUTBUFFER	POINTER	BYTE	Q +046	
BCHAMP	POINTER	BYTE	Q +047	
BUFADR	POINTER	INTEGER	Q +042	
BUFINX	POINTER	INTEGER	Q +043	
C1	POINTER	INTEGER	Q +037	
C2	POINTER	INTEGER	Q +040	
CCE	EQUATE			VALUE = %2
CCL	EQUATE			VALUE = %1

```
CHAMP                POINTER        INTEGER     Q +045
CONDCODE             DEFINE                                 STAT.(6:2)
CPUTIME              SIMP. VAR.     DOUBLE      Q +007
DELTA                EQUATE                                 VALUE = %20
ELAPSEDTIME          SIMP. VAR.     DOUBLE      Q +011
ERROR                SUBROUTINE                 PB+721
ERRORPROC            PROCEDURE                  Q -010
ERRORPROCBIT         EQUATE                                 VALUE = %14
F                    EQUATE                                 VALUE = %17
FAILED               SIMP. VAR.     LOGICAL     Q +033
FAILURE              SIMP. VAR.     LOGICAL     Q -005
FAILUREBIT           EQUATE                                 VALUE = %17
FILL                 SUBROUTINE                 PB+757
FINI                 LABEL                      PB+1664
I                    SIMP. VAR.     INTEGER     Q +017
INPUTFILES           ARRAY          INTEGER     Q -017
INPUTFILESBIT        EQUATE                                 VALUE = %5
IOERROR              SUBROUTINE                 PB+751
KEYCODE              POINTER        INTEGER     Q +036
KEYCOMPARE           PROCEDURE      LOGICAL     Q -007
KEYCOMPARE!          SIMP. VAR.     LOGICAL     Q +032
KEYCOMPAREBIT        EQUATE                                 VALUE = %15
KEYLENGTH            POINTER        INTEGER     Q +035
KEYPOSITION          POINTER        INTEGER     Q -013
KEYS                 ARRAY          INTEGER     Q -013
KEYSBIT              EQUATE                                 VALUE = %11
KEYSONLY             SIMP. VAR.     LOGICAL     Q -015
KEYSONLYBIT          EQUATE                                 VALUE = %7
L                    SIMP. VAR.     INTEGER     Q +020
LEFT                 DEFINE                                 (8:4)
MATCH                SUBROUTINE                 PB+000
MAXRECLEN            SIMP. VAR.     INTEGER     Q +026
MERGESTATISTICS      ARRAY          LOGICAL     Q +001
MESS                 ARRAY          LOGICAL     Q +034
MINLENGTH            SIMP. VAR.     INTEGER     Q +014
MINUSZERO            EQUATE                                 VALUE = %175
NUMCOMPARES          SIMP. VAR.     DOUBLE      Q +005
NUMINFILES           SIMP. VAR.     INTEGER     Q +001
NUMINPUTFILEBIT      EQUATE                                 VALUE = %4
NUMINPUTFILES        SIMP. VAR.     INTEGER     Q -020
NUMKEYS              SIMP. VAR.     INTEGER     Q -014
NUMKEYSBIT           EQUATE                                 VALUE = %10
NUMRECS              SIMP. VAR.     DOUBLE      Q +002
OPTIONS              SIMP. VAR.     LOGICAL     Q -004
ORIGINALZ            SIMP. VAR.     INTEGER     Q +013
OUTPUTBUFFER         POINTER        INTEGER     Q +041
OUTPUTFILE           SIMP. VAR.     INTEGER     Q -016
OUTPUTFILEBIT        EQUATE                                 VALUE = %6
OUTPUTLOOP           LABEL                      PB+1500
PACKEDCODE           EQUATE                                 VALUE = %3
PACKEDMINUS          EQUATE                                 VALUE = %15
PLUSZERO             EQUATE                                 VALUE = %173
POSTPROCESSBIT       EQUATE                                 VALUE = %13
POSTPROCESSOR        PROCEDURE                  Q -011
PREPROCESSOR         PROCEDURE                  Q -012
PREPROCESSORBIT      EQUATE                                 VALUE = %12
Q0                   SIMP. VAR.     INTEGER     Q +000
```

R1	POINTER	BYTE	Q +050	
R1SIGN	SIMP. VAR.	INTEGER	Q +021	
R2	POINTER	BYTE	Q +051	
R2SIGN	SIMP. VAR.	INTEGER	Q +022	
RECLEN	SIMP. VAR.	INTEGER	Q +027	
RIGHT	DEFINE			(12:4)
S0	SIMP. VAR.	INTEGER	S -000	
SPACEAVAIL	SIMP. VAR.	INTEGER	Q +004	
SPACEPERFILE	SIMP. VAR.	INTEGER	Q +031	
STACKFUDGE	EQUATE			VALUE = %1750
STAT	SIMP. VAR.	INTEGER	Q -001	
STATISTICS	ARRAY	INTEGER	Q -006	
STATISTICSBIT	EQUATE			VALUE = %16
T	SIMP. VAR.	INTEGER	Q +016	
TOTALKEYS	SIMP. VAR.	INTEGER	Q +015	
W	SIMP. VAR.	INTEGER	Q +023	
W1	SIMP. VAR.	INTEGER	Q +024	
W2	SIMP. VAR.	INTEGER	Q +025	
WINNER	POINTER	INTEGER	Q +044	
WORDSPERRECORD	SIMP. VAR.	INTEGER	Q +030	
X	SIMP. VAR.	INTEGER	XREG	

```
00000  041701 071702 004300 047444 051424 000400 047444 051425    00010  131425 047443 051440 043440 022000 145136 131424 047443
00020  051437 043437 022000 145131 151405 034030 001100 161405    00030  041437 010201 003300 051450 041440 010201 003300 051451
00040  041432 013717 000600 021401 177450 041437 177451 041440    00050  041607 031000 142005 000636 000641 000000 000001 000631
00060  000600 051417 131417 047435 051420 047613 004300 177450    00070  051450 177451 051451 131417 047436 037777 004300 144001
00100  140035 140005 140025 140052 140331 140050 152450 010210    00110  152451 010210 001700 145511 021401 177450 177451 041420
00120  003400 020263 145503 142003 000554 000553 000530 152450    00130  022200 141614 152451 022200 141611 041450 041451 041420
00140  020263 145502 142002 000535 000512 041451 041450 041420    00150  020263 145502 142002 000525 000502 004400 051421 041420
00160  003443 156450 037417 022015 145516 156451 037417 022015    00170  145515 041421 022003 141513 041451 041450 041420 020263
00200  145503 142003 000031 000475 000452 000025 041420 022001    00210  141320 152451 037417 152450 037417 001700 141510 021401
00220  177451 177450 041420 003400 020263 141203 142001 000451    00230  142001 000425 140061 156451 037417 022015 145220 041421
00240  022003 141520 041450 041451 004400 020263 141507 156450    00250  026604 156451 026604 001700 141204 142002 000035 000421
00260  140376 041420 022001 141327 152450 037417 152451 037417    00270  001700 141520 021401 177450 177451 041420 023002 020263
00300  141511 041420 003443 156450 026604 156451 026604 001700    00310  141202 140367 140344 041421 022003 141537 156450 026604
00320  022000 141530 156451 026604 022000 141524 041420 022001    00330  141220 041450 041451 004400 020263 141514 152450 022000
00340  141511 041450 021401 177450 041420 023002 020263 141502    00350  140306 140053 000304 000051 041420 022001 145244 156450
00360  026604 022000 145547 156451 026604 022000 141536 152450    00370  037417 022000 141532 152451 037417 022000 141526 021401
00400  177450 177451 041420 023002 020263 141517 041420 022002    00410  141213 156450 022000 141511 177450 021402 177450 041420
00420  023003 020263 141502 140233 041420 003443 156450 037417    00430  022015 000600 141502 003400 140247 041420 003443 156450
00440  051421 156451 051422 041421 022112 145124 041421 022173    00450  145222 041422 022112 141134 041422 022173 141231 041451
00460  041450 004400 020263 145510 041422 022175 141506 000600    00470  140005 000055 000054 000205 041422 041421 022175 141503
00500  000600 140002 041421 001700 145502 140151 000172 041421    00510  022175 141533 041422 022173 141204 041422 022060 141525
00520  041420 022001 141221 152450 022060 141517 041450 021401    00530  177450 041420 023002 020263 141510 041450 041451 041420
00540  003400 020263 141502 140113 021001 140136 041422 022112    00550  141104 041422 022173 145513 041450 041451 004400 020263
00560  145507 041421 022101 141605 041421 140012 000031 000111    00570  041421 022173 141503 021060 140003 041421 023020 041422
00600  022101 141603 041422 140010 041422 022173 141503 021060    00610  140003 041422 023020 001700 145502 140041 000062 041422
00620  022175 141533 041421 022173 141204 041421 022060 141525    00630  041420 022001 141221 152451 022060 141517 041451 021401
00640  177451 041420 023002 020263 141510 041450 041451 041420    00650  003400 020263 141502 140003 000600 140026 131417 047613
00660  002443 177450 051450 177451 051451 041417 022403 004500    00670  051417 061614 145505 041437 061440 141114 140017 177163
00700  000600 141602 003300 131417 047436 022377 141302 006400    00710  013705 041424 131702 057444 140004 041425 131702 057444
00720  032001 041604 026701 013705 171701 041610 031000 140012    00730  041701 041434 010201 171420 000000 041434 041420 002406
00740  000000 025001 051433 142001 000002 032001 172002 032000    00750  000714 041702 000000 041701 170002 140434 032002 041701
00760  111431 071442 131702 004500 057443 051437 041437 071431    00770  051440 000600 131702 047617 021401 177437 041426 002400
01000  000000 053437 141212 141304 025002 053437 032001 131701    01010  047617 021014 170002 140442 043437 061414 141606 131701
```

```
01020  047617 021016 170002 140452 041604 026641 013710 171701    01030  041437 003300 010201 041437 041612 031000 043437 022403
01040  010301 071437 051437 041440 101437 061430 145404 025001    01050  053437 032001 177717 035001 000700 035001 000700 035005
01060  000700 035010 000600 035003 000700 035017 171700 051434    01070  035043 024420 051413 041604 026741 013707 000700 000000
01100  161407 000700 000000 161411 041604 026501 013604 021001    01110  170002 140570 041620 004500 051401 022000 141404 021002
01120  170002 140600 041604 026521 013604 021003 170002 140606    01130  041604 026541 013607 041604 026661 013604 021004 170002
01140  140617 041604 026561 013603 000600 051615 041604 026721    01150  013723 041607 022000 141220 041615 013704 021017 170002
01160  140637 041604 026602 022000 141204 021005 170002 140646    01170  025001 051432 140105 041604 026602 022003 141204 021006
01200  170002 140660 021001 040006 131614 012605 021007 170003    01210  140667 023420 041614 023403 051614 041613 003300 051435
01220  041613 022402 051436 000300 047613 022000 141310 047435    01230  022000 141305 040007 107613 067435 141605 021010 170003
01240  140717 077777 047613 077435 003445 051416 061414 141303    01250  041416 051414 047436 026407 022000 141204 021011 170002
01260  140737 047436 037777 022005 141304 021012 170002 140746    01270  041415 077435 051415 032403 004400 061614 145507 000300
01300  021001 041620 003400 054003 140045 177717 047617 035003    01310  171427 035017 034007 000000 141207 047617 021013 170004
01320  140747 000010 100000 041427 022000 141604 041427 002400    01330  140003 041427 010201 051427 041427 061414 141605 021010
01340  170003 142001 177357 041427 061426 141303 041427 051426    01350  056442 041426 022403 010301 051430 041604 026541 013714
01360  041615 013712 041415 003300 010301 171700 051441 171700    01370  010201 051446 035000 171700 051444 041620 010201 003400
01400  035000 041620 171700 051443 035000 000600 040021 000000    01410  040020 002100 171701 002100 051404 041404 041620 002340
01420  051431 041431 061430 141407 021020 170005 142003 077777    01430  001750 177270 041620 111431 171700 051442 035000 000600
01440  051416 171416 021001 041620 003400 050002 140013 041416    01450  170003 142001 177305 041416 071620 041416 003243 057444
01460  052411 041620 003400 051416 171416 025001 021001 050002    01470  140010 041416 170005 142001 176304 000146 000107 052406
01500  021401 047444 004500 051423 004300 047443 051445 043445    01510  022000 145154 151402 034054 001100 161402 041604 026661
01520  013707 041445 003300 010201 041445 041611 031000 041604    01530  026541 017773 041615 013736 041445 010201 003300 051447
01540  000600 051417 041446 131417 047613 004300 177447 131417    01550  047435 020062 041417 022403 004500 051417 061614 141554
01560  004000 041616 041441 041415 002406 000000 140011 000000    01570  000001 041616 021401 177445 043445 002406 000011 141206
01600  041616 021015 170003 142001 177145 043445 022403 010301    01610  071445 004500 051445 131423 057443 043445 026001 141505
01620  041423 170003 142001 177134 041423 071620 051423 041423    01630  010301 004500 051423 022000 141306 041423 170003 142001
01640  176140 140412 140542 041604 026741 013717 000700 000550    01650  151407 001200 161407 000700 000552 151411 001200 161411
01660  173606 171401 021012 020023 041604 037401 013703 041433    01670  053605 041601 041433 013703 021001 140002 021002 027142
01700  051601 131413 171600 027401 000644 000276 004000 031415
```

```
00687000  00000 1   $CONTROL SEGMENT=MERGE
00688000  00000 1   INTEGER PROCEDURE GETERRORMESS(BUF,N);
00689000  00000 1   VALUE N;
00690000  00000 1   INTEGER N;
00691000  00000 1   BYTE ARRAY BUF;
00692000  00000 1
00693000  00000 1   <<FILL BUF WITH ERROR MESSAGE N, AND RETURN THE LENGTH AS A RESULT>>
00694000  00000 1
00695000  00000 1   BEGIN
00696000  00000 2   <<MESSAGES CONTAINS ALL THE MESSAGES, THE FORMAT IS THE SAME AS FOR
00697000  00000 2     MERGEERRORMESS>>
00698000  00000 2
00699000  00000 2   BYTE ARRAY MESSAGES(0:0)=PB:=
00700000  00000 2   << 1>> 29,"FAILURE ON FOPEN OF LIST FILE"
00701000  00017 2   << 2>>,22,"LIST FILE IS READ-ONLY"
00702000  00032 2   << 3>>,29,"FAILURE ON FOPEN OF TEXT FILE"
00703000  00051 2   << 4>>,23,"TEXT FILE IS WRITE-ONLY"
00704000  00065 2   << 5>>,15,"ILLEGAL COMMAND"
00705000  00075 2   << 6>>,22,"NO KEYS WERE SPECIFIED"
00706000  00111 2   << 7>>,36,"FILENAME CANNOT EXCEED 35 CHARACTERS"
00707000  00133 2   << 8>>,13,"MISSING COMMA"
00708000  00142 2   << 9>>,17,"MISSING PARAMETER"
00709000  00153 2   <<10>>,25,"ILLEGAL NUMBER OF RECORDS"
00710000  00170 2   <<11>>,40,"NUMBER OF RECORDS TOO LARGE OR TOO SMALL"
00711000  00215 2   <<12>>,19,"TOO MANY PARAMETERS"
00712000  00227 2   <<13>>,24,"INSUFFICIENT STACK SPACE"
00713000  00243 2   <<14>>,30,"FAILURE ON FOPEN OF INPUT FILE"
00714000  00263 2   <<15>>,16,"ILLEGAL POSITION"
00715000  00273 2   <<16>>,21,"POSITION OUT OF RANGE"
00716000  00306 2   <<17>>,19,"LENGTH OUT OF RANGE"
00717000  00320 2   <<18>>,31,"LENGTH PARAMETER NOT AN INTEGER"
00718000  00340 2   <<19>>,53,"LENGTH NOT SPECIFIED FOR TYPE BYTE, PACKED OR DISPLAY"
00719000  00373 2   <<20>>,12,"MISSING DESC"
00720000  00402 2   <<21>>,24,"INPUT FILE IS WRITE-ONLY"
00721000  00416 2   <<22>>,31,"FAILURE ON FOPEN OF OUTPUT FILE"
00722000  00436 2   <<23>>,24,"OUTPUT FILE IS READ-ONLY"
00723000  00453 2   <<24>>,29,"NO INPUT FILES WERE SPECIFIED"
00724000  00472 2   <<25>>,32,"FAILURE ON FCLOSE OF OUTPUT FILE"
00725000  00512 2   <<26>>,31,"SUM OF KEYFIELD SIZES TOO LARGE"
00726000  00532 2   <<27>>,35,"FAILURE ON PURGE OF OLD OUTPUT FILE"
00727000  00554 2   <<28>>,35,"FAILURE ON FOPEN OF OLD OUTPUT FILE"
00728000  00576 2   <<29>>,33,"FAILURE ON FRENAME OF OUTPUT FILE"
00729000  00617 2   <<30>>,32,"FAILURE ON FWRITE OF PROMPT FILE"
00730000  00640 2   <<31>>,29,"FAILURE ON FREAD OF TEXT FILE"
00731000  00657 2   ;
00732000  00657 2
00733000  00657 2     BYTE I;           <<USED FOR EXTRACTING BYTES FROM MESSAGES>>
00734000  00657 2     TOS:=@I; TOS:=@MESSAGES;
00735000  00664 2     X:=N;             <<SKIP TO THE NTH MESSAGE>>
00736000  00665 2     WHILE (X:=X-1)>0 DO
00737000  00667 2     BEGIN
00738000  00667 3       MOVE *:=* PB,(1),1;      <<READ MESSAGE LENGTH>>
00739000  00671 3       TOS:=TOS+LOGICAL(I);     <<SKIP MESSAGE>>
00740000  00672 3       ASSEMBLE(DECB);          <<ADJUST I POINTER>>
00741000  00674 3     END;
00742000  00676 2   <<NOW TOS POINTS TO MESSAGE>>
00743000  00676 2     MOVE *:=* PB,(1),1;
```

```
00744000  00700 2      S1:=@BUF;
00745000  00702 2      MOVE *:=* PB,(GETERRORMESS:=I);
00746000  00706 2    END <<GETERRORMESS>>;
```

IDENTIFIER	CLASS	TYPE	ADDRESS
BUF	ARRAY	BYTE	Q -005
I	SIMP. VAR.	BYTE	Q +001
MESSAGES	ARRAY	BYTE	PB+000
N	SIMP. VAR.	INTEGER	Q -004

```
00000  016506 040511 046125 051105 020117 047040 043117 050105   00010  047040 047506 020114 044523 052040 043111 046105 013114
00020  044523 052040 043111 046105 020111 051440 051105 040504   00030  026517 047114 054435 043101 044514 052522 042440 047516
00040  020106 047520 042516 020117 043040 052105 054124 020106   00050  044514 042427 052105 054124 020106 044514 042440 044523
00060  020127 051111 052105 026517 047114 054417 044514 046105   00070  043501 046040 041517 046515 040516 042026 047117 020113
00100  042531 051440 053505 051105 020123 050105 041511 043111   00110  042504 022106 044514 042516 040515 042440 041501 047116
00120  047524 020105 054103 042505 042040 031465 020103 044101   00130  051101 041524 042522 051415 046511 051523 044516 043440
00140  041517 046515 040421 046511 051523 044516 043440 050101   00150  051101 046505 052105 051031 044514 046105 043501 046040
00160  047125 046502 042522 020117 043040 051105 041517 051104   00170  051450 047125 046502 042522 020117 043040 051105 041517
00200  051104 051440 052117 047440 046101 051107 042440 047522   00210  020124 047517 020123 046501 046114 011524 047517 020115
00220  040516 054440 050101 051101 046505 052105 051123 014111   00230  047123 052506 043111 041511 042516 052040 051524 040503
00240  045440 051520 040503 042436 043101 044514 052522 042440   00250  047516 020106 047520 042516 020117 043040 044516 050125
00260  052040 043111 046105 010111 046114 042507 040514 020120   00270  047523 044524 044517 047025 050117 051511 052111 047516
00300  020117 052524 020117 043040 051101 047107 042423 046105   00310  047107 052110 020117 052524 020117 043040 051101 047107
00320  042437 046105 047107 052110 020120 040522 040515 042524   00330  042522 020116 047524 020101 047040 044516 052105 043505
00340  051065 046105 047107 052110 020116 047524 020123 050105   00350  041511 043111 042504 020106 047522 020124 054520 042440
00360  041131 052105 026040 050101 041513 042504 020117 051040   00370  042111 051520 046101 054414 046511 051523 044516 043440
00400  042105 051503 014111 047120 052524 020106 044514 042440   00410  044523 020127 051111 052105 026517 047114 054437 043101
00420  044514 052522 042440 047516 020106 047520 042516 020117   00430  043040 047525 052120 052524 020106 044514 042430 047525
00440  052120 052524 020106 044514 042440 044523 020122 042501   00450  042055 047516 046131 016516 047440 044516 050125 052040
00460  043111 046105 051440 053505 051105 020123 050105 041511   00470  043111 042504 020106 040511 046125 051105 020117 047040
00500  043103 046117 051505 020117 043040 047525 052120 052524   00510  020106 044514 042437 051525 046440 047506 020113 042531
00520  043111 042514 042040 051511 055105 051440 052117 047440   00530  046101 051107 042443 043101 044514 052522 042440 047516
00540  020120 052522 043505 020117 043040 047514 042040 047525   00550  052120 052524 020106 044514 042443 043101 044514 052522
00560  042440 047516 020106 047520 042516 020117 043040 047514   00570  042040 047525 052120 052524 020106 044514 042441 043101
00600  044514 052522 042440 047516 020106 051105 047101 046505   00610  020117 043040 047525 052120 052524 020106 044514 042440
00620  043101 044514 052522 042440 047516 020106 053522 044524   00630  042440 047506 020120 051117 046520 052040 043111 046105
00640  016506 040511 046125 051105 020117 047040 043122 042501   00650  042040 047506 020124 042530 052040 043111 046105 035001
00660  171401 010201 172013 010201 131604 000500 141310 021001   00670  020041 150401 006000 007400 140407 177103 021001 020041
00700  041605 051702 150401 004500 051606 020043 031402
```

```
00748000  00000 1   PROCEDURE TRAP;
00749000  00000 1
00750000  00000 1   <<THIS IS THE TRAP ROUTINE WHICH IS CALLED WHEN CONTROL-Y IS HIT>>
00751000  00000 1
00752000  00000 1   BEGIN
00753000  00000 2      INTEGER EXITPARAM=Q+1;      <<PASSED IN BY MPE>>
00754000  00000 2      ARRAY WUF(*)=Q+2;           <<ARRAY FOR PRINTING>>
00755000  00000 2      BYTE ARRAY BUF(*)=WUF;
00756000  00000 2
00757000  00000 2      IF DEBUGGING THEN DEBUG ELSE
00758000  00004 2      BEGIN <<PRINT STATUS INFORMATION>>
00759000  00004 3         ASSEMBLE(ADDS 36);          <<FOR WUF>>
00760000  00005 3         MOVE BUF(DASCII(NUMRECSPTR,10,BUF));=
00761000  00016 3                  " RECORDS HAVE BEEN OUTPUT",2;
00762000  00041 3         PRINT(WUF,0,0);     <<SKIP A LINE>>
00763000  00044 3         PRINT(WUF,LOGICAL(@BUF)-LOGICAL(S0),0);
00764000  00052 3      END;
00765000  00052 2      RESETCONTROL;        <<ALLOW CONTROL Y AGAIN>>
00766000  00053 2      TOS:=EXITPARAM;        <<EXIT PROPERLY>>
00767000  00054 2      ASSEMBLE(ADDM *+2; XEQ 0);
00768000  00056 2   END <<TRAP>>;
```

IDENTIFIER	CLASS	TYPE	ADDRESS
BUF	ARRAY	BYTE	Q +002
EXITPARAM	SIMP, VAR,	INTEGER	Q +001
WUF	ARRAY	LOGICAL	Q +002

```
00000  041000 013703 000000 140047 035044 000600 153001 021012    00010  171402 010201 000000 004300 171402 010201 003700 170003
00020  010201 140016 020122 042503 047522 042123 020110 040526    00030  042440 041105 042516 020117 052524 050125 052040 021031
00040  020042 171402 000700 000000 171402 171402 010201 041702    00050  006106 000006 000000 041401 070002 030140 031400
```

```
00770000  00000 1   PROCEDURE MERGEMAIN;
00771000  00000 1
00772000  00000 1   <<THIS IS THE MAIN PROGRAM FOR STANDALONE MERGE>>
00773000  00000 1
00774000  00000 1   BEGIN
00775000  00000 2   BYTE ARRAY TYPES(0:41)=PB1="" BYTE   INT    REAL  PACKED DISPLAYPACKED*";
00776000  00025 2   LABEL NEXTCOMMAND;        <<RECOGNIZED BY SUBROUTINE SYNTAXERROR>>
00777000  00025 2
00778000  00025 2   <<THE FOLLOWING ARE THE STATISTICS GATHERED FROM THE MERGE PROCEDURE.>>
00779000  00025 2
00780000  00025 2   ARRAY STATS(0:NUMSTATS)=Q;
00781000  00025 2   INTEGER
00782000  00025 2      NUMINPUTFILES=STATS,
00783000  00025 2      ACTUALNUMRECS=NUMINPUTFILES+1, <<ELEMENTS OF STATS>>
00784000  00025 2      SPACEAVAIL=ACTUALNUMRECS+2,
00785000  00025 2      NUMCOMPARES=SPACEAVAIL+1,
00786000  00025 2      CPUTIME=NUMCOMPARES+2,
00787000  00025 2      ELAPSEDTIME=CPUTIME+2;
00788000  00025 2   DOUBLE
00789000  00025 2      D'CPUTIME=CPUTIME,
00790000  00025 2      D'ELAPSEDTIME=ELAPSEDTIME;
00791000  00025 2
00792000  00025 2   INTEGER
00793000  00025 2      LISTFILE,                 <<FILENUMBER OF "LIST" FILE>>
00794000  00025 2      TEXTFILE,                 <<FILENUMBER OF "TEXT" FILE>>
00795000  00025 2      PROMPTFILE,               <<FILENUMBER OF "PROMPT" FILE>>
00796000  00025 2      OUTPUTFILE,               <<FILENUMBER OF "OUTPUT" FILE>>
00797000  00025 2      OUTPUTNUMBER:=-1,         <<# USED TO GENERATE FILE NAMES>>
00798000  00025 2      LISTRECLEN,               <<-# OF BYTES IN MAX LISTFILE RECORD>>
00799000  00025 2      TEXTRECLEN,               <<-# OF BYTES IN MAX TEXTFILE RECORD>>
00800000  00025 2      OUTPUTRECLEN,             <<-# OF BYTES IN MAX OUTPUTFILE RECORD>>
00801000  00025 2      OUTPUTFNAMELEN:=7,        <<LENGTH OF OUTPUT FILENAME + 1>>
00802000  00025 2      OUTPUTFOPTIONS,           <<OUTPUT FILE FOPTIONS>>
00803000  00025 2      TEMPNUMINFILES,           <<USED IN INPUT COMMAND TO COUNT FILES>>
00804000  00025 2      OUTPUTFILECODE,           <<FILE CODE FOR OUTPUT FILE>>
00805000  00025 2      OUTPUTBLOCKFAC,           <<BLOCK FACTOR FOR OUTPUT FILE>>
00806000  00025 2      RECLEN,                   <<INPUT FILE RECORD LENGTH>>
00807000  00025 2      TOTALKEYS,                <<SUM OF KEYFIELD LENGTHS>>
00808000  00025 2      FILECODE,                 <<INPUTFILE FILE CODE>>
00809000  00025 2      BLKSIZE,                  <<INPUT FILE BLOCKSIZE>>
00810000  00025 2      BLOCKFACTOR;              <<INPUT FILE BLOCK FACTOR>>
00811000  00025 2   DOUBLE
00812000  00025 2      USERNUMRECS:=0D,          <<NUMBER OF RECORDS SPECIFIED IN OUT COM>>
00813000  00025 2      NUMRECS,                  <<COMPUTED SUM OF ALL FLIMITS>>
00814000  00025 2      FLIMIT,                   <<FILE LIMIT OF INPUT FILES>>
00815000  00025 2      EOF:=0D;                  <<NUMBER OF RECORDS PASSED TO MERGEINITIAL>>
00816000  00025 2   LOGICAL
00817000  00025 2      LISTTEXTRELATED,          <<FRELATE(TEXTFILE,LISTFILE)>>
00818000  00025 2      FIRSTLINE,                <<TRUE IF 1ST LINE OF A COMMAND>>
00819000  00025 2      PARAM,                    <<BIT ARRAY USED TO MARK PARAMETERS
00820000  00025 2                                  SPECIFIED IN COMMANDS>>
00821000  00025 2      KEYSONLY:=FALSE,          <<TRUE IF "KEY" SPECIFIED IN OUTPUT COM>>
00822000  00025 2      ENDOFTEXT:=FALSE;         <<TRUE WHEN EOF ON TEXT FILE>>
00823000  00025 2   ARRAY
00824000  00025 2      WUF(-1:40);                <<LIST/TEXT BUFFER>>
00825000  00025 2   INTEGER POINTER
00826000  00025 2      INPUTFILES;               <<ARRAY OF INPUT FILE NUMBERS>>
```

```
00827000  00025 2      KEYS,                    <<ARRAY OF 3-WORD KEY DESCRIPTORS>>
00828000  00025 2      LASTKEY;                 <<LAST WORD OF ARRAY KEYS>>
00829000  00025 2    BYTE ARRAY
00830000  00025 2      INITINFILES(*)=DB+0,     <<MARKS END OF INPUTFILE NAMES>>
00831000  00025 2      OUTPUTFILENAME(0:35),    <<OUTPUT FILE FORMAL DESIGNATOR>>
00832000  00025 2      TEMPFILENAME(0:35);      <<TEMP FOR EITHER OF THE ABOVE>>
00833000  00025 2    BYTE ARRAY
00834000  00025 2      BUF(*)=WUF;              <<BYTE VERSION OF WUF>>
00835000  00025 2    BYTE POINTER
00836000  00025 2      INFILES,                 <<PTS TO LIST OF INPUT FILE NAMES>>
00837000  00025 2      TEMPINFILES,             <<TEMP OF INFILES DURING INPUT COMMAND>>
00838000  00025 2      BP,                      <<POINTS TO END OF CURRENT TOKEN>>
00839000  00025 2      TOKEN',                  <<POINTS TO START OF CURRENT TOKEN>>
00840000  00025 2      TOKEN;                   <<=TOKEN'-1, CONTAINS TOKEN LENGTH>>
00841000  00025 2    INTEGER AOPTIONS,FOPTIONS,L,N,I;  <<MISC. TEMPS>>
00842000  00025 2    DOUBLE TEMPNUMRECS;
00843000  00025 2    DEFINE
00844000  00025 2      ASCBIN=(13:1)#,          <<FOPTIONS BIT>>
00845000  00025 2      RECORDFORMAT=(8:2)#,     <<FOPTIONS BITS>>
00846000  00025 2      FIXEDFORMAT=0#,          <<RECORDFORMAT FIELD>>
00847000  00025 2      VARIABLEFORMAT=1#,       <<   "          "  >>
00848000  00025 2      INTERACTIVE=LISTTEXTRELATED.(15:1)#,    <<LIST/TEXT INTERACTIVE>>
00849000  00025 2      DUPLICATIVE=LISTTEXTRELATED.(0:1)#;     <<LIST/TEXT DUPLICATIVE>>
00850000  00025 2
00851000  00025 2    SUBROUTINE PR(B);
00852000  00025 2    VALUE B;
00853000  00025 2    INTEGER B;
00854000  00025 2    BEGIN <<PRINT BUF UP TO BUT NOT INCLUDING THE BYTE WHICH B POINTS TO. B
00855000  00025 3            B ACTUALLY CONTAINS A BYTE ADDRESS.>>
00856000  00025 3      IF (B:=@BUF-B)<LISTRECLEN THEN B:=LISTRECLEN;  <<-#CHARS TO PRINT>>
00857000  00035 3      IF LISTFILE<>0 THEN FWRITE(LISTFILE,WUF,B,0) ELSE PRINT(WUF,B,0);
00858000  00052 3      IF <> THEN
00859000  00053 3      BEGIN
00860000  00053 4        PRINT FILE INFO(LISTFILE);
00861000  00055 4        TERMINATE;
00862000  00056 4      END;
00863000  00056 3    END <<PR>>;
```

IDENTIFIER	CLASS	TYPE	ADDRESS
B	SIMP. VAR.	INTEGER	S -001

```
00864000  00057 2
00865000  00057 2    SUBROUTINE PRINTERROR(N);
00866000  00057 2    VALUE N;
00867000  00057 2    INTEGER N;
00868000  00057 2    BEGIN <<PRINT ERROR NUMBER N>>
00869000  00057 3      PR(@BUF(GETERRORMESS(BUF,N)));
00870000  00067 3    END <<PRINTERROR>>;
```

IDENTIFIER	CLASS	TYPE	ADDRESS
N	SIMP. VAR.	INTEGER	S -001

```
00871000  00070 2
00872000  00070 2   SUBROUTINE PRINTIOERROR(F,N);
00873000  00070 2   VALUE F,N;
00874000  00070 2   INTEGER F,N;
00875000  00070 2   BEGIN <<PRINT ERROR N PRECEDED BY A FILE INFO BLOCK FOR FILE F>>
00876000  00070 3      PRINT'FILE'INFO(F);
00877000  00072 3      PRINTERROR(N);
00878000  00075 3   END <<PRINTIOERROR>>;
```

IDENTIFIER	CLASS	TYPE	ADDRESS
F	SIMP. VAR.	INTEGER	S -002
N	SIMP. VAR.	INTEGER	S -001

```
00879000  00076 2   SUBROUTINE PROMPT(B);
00880000  00076 2   VALUE B;
00881000  00076 2   INTEGER B;
00882000  00076 2   BEGIN <<OUTPUT BUF TO THE PROMPTFILE, USING %320 CARRIAGE CONTROL. B
00883000  00076 3           IS ACTUALLY A BYTE POINTER TO THE 1ST CHAR, NOT TO BE
00884000  00076 3           PRINTED.>>
00885000  00076 3      IF PROMPTFILE <>0 THEN
00886000  00101 3      BEGIN
00887000  00101 4         FWRITE(PROMPTFILE,WUF,@BUF-B,%320);
00888000  00107 4         IF <> THEN
00889000  00110 4         BEGIN
00890000  00110 5            IF LISTFILE=PROMPTFILE THEN LISTFILE:=0;
00891000  00115 5            PRINTIOERROR(PROMPTFILE,30);
00892000  00121 5            TERMINATE;
00893000  00122 5         END;
00894000  00122 4      END;
00895000  00122 3   END <<PROMPT>>;
```

IDENTIFIER	CLASS	TYPE	ADDRESS
B	SIMP. VAR.	INTEGER	S -001

```
00896000  00123 2   SUBROUTINE SYNTAXERROR(N);
00897000  00123 2   VALUE N;
00898000  00123 2   INTEGER N;
00899000  00123 2   BEGIN <<PRINT A SYNTAX ERROR. IF IN INTERACTIVE MODE, RECOVER BY
00900000  00123 3           CUTTING BACK THE STACK TO @LASTKEY AND EXITING TO NEXTCOMMAND.
00901000  00123 3           OTHERWISE TERMINATE.>>
00902000  00123 3      PRINTERROR(N);
00903000  00126 3      IF NOT INTERACTIVE THEN TERMINATE;
00904000  00132 3      TOS:=@LASTKEY+1; SET(S);
00905000  00135 3      TOS:=@NEXTCOMMAND;
00906000  00136 3   END <<SYNTAXERROR>>;
```

IDENTIFIER	CLASS	TYPE	ADDRESS
N	SIMP. VAR.	INTEGER	S -001

```
00907000  00140 2
00908000  00140 2   SUBROUTINE READTEXT;
00909000  00140 2   BEGIN <<READ A LINE FROM THE TEXT FILE, INSERT A NULL CHARACTER AFTER
00910000  00140 3         THE ONES ACTUALLY ENTERED, SET CCG AND CLEAR INTERACTIVE IF EOF,
00911000  00140 3          OTHERWISE SET CCE.>>
00912000  00140 3      TOS:=FREAD(TEXTFILE,WUF,TEXTRECLEN);
00913000  00145 3      IF < THEN
00914000  00146 3      BEGIN <<FATAL ERROR>>
00915000  00146 4         PRINTIOERROR(TEXTFILE,31);
00916000  00152 4         TERMINATE;
00917000  00153 4      END;
00918000  00153 3      IF > THEN
00919000  00154 3      BEGIN <<END OF DATA>>
00920000  00154 4         INTERACTIVE:=FALSE;
00921000  00157 4         ENDOFTEXT:=TRUE;
00922000  00161 4         BUF:=TOS;                  <<STORE ZERO STOPPER>>
00923000  00162 4         ASSEMBLE(ZROX,INCX);    <<FORCE CCG>>
00924000  00163 4      END ELSE
00925000  00164 3      BEGIN <<INSERT A NULL AT END OF BUFFER>>
00926000  00164 4         X:=TOS; BUF(X):=0;
00927000  00166 4         IF NOT DUPLICATIVE THEN PR(@BUF(X)); <<ECHO THE INPUT>>
00928000  00174 4         ASSEMBLE(DZRO,CMP);          <<FORCE CCE>>
00929000  00175 4      END;
00930000  00175 3   END <<READTEXT>>;
00931000  00176 2
00932000  00176 2   SUBROUTINE SCANIT;
00933000  00176 2
00934000  00176 2   <<SCAN THE NEXT SYMBOL IN THE TEXT BUFFER, A SYMBOL IS A SEQUENCE
00935000  00176 2     OF NONBLANK CHARACTERS DEFINED AS FOLLOWS:
00936000  00176 2        A SEMICOLON
00937000  00176 2        A COMMA
00938000  00176 2        A SEQUENCE OF CHARACTERS NOT INCLUDING COMMA, SEMICOLON,
00939000  00176 2          BLANK,
00940000  00176 2     THE POINTER TOKEN' IS SET TO THE 1ST CHARACTER, THE POINTER BP IS
00941000  00176 2     SET TO THE 1ST CHARACTER FOLLOWING, THE POINTER TOKEN IS SET TO
00942000  00176 2     TOKEN-1, AND THAT CHARACTER IS SET TO THE LENGTH OF THE SYMBOL,
00943000  00176 2     CCE IS SET IF THE END OF LINE IS REACHED, ELSE CCG,
00944000  00176 2     SCANIT ALSO HANDLES CONTINUATIONS, UPSHIFTING, AND ECHOING,
00945000  00176 2   >>
00946000  00176 2   BEGIN
00947000  00176 3   START:
00948000  00176 3      SCAN BP WHILE " ",1; @TOKEN':=TOS;
00949000  00202 3      IF CARRY THEN @BP:=@TOKEN' ELSE
00950000  00206 3      BEGIN <<NOT END OF LINE>>
00951000  00206 4         IF TOKEN'="&" THEN
00952000  00211 4         BEGIN <<CHECK FOR CONTINUATION>>
00953000  00211 5            SCAN TOKEN'(1) WHILE " ";
00954000  00215 5            IF CARRY THEN
00955000  00216 5            BEGIN <<READ THE NEXT LINE>>
00956000  00216 6               MOVE BUF:=">>";      <<SET UP PROMPT>>
00957000  00225 6               PROMPT(@BUF(IF FIRSTLINE THEN 1 ELSE 2));
00958000  00236 6               FIRSTLINE:=FALSE;
00959000  00240 6               READTEXT; @BP:=@BUF;
00960000  00244 6               GO TO START;
00961000  00245 6            END <<READ LINE>>;
00962000  00245 5         END <<&>>;
00963000  00245 4         IF TOKEN'="," OR TOKEN'=";" THEN @BP:=@TOKEN'+1 ELSE
```

```
00964000  00257 4          BEGIN <<NOT A SPECIAL, UPSHIFT UNTIL END>>
00965000  00257 5             TOS:=TOS:=@TOKEN';
00966000  00260 5   L;        MOVE *:=* WHILE ANS,0;
00967000  00262 5             @BP:=TOS;
00968000  00263 5             IF BP<>"," AND BP<>";" AND BP<>0 AND BP<>"&" AND BP<>" " THEN
00969000  00302 5             BEGIN TOS:=TOS:=TOS+1;          <<SKIP OVER SPECIAL>>
00970000  00303 6                GO TO L;
00971000  00304 6             END;
00972000  00304 5             DEL;                    <<KILL HANGING POINTER>>
00973000  00304 5          END;
00974000  00305 4       END;
00975000  00305 3       @TOKEN:=@TOKEN'-1;
00976000  00310 3       TOKEN:=@BP-@TOKEN';          <<STORE LENGTH & SET CC>>
00977000  00313 3    END <<SCANIT>>;
```

IDENTIFIER	CLASS	TYPE	ADDRESS
L	LABEL		PB+261
START	LABEL		PB+176

```
00979000  00314 2      MERGETITLE;
00980000  00347 2
00981000  00347 2   <<INITIALIZE INFILES POINTER>>
00982000  00347 2
00983000  00347 2      @INFILES:=NUMINPUTFILES:=0;
00984000  00352 2
00985000  00352 2   <<INITIALIZE THE OUTPUT FILE FORMAL DESIGNATOR>>
00986000  00352 2
00987000  00352 2      MOVE OUTPUTFILENAME:="OUTPUT=";
00988000  00364 2
00989000  00364 2   <<OPEN THE LIST FILE. PARAMETERS ARE:
00990000  00364 2      FORMAL DESIGNATOR: "LIST"
00991000  00364 2      FOPTIONS: OLD/OLDTEMP, ASCII, $STDLIST, VARIABLE-LENGTH RECORDS,
00992000  00364 2                CCTL, ALLOW FILE EQUATION
00993000  00364 2      AOPTIONS: WRITE-ACCESS ONLY
00994000  00364 2      ALL OTHER PARAMETERS: DEFAULT
00995000  00364 2   >>
00996000  00364 2
00997000  00364 2      MOVE BUF:="LIST=";
00998000  00375 2      LISTFILE:=FOPEN(BUF,%517,1);
00999000  00405 2      IF <> THEN
01000000  00406 2      BEGIN <<CAN'T OPEN LISTFILE>>
01001000  00406 3         PRINTIOERROR(0,1);
01002000  00414 3         TERMINATE;
01003000  00415 3      END;
01004000  00415 2
01005000  00415 2   <<GET THE RECORD SIZE AND CHECK FOR WRITE ACCESS>>
01006000  00415 2
01007000  00415 2      FGETINFO(LISTFILE,,,AOPTIONS,LISTRECLEN);
01008000  00424 2      IF LISTRECLEN>0 THEN LISTRECLEN:=-LISTRECLEN&ASL(1);
01009000  00433 2      IF AOPTIONS.(12:4)=0 THEN
01010000  00437 2      BEGIN <<READ-ONLY FILE. FATAL ERROR.>>
01011000  00437 3         LISTFILE:=0;          <<AVOID PRINTING HERE>>
01012000  00441 3         PRINTERROR(2);
01013000  00446 3         TERMINATE;
01014000  00447 3      END;
01015000  00447 2
01016000  00447 2   <<OPEN THE TEXTFILE. PARAMETERS ARE:
01017000  00447 2      FORMAL DESIGNATOR: "TEXT"
01018000  00447 2      FOPTIONS: OLD/OLDTEMP, ASCII, $STDIN, FIXED-LENGTH RECORDS,
01019000  00447 2                ALLOW FILE EQUATION
01020000  00447 2      AOPTIONS: READ-ACCESS ONLY
01021000  00447 2      ALL OTHER PARAMETERS: DEFAULT
01022000  00447 2   >>
01023000  00447 2
01024000  00447 2      MOVE BUF:="TEXT=";
01025000  00460 2      TEXTFILE:=FOPEN(BUF,%57);
01026000  00467 2      IF <> THEN
01027000  00470 2      BEGIN <<CAN'T OPEN TEXT FILE>>
01028000  00470 3         PRINTIOERROR(0,3);
01029000  00476 3         TERMINATE;
01030000  00477 3      END;
01031000  00477 2
01032000  00477 2   <<GET THE RECORD SIZE AND CHECK FOR READ ACCESS.>>
01033000  00477 2
01034000  00477 2      FGETINFO(TEXTFILE,,FOPTIONS,AOPTIONS,TEXTRECLEN);
01035000  00507 2      IF TEXTRECLEN>0 THEN TEXTRECLEN:=-TEXTRECLEN&ASL(1);
```

```
01036000  00516 2      IF 1<=AOPTIONS.(12:4)<=3 THEN
01037000  00527 2      BEGIN <<TEXTFILE IS WRITE-ONLY>>
01038000  00527 3         PRINTFRROR(4);
01039000  00533 3         TERMINATE;
01040000  00534 3      END;
01041000  00534 2
01042000  00534 2   <<SET DUPLICATIVE/INTERACTIVE BITS>>
01043000  00534 2
01044000  00534 2      LISTTEXTRELATED:=FRELATE(TEXTFILE,LISTFILE);
01045000  00541 2      IF INTERACTIVE THEN PROMPTFILE:=LISTFILE ELSE
01046000  00547 2      BEGIN <<OPEN $STDLIST FOR PROMPTING>>
01047000  00547 3         MOVE BUF:="PROMPT-";
01048000  00561 3         PROMPTFILE:=FOPEN(BUF,%2414,1);     <<ASCII,$STDLIST,CCTL,NO FEQ>>
01049000  00571 3         IF = AND NOT FRELATE(TEXTFILE,PROMPTFILE) THEN
01050000  00577 3         BEGIN <<PROMPTFILE & TEXTFILE NOT INTERACTIVE, NO PROMPTING>>
01051000  00577 4            FCLOSE(PROMPTFILE,0,0);
01052000  00602 4            PROMPTFILE:=0;
01053000  00604 4         END;
01054000  00604 3      END;
01055000  00604 2
01056000  00604 2   <<ALLOCATE THE LIST/TEXT BUFFER IF THE ORIGINAL 80 CHARACTERS IS
01057000  00604 2     INSUFFICIENT. THE BUFFER SIZE IS DETERMINED BY
01058000  00604 2     TEXTRECLEN. THERE IS AN ADDITIONAL CHARACTER TO THE LEFT FOR TOKEN
01059000  00604 2     TO POINT TO, AND AN ADDITIONAL ONE TO THE RIGHT TO INSERT A NULL
01060000  00604 2     SCANNING STOPPER.>>
01061000  00604 2
01062000  00604 2      IF TEXTRECLEN<=80 THEN
01063000  00607 2      BEGIN <<ALLOCATE THE BUFFER>>
01064000  00607 3         TOS:=0;              <<LEAVE A WORD TO THE LEFT FOR TOKEN>>
01065000  00607 3         TOS:=(2-TEXTRECLEN)&ASR(1);   <<# OF WORDS IN BUFFER>>
01066000  00613 3         @BUF:=(@WUF:=@S0)&LSL(1);  <<SET BUFFER POINTERS>>
01067000  00620 3         ASSEMBLE(ADDS 0);          <<ALLOCATE THE SPACE>>
01068000  00621 3      END <<ALLOCATE BUFFER>>;
01069000  00621 2
01070000  00621 2   <<SET POINTER TO KEYS TO LOCATION S+1, AS KEYS ARE ENTERED, THE INFORMA-
01071000  00621 2     TION IS ADDED TO THE STACK, 3 WORDS AT A TIME.>>
01072000  00621 2
01073000  00621 2      @KEYS:=@S0+1; @LASTKEY:=@S0;
01074000  00626 2
01075000  00626 2   <<PROCESS USER COMMANDS>>
01076000  00626 2
01077000  00626 2   NEXTCOMMAND:
01078000  00626 2
01079000  00626 2      @BP:=@BUF; FIRSTLINE:=1; WUF:=[8/"&",8/0]; <<FORCE SCANIT TO READ>>
01080000  00634 2      SCANIT;            <<READ COMMAND>>
01081000  00642 2      IF = THEN IF ENDOFTEXT THEN GO TO END'COMMAND ELSE GO TO NEXTCOMMAND;
01082000  00647 2      IF TOKEN=(5,"INPUT") THEN GO TO INPUT'COMMAND;
01083000  00661 2      IF TOKEN=(6,"OUTPUT") THEN GO TO OUTPUT'COMMAND;
01084000  00675 2      IF TOKEN=(3,"KEY") THEN GO TO KEY'COMMAND;
01085000  00707 2      IF TOKEN=(5,"RESET") THEN GO TO RESET'COMMAND;
01086000  00722 2      IF TOKEN=(6,"VERIFY") THEN GO TO VERIFY'COMMAND;
01087000  00736 2      IF TOKEN=(3,"END") THEN GO TO END'COMMAND;
01088000  00750 2      IF DEBUGGING AND TOKEN=(5,"DEBUG") THEN
01089000  00765 2      BEGIN <<SPECIAL DEBUGGING COMMAND, GOOD ONLY IF ENTRY BUGGER USED>>
01090000  00765 3         DEBUG;
01091000  00766 3         GO TO NEXTCOMMAND;
01092000  00767 3      END;
```

```
01093000  00767 2      SYNTAXERROR(5);        <<UNRECOGNIZED COMMAND>>
01094000  00773 2
01095000  00773 2   INPUT'COMMAND;
01096000  00773 2
01097000  00773 2      TEMPNUMINFILES:=0;              <<INITIALIZE # OF FILES COUNTER>>
01098000  00775 2      @TEMPINFILES:=@INFILES;         <<& PTR TO THE NAMES>>
01099000  00777 2      DO
01100000  00777 2      BEGIN <<PROCESS ANOTHER FILE NAME>>
01101000  00777 3         SCANIT; IF = OR TOKEN'="," OR TOKEN'="}" THEN SYNTAXERROR(9);
01102000  01015 3         IF TOKEN>35 THEN SYNTAXERROR(7);      <<NAME TOO LONG>>
01103000  01024 3         TEMPNUMINFILES:=TEMPNUMINFILES+1;   <<BUMP INPUT FILE COUNTER>>
01104000  01025 3         @TEMPINFILES:=@TEMPINFILES-INTEGER(TOKEN)-1; <<ROOM FOR NAME>>
01105000  01031 3         PUSH(DL);
01106000  01032 3         IF TOS>@TEMPINFILES&ASR(1) THEN
01107000  01036 3         BEGIN <<GET MORE DL SPACE>>
01108000  01036 4            DLSIZE(@TEMPINFILES&ASR(1));
01109000  01042 4            IF <> THEN SYNTAXERROR(13);
01110000  01050 4         END;
01111000  01050 3
01112000  01050 3   <<SLIDE UP THE FILENAMES SO FAR, AND INSERT THE NEW ONE>>
01113000  01050 3
01114000  01050 3         MOVE TEMPINFILES:=TEMPINFILES(INTEGER(TOKEN)+1),
01115000  01054 3                          (@INFILES-@TEMPINFILES(X)),2; <<MAKE ROOM>>
01116000  01060 3         MOVE *:=TOKEN',(TOKEN),2;          <<MOVE IN NEW FILENAME>>
01117000  01063 3         MOVE *:=",";                       <<MOVE IN TERMINATOR>>
01118000  01071 3         SCANIT;
01119000  01074 3      END UNTIL TOKEN'<>",";
01120000  01077 2      IF TOKEN'<>0 THEN SYNTAXERROR(8);
01121000  01107 2
01122000  01107 2   <<MOVE THE NEW NAMES IN OVER THE OLD ONES>>
01123000  01107 2
01124000  01107 2      MOVE INITINFILES(-1):=INFILES(-1),(@TEMPINFILES-@INFILES),2;
01125000  01117 2      @INFILES:=TOS+1;                        <<MOVE IN NEW FILE NAMES>>
01126000  01121 2      NUMINPUTFILES:=TEMPNUMINFILES;
01127000  01123 2      GO TO NEXTCOMMAND;
01128000  01124 2
01129000  01124 2   OUTPUT'COMMAND;
01130000  01124 2
01131000  01124 2      SCANIT; IF = THEN SYNTAXERROR(9);
01132000  01134 2      IF TOKEN'=","  THEN PARAM:=0 ELSE
01133000  01142 2      BEGIN <<FILENAME>>
01134000  01142 3         PARAM:=%(2)100;
01135000  01144 3         IF TOKEN>35 THEN SYNTAXERROR(7);
01136000  01153 3         MOVE TEMPFILENAME:=TOKEN',(N):=INTEGER(TOKEN)+1);
01137000  01161 3         SCANIT; IF = THEN GO TO FINOUTPUTCOM;
01138000  01165 3   OUTPUT1:
01139000  01165 3         IF TOKEN'<>"," THEN SYNTAXERROR(8);
01140000  01175 3      END;
01141000  01175 2   <<GET NUMBER OF RECORDS OR KEY>>
01142000  01175 2      SCANIT; IF = THEN SYNTAXERROR(9);
01143000  01205 2      IF TOKEN=(3,"KEY") THEN PARAM:=PARAM LOR %(2)001 ELSE
01144000  01222 2      BEGIN <<IT'S A NUMBER OF RECORDS>>
01145000  01222 3         PARAM:=PARAM LOR %(2)010;
01146000  01225 3         TEMPNUMRECS:=DBINARY(TOKEN',TOKEN);
01147000  01232 3         IF <> THEN SYNTAXERROR(IF > THEN 11 ELSE 10);
01148000  01242 3         IF TEMPNUMRECS<=0D THEN SYNTAXERROR(11);
01149000  01251 3      END <<NUMBER OF RECORDS>>;
```

```
01150000  01251 2      SCANIT; IF <> THEN GO TO OUTPUT1;
01151000  01255 2    FINOUTPUTCOM;
01152000  01255 2      KEYSONLY:=PARAM.(15:1);
01153000  01260 2      USERNUMRECS:=IF PARAM.(14:1) THEN TEMPNUMRECS ELSE 0D;
01154000  01270 2      OUTPUTFNAMELEN:=7;
01155000  01272 2      IF PARAM.(13:1) THEN
01156000  01275 2         MOVE OUTPUTFILENAME:=TEMPFILENAME,(OUTPUTFNAMELEN:=N)
01157000  01302 2            ELSE MOVE OUTPUTFILENAME:="OUTPUT-";
01158000  01316 2      GO TO NEXTCOMMAND;
01159000  01320 2
01160000  01320 2    KEY'COMMAND;
01161000  01320 2
01162000  01320 2      SCANIT; IF = THEN SYNTAXERROR(18);
01163000  01330 2      TOS:=BINARY(TOKEN',TOKEN);          <<COMPUTE POSITION>>
01164000  01334 2      IF <> THEN SYNTAXERROR(IF > THEN 16 ELSE 15);
01165000  01344 2      IF S0<=0 THEN SYNTAXERROR(16);    <<MUST BE > 0>>
01166000  01353 2      SCANIT; IF TOKEN'<>"," THEN SYNTAXERROR(8); <<NOT ENOUGH PARAMETERS>>
01167000  01365 2      SCANIT; IF = THEN SYNTAXERROR(9);  <<DANGLING COMMA>>
01168000  01375 2      IF TOKEN'=ALPHA THEN TOS:=0 ELSE
01169000  01402 2      BEGIN <<LENGTH SPECIFIED>>
01170000  01402 3         TOS:=BINARY(TOKEN',TOKEN);
01171000  01406 3         IF <> THEN SYNTAXERROR(IF > THEN 17 ELSE 18);
01172000  01416 3         IF S0<=0 THEN SYNTAXERROR(17);     <<LENGTH MUST BE POSITIVE>>
01173000  01425 3         SCANIT; IF TOKEN'<>"," THEN
01174000  01433 3         BEGIN <<DEFAULT TO BYTE>>
01175000  01433 4            TOS:=0;
01176000  01433 4            GO TO LASTKEYCHECK;
01177000  01435 4         END;
01178000  01435 3         SCANIT; IF = THEN SYNTAXERROR(9); <<DANGLING COMMA>>
01179000  01445 3      END;
01180000  01445 2    <<WE HAVE TYPE OR DESC. FIRST CHECK THE TYPES>>
01181000  01445 2      IF TOKEN=(3,"INT") THEN
01182000  01456 2      BEGIN IF S0=0 THEN TOS:=TOS+2;      <<DEFAULT LENGTH=2>>
01183000  01462 3         TOS:=1; GO TO GETDESC;           <<TYPE IS 2'S COMPLEMENT>>
01184000  01464 3      END;
01185000  01464 2      IF TOKEN=(6,"DOUBLE") THEN
01186000  01477 2      BEGIN IF S0=0 THEN TOS:=TOS+4;      <<DEFAULT LENGTH=4>>
01187000  01503 3         TOS:=1; GO TO GETDESC;           <<TYPE IS 2'S COMPLEMENT>>
01188000  01505 3      END;
01189000  01505 2      IF TOKEN=(4,"REAL") THEN
01190000  01517 2      BEGIN IF S0=0 THEN TOS:=TOS+4;      <<DEFAULT LENGTH=4>>
01191000  01523 3         TOS:=2; GOTO GETDESC;            <<TYPE IS REAL>>
01192000  01525 3      END;
01193000  01525 2      IF TOKEN=(4,"LONG") THEN
01194000  01537 2      BEGIN IF S0=0 THEN TOS:=TOS+6;      <<DEFAULT LENGTH=6>>
01195000  01543 3         TOS:=2; GO TO GETDESC;           <<TYPE IS REAL>>
01196000  01545 3      END;
01197000  01545 2      IF S0=0 THEN SYNTAXERROR(19);  <<NO LENGTH FOR BYTE,PACK,DISPLAY>>
01198000  01554 2      IF TOKEN=(4,"BYTE") THEN TOS:=0 ELSE
01199000  01570 2      IF TOKEN=(6,"PACKED") THEN TOS:=3 ELSE
01200000  01605 2      IF TOKEN=(7,"DISPLAY") THEN TOS:=4 ELSE
01201000  01622 2      IF TOKEN=(7,"PACKED*") THEN TOS:=5 ELSE
01202000  01637 2      BEGIN <<TYPE NOT SPECIFIED. USE BYTE>>
01203000  01637 3         TOS:=0; GO TO TESTDEC;
01204000  01641 3      END;
01205000  01641 2    GETDESC;
01206000  01641 2      SCANIT; IF TOKEN'<>"," THEN GO TO LASTKEYCHECK;
```

```
01207000  01647 2      SCANIT; IF = THEN SYNTAXERROR(9);
01208000  01657 2    TESTDEC;
01209000  01657 2      IF TOKEN<>(4,"DESC") THEN SYNTAXERROR(20);
01210000  01675 2      TOS:=TOS+256;          <<SET DESC BIT>>
01211000  01676 2      SCANIT;
01212000  01703 2    LASTKEYCHECK;
01213000  01703 2      IF TOKEN:="|" THEN GO TO KEY:COMMAND; <<MORE KEYS>>
01214000  01706 2      IF TOKEN:<>0 THEN SYNTAXERROR(12);    <<DANGLING JUNK>>
01215000  01716 2      @LASTKEY:=@S0;          <<MARK THE KEYS>>
01216000  01720 2      GO TO NEXTCOMMAND;
01217000  01721 2
01218000  01721 2    RESET:COMMAND;
01219000  01722 2
01220000  01722 2      SCANIT; IF <> THEN SYNTAXERROR(12);
01221000  01732 2      TOS:=@LASTKEY:=@KEYS-1; SET(S);
01222000  01736 2      GO TO NEXTCOMMAND;
01223000  01740 2
01224000  01740 2    VERIFY:COMMAND;
01225000  01740 2
01226000  01740 2      SCANIT; IF <> THEN SYNTAXERROR(12);   <<NO PARAMS ALLOWED>>
01227000  01750 2      PR(@BUF);                             <<SKIP A LINE>>
01228000  01754 2      MOVE BUF:="INPUT FILES = ",2; N:=58;    <<N IS NUM CHARS LEFT>>
01229000  01773 2      @BP:=@INFILES; I:=1;
01230000  01777 2      WHILE I<=NUMINPUTFILES DO
01231000  02002 2      BEGIN <<PROCESS ANOTHER INPUT FILE, FIRST SCAN THE NAME>>
01232000  02002 3        SCAN BP UNTIL ",",1;
01233000  02005 3        L:=TOS-@BP;             <<LENGTH OF NAME>>
01234000  02007 3        IF I<NUMINPUTFILES THEN L:=L+1;  <<INCLUDE , IF NOT LAST>>
01235000  02013 3        IF L>N THEN
01236000  02016 3        BEGIN <<CAN'T FIT ON THIS LINE>>
01237000  02016 4          PR(*);
01238000  02022 4          MOVE BUF:="                ",2; N:=58;
01239000  02041 4        END;
01240000  02041 3        MOVE *:=BP,(L),1; @BP:=TOS; N:=N-L;
01241000  02050 3        I:=I+1;
01242000  02051 3      END;
01243000  02052 2      PR(*);                      <<DUMP OUT LAST LINE>>
01244000  02055 2      MOVE BUF:="OUTPUT FILE = ",2;
01245000  02072 2        MOVE *:=OUTPUTFILENAME,(OUTPUTFNAMELEN-1),2;
01246000  02076 2      IF KEYSONLY THEN MOVE *:=",KEY",2;
01247000  02107 2      PR(*);
01248000  02112 2      IF USERNUMRECS>0D THEN
01249000  02115 2      BEGIN
01250000  02115 3        MOVE BUF:="NUMBER OF RECORDS = ";
01251000  02136 3        TOS:=@USERNUMRECS;
01252000  02137 3        INEXT:(*,-1,52,0,-1,0,BUF(20),N);    <<CONVERT TO DECIMAL>>
01253000  02150 3        SCAN BUF(20) WHILE " ",1;            <<LEFT JUSTIFY>>
01254000  02153 3        N:=@BUF(71)-S0;
01255000  02157 3        TOS:=@BUF(20); ASSEMBLE(XCH);
01256000  02162 3        MOVE *:=*,(N),2;
01257000  02164 3        PR(*);
01258000  02167 3      END <<NUMRECS>>;
01259000  02167 2      IF @LASTKEY<@KEYS THEN
01260000  02172 2      BEGIN
01261000  02172 3        MOVE BUF:="NO KEYS ENTERED",2;
01262000  02210 3        PR(*);
01263000  02213 3      END
```

```
01264000  02213 2   ELSE
01265000  02214 2       BEGIN
01266000  02214 3           MOVE BUF:="KEY POSITION   LENGTH     TYPE     ASC/DESC",2;
01267000  02247 3           PR(*);
01268000  02252 3           FOR I:=0 STEP 3 UNTIL @LASTKEY-@KEYS DO
01269000  02260 3           BEGIN <<FORMAT A LINE DESCRIBING A KEY>>
01270000  02262 4               INEXT'(KEYS(I),0,9,0,0,0,BUF,N);
01271000  02272 4               INEXT'(KEYS(I+1),0,10,0,0,0,BUF(9),N);
01272000  02304 4               MOVE BUF(19):="      ",2;
01273000  02316 4               MOVE *:=TYPES(7*KEYS(I+2).(8:8)),(7),2;
01274000  02332 4               MOVE *:="    ",2;
01275000  02342 4               IF KEYS(I+2)>255 THEN MOVE *:="DESC",2
01276000  02356 4                                ELSE MOVE *:=" ASC",2;
01277000  02367 4               IF I=0 THEN MOVE *:="   (MAJOR KEY)",2;
01278000  02406 4               PR(*);
01279000  02411 4           END;
01280000  02412 3       END <<KEYS>>;
01281000  02412 2       PR(@BUF);                <<SKIP A LINE>>
01282000  02416 2       GO TO NEXTCOMMAND;
01283000  02420 2
01284000  02420 2   END'COMMAND;
01285000  02420 2
01286000  02420 2       SCANI; IF <> THEN SYNTAXERROR(12);  <<NO PARAMS ALLOWED>>
01287000  02430 2       IF NUMINPUTFILES=0 THEN SYNTAXERROR(24);
01288000  02437 2       IF @LASTKEY<@KEYS THEN SYNTAXERROR(6); <<NO KEYS>>
01289000  02446 2   <<COMPUTE TOTAL LENGTH OF OUTPUT RECORD>>
01290000  02446 2
01291000  02446 2       TOS:=0;
01292000  02446 2       X:=@LASTKEY-@KEYS-1;    <<POINT TO LAST LENGTH FIELD>>
01293000  02452 2       DO
01294000  02452 2       BEGIN <<ADD NEXT LENGTH, GUARDING AGAINST OVERFLOW>>
01295000  02452 3           IF 32767-S0<KEYS(X) THEN SYNTAXERROR(26);
01296000  02463 3           TOS:=TOS+KEYS(X);
01297000  02464 3           X:=X-3;
01298000  02465 3       END UNTIL < ;
01299000  02466 2       TOTALKEYS:=TOS;
01300000  02467 2
01301000  02467 2   <<ALLOCATE THE INPUTFILES ARRAY ON THE STACK>>
01302000  02467 2
01303000  02467 2       @INPUTFILES:=@S0+1;
01304000  02472 2
01305000  02472 2   <<OPEN THE INPUT FILES, THE PARAMETERS USED ARE:
01306000  02472 2       FORMAL DESIGNATOR: AS SPECIFIED IN THE INPUT COMMAND
01307000  02472 2       FOPTIONS: DOMAIN=OLD OR OLDTEMP; OTHERS DEFAULT
01308000  02472 2       AOPTIONS: READ-ACCESS ONLY; OTHERS DEFAULT
01309000  02472 2       ALL OTHER PARAMETERS DEFAULT
01310000  02472 2   >>
01311000  02472 2
01312000  02472 2       @BP:=@INFILES;
01313000  02474 2       OUTPUTFOPTIONS:=OUTPUTRECLEN:=0;
01314000  02477 2       NUMRECS:=0D;
01315000  02501 2
01316000  02501 2       I:=0;
01317000  02503 2       WHILE I<NUMINPUTFILES DO
01318000  02506 2       BEGIN <<PROCESS THE I'TH FILE>>
01319000  02506 3           SCAN BP UNTIL ",",1; N:=TOS-@BP;    <<LENGTH OF NAME>>
01320000  02513 3           TOS:=FOPEN(BP,3);
```

```
01321000  02521 3          IF <> THEN
01322000  02522 3          BEGIN <<FAILURE TO OPEN I'TH FILE>>
01323000  02522 4             DEL;
01324000  02522 4             MOVE BUF:=BP,(N),2; PR(*);      <<PRINT FILE NAME>>
01325000  02534 4             PRINTIOERROR(0,14);
01326000  02541 4          <<CLOSE OTHER FILES>>
01327000  02541 4    CLOSEINPUT;
01328000  02541 4             WHILE (I:=I-1)>=0 DO FCLOSE(*,0,0);
01329000  02546 4             IF INTERACTIVE THEN GO TO NEXTCOMMAND ELSE TERMINATE;
01330000  02552 4          END <<CAN'T OPEN>>;
01331000  02552 3          FGETINFO(S0;,FOPTIONS,AOPTIONS,RECLEN,,,,FILECODE,,
01332000  02561 3             EOF,FLIMIT,,,BLKSIZE);
01333000  02571 3          IF 1<=AOPTIONS,(12:4)<=3 THEN
01334000  02603 3          BEGIN <<WRITE-ONLY FILE>>
01335000  02603 4             PRINTIOERROR(S0,21);
01336000  02610 4             I:=I+1; GO TO CLOSEINPUT;
01337000  02612 4          END <<WRITE-ONLY>>;
01338000  02612 3          BLOCKFACTOR:=BLKSIZE/RECLEN;
01339000  02616 3          IF RECLEN>0 THEN RECLEN:=-RECLEN&ASL(1);
01340000  02625 3          IF I=0 THEN
01341000  02630 3          BEGIN <<SET OUTPUT FILES OPTIONS WHICH DEPEND ON THE FIRST FILE>>
01342000  02630 4             OUTPUTFOPTIONS,ASCBIN:=FOPTIONS,ASCBIN;
01343000  02635 4             OUTPUTFILECODE:=FILECODE;
01344000  02637 4          END;
01345000  02637 3          IF FOPTIONS,RECORDFORMAT<>FIXEDFORMAT OR
01346000  02643 3                                          I>0 AND RECLEN<>OUTPUTRECLEN THEN
01347000  02651 3                OUTPUTFOPTIONS,RECORDFORMAT:=VARIABLEFORMAT;
01348000  02655 3          IF RECLEN<OUTPUTRECLEN THEN OUTPUTRECLEN:=RECLEN;
01349000  02662 3          IF I=0 THEN OUTPUTBLOCKFAC:=BLOCKFACTOR ELSE
01350000  02670 3             IF OUTPUTBLOCKFAC<>BLOCKFACTOR THEN OUTPUTBLOCKFAC:=0;
01351000  02675 3          NUMRECS:=NUMRECS+(IF FLIMIT=0D THEN 10000D ELSE EOF);
01352000  02710 3          @BP:=@BP+N+1;              <<MOVE UP POINTER>>
01353000  02714 3          I:=I+1;
01354000  02715 3       END <<I'TH INPUT FILE>>;
01355000  02717 2       IF USERNUMRECS>0D THEN NUMRECS:=USERNUMRECS;
01356000  02724 2       IF KEYSONLY THEN
01357000  02726 2       BEGIN <<SPECIAL MODS TO THE OUTPUT FILE PARAMETERS>>
01358000  02726 3          OUTPUTFOPTIONS,RECORDFORMAT:=FIXEDFORMAT;
01359000  02732 3          OUTPUTRECLEN:=-TOTALKEYS;
01360000  02735 3          OUTPUTBLOCKFAC:=0;
01361000  02737 3       END <<KEYSONLY>>;
01362000  02737 2
01363000  02737 2    <<OPEN THE OUTPUT FILE>>
01364000  02737 2
01365000  02737 2       OUTPUTFILE:=FOPEN(OUTPUTFILENAME,OUTPUTFOPTIONS,1,OUTPUTRECLEN,,,,
01366000  02744 2                         OUTPUTBLOCKFAC,,NUMRECS,,32,OUTPUTFILECODE);
01367000  02756 2       IF <> THEN
01368000  02757 2       BEGIN <<FAILURE TO OPEN OUTPUT FILE>>
01369000  02757 3          PRINTIOERROR(0,22);
01370000  02764 3          GO TO CLOSEINPUT;
01371000  02765 3       END;
01372000  02765 2       FGETINFO(OUTPUTFILE,,,AOPTIONS);
01373000  02773 2       IF AOPTIONS,(12:4)=0 THEN
01374000  02777 2       BEGIN <<READ-ONLY>>
01375000  02777 3          PRINTIOERROR(OUTPUTFILE,23);
01376000  03006 3          FCLOSE(OUTPUTFILE,0,0);
01377000  03011 3          GO TO CLOSEINPUT;
```

```
01378000  03012 3      END;
01379000  03012 2
01380000  03012 2
01381000  03012 2  <<HERE WE GO...>>
01382000  03012 2
01383000  03012 2      @NUMRECSPTR:=@S0+6+MERGEEXITPARAM;
01384000  03016 2      XCONTRAP(@TRAP,I);          <<ALLOW CONTROL Y>>
01385000  03021 2
01386000  03021 2      MERGE(NUMINPUTFILES,INPUTFILES,OUTPUTFILE,KEYSONLY,
01387000  03025 2                  (@LASTKEY+1-@KEYS)/3,KEYS,,,,,STATS);
01388000  03037 2
01389000  03037 2      XCONTRAP(0,I);              <<DISALLOW CONTROL Y>>
01390000  03042 2
01391000  03042 2  CLOSEOUT;
01392000  03042 2   <<CLOSE THE OUTPUT FILE, IF NEW, USE THE PERMANENT KRUNCH
01393000  03042 2          OPTION; OTHERWISE LEAVE AS IS.>>
01394000  03042 2      FCLOSE(OUTPUTFILE,
01395000  03043 2            IF OUTPUTFOPTIONS.(14:2)=0 THEN %11 ELSE 0,0);
01396000  03055 2      IF <> THEN
01397000  03056 2      BEGIN <<FILE NOT CLOSED, ATTEMPT RECOVERY,>>
01398000  03056 3         FCHECK(OUTPUTFILE,N);
01399000  03063 3         IF N=100 THEN
01400000  03066 3         BEGIN <<DUPLICATE NAME>>
01401000  03066 4            IF INTERACTIVE THEN
01402000  03071 4            BEGIN <<REQUEST PURGE OF OLD FILE>>
01403000  03071 5               MOVE BUF:="PURGE OLD OUTPUT FILE ";
01404000  03115 5               FGETINFO(OUTPUTFILE,BUF(22));
01405000  03123 5               X:=50; DO UNTIL BUF(X:=X-1)<>" ";
01406000  03130 5               MOVE BUF(X:=X+1):=" ? ",2; PROMPT(*);
01407000  03146 5               READTEXT; IF <> THEN GO TO BUILDNAME;
01408000  03153 5               SCAN BUF WHILE " ",1; @BP:=TOS;
01409000  03157 5               IF (LOGICAL(BP) LAND %137) = "Y" THEN
01410000  03164 5               BEGIN <<PURGE OLD FILE>>
01411000  03164 6                  FGETINFO(OUTPUTFILE,BUF);
01412000  03171 6                  N:=FOPEN(BUF,%2001,%100);    <<OLD, NO FEQ,EXCLUSIVE>>
01413000  03201 6                  IF = THEN
01414000  03202 6                  BEGIN <<CLOSE WITH DELETE>>
01415000  03202 7                     FCLOSE(N,4,0);
01416000  03206 7                     IF = THEN GO TO CLOSEOUT;
01417000  03207 7                  <<CAN'T PURGE IT>>
01418000  03207 7                     PRINTERROR(27);
01419000  03221 7                     FCLOSE(N,0,0);
01420000  03224 7                  END ELSE PRINTERROR(28); <<CAN'T OPEN IT>>
01421000  03231 6               END <<PURGE ATTEMPT>>;
01422000  03231 5            <<REQUEST NEW NAME>>
01423000  03231 5  RENAME;      MOVE BUF:="ENTER NEW NAME FOR OUTPUT FILE: ",2;
01424000  03257 5               PROMPT(*);
01425000  03262 5               READTEXT; IF <> THEN GO TO BUILDNAME;
01426000  03266 5               SCAN BUF WHILE " ",1; @BP:=TOS;
01427000  03272 5               IF BP=ALPHA THEN
01428000  03275 5               BEGIN <<NAME OK>>
01429000  03275 6                  FRENAME(OUTPUTFILE,BP);
01430000  03300 6                  IF = THEN GO TO CLOSEOUT;
01431000  03301 6               END <<RENAMING>>;
01432000  03301 5               PRINTERROR(29);
01433000  03306 5               GO TO RENAME;
01434000  03307 5            END <<INTERACTIVE CASE>>;
```

```
01435000  03307 4          <<CONSTRUCT NEW NAME BY HAND IN FORM OUTPUTNN>>
01436000  03307 4    BUILDNAME;
01437000  03307 4            IF (OUTPUTNUMBER:=OUTPUTNUMBER+1)<100 THEN
01438000  03313 4            BEGIN <<WE HAVEN'T RUN OUT OF NAMES YET>>
01439000  03313 5               MOVE BUF:="OUTPUT---";
01440000  03326 5               ASCII(OUTPUTNUMBER,10,BUF(6));
01441000  03334 5               FRENAME(OUTPUTFILE,BUF);
01442000  03340 5               GO TO CLOSEOUT;
01443000  03341 5            END;
01444000  03341 4         END <<DUPLICATE NAME RECOVERY>>;
01445000  03341 3         PRINTIOERROR(OUTPUTFILE,25);
01446000  03346 3      END <<FAILURE TO CLOSE OUTPUT FILE>>
01447000  03346 2    ELSE
01448000  03347 2       IF OUTPUTNUMBER>=0 THEN
01449000  03352 2       BEGIN <<WE RENAMED BY HAND-- PRINT WARNING>>
01450000  03352 3          MOVE BUF:="OUTPUT FILE CLOSED WITH FILENAME OUTPUT",2;
01451000  03405 3          @BP:=TOS;
01452000  03406 3          PR(@BP+ASCII(OUTPUTNUMBER,10,BP));
01453000  03417 3       END;
01454000  03417 2
01455000  03417 2    <<PRINT STATISTICS>>
01456000  03417 2
01457000  03417 2       PR(@BUF);
01458000  03423 2       MOVE BUF:="                         STATISTICS",2; PR(*);
01459000  03453 2       PR(@BUF);
01460000  03457 2       @BP:=@BUF+50;
01461000  03462 2
01462000  03462 2       MOVE WUF:="NUMBER OF INPUT FILES =";
01463000  03503 2          INEXT(NUMINPUTFILES,0,28,0,-1,0,BUF(23),I); PR(@BP);
01464000  03521 2       MOVE WUF:="NUMBER OF RECORDS =";
01465000  03540 2          INEXT(ACTUALNUMRECS,-1,32,0,-1,0,BUF(19),I); PR(@BP);
01466000  03556 2       MOVE WUF:="RECORD SIZE (IN BYTES) =";
01467000  03577 2          OUTPUTRECLEN:=-OUTPUTRECLEN;
01468000  03602 2          INEXT(OUTPUTRECLEN,0,27,0,-1,0,BUF(24),I); PR(@BP);
01469000  03620 2       MOVE WUF:="SPACE AVAILABLE (IN WORDS) =";
01470000  03643 2          INEXT(SPACEAVAIL,0,23,0,-1,0,BUF(28),I); PR(@BP);
01471000  03661 2       MOVE WUF:="NUMBER OF COMPARES =";
01472000  03700 2          INEXT(NUMCOMPARES,-1,31,0,-1,0,BUF(20),I); PR(@BP);
01473000  03716 2       MOVE WUF:="CPU TIME (MINUTES) =";
01474000  03735 2          D(CPUTIME):=REAL(D(CPUTIME)/60000.;
01475000  03742 2          INEXT(CPUTIME,1,30,2,-1,0,BUF(20),I); PR(@BP);
01476000  03762 2       MOVE WUF:="ELAPSED TIME (MINUTES) =";
01477000  04003 2          D(ELAPSEDTIME):=REAL(D(ELAPSEDTIME)/60000.;
01478000  04010 2          INEXT(ELAPSEDTIME,1,26,2,-1,0,BUF(24),I); PR(@BP);
01479000  04030 2
01480000  04030 2    END <<MERGEMAIN>>;
```

IDENTIFIER	CLASS	TYPE	ADDRESS	
ACTUALNUMRECS	SIMP. VAR.	INTEGER	Q +002	
AOPTIONS	SIMP. VAR.	INTEGER	Q +067	
ASCBIN	DEFINE			(13:1)
BLKSIZE	SIMP. VAR.	INTEGER	Q +034	
BLOCKFACTOR	SIMP. VAR.	INTEGER	Q +035	
BP	POINTER	BYTE	Q +064	
BUF	ARRAY	BYTE	Q +061	

BUILDNAME	LABEL		PB+3307	
CLOSEINPUT	LABEL		PB+2541	
CLOSEOUT	LABEL		PB+3042	
CPUTIME	SIMP. VAR.	INTEGER	Q +007	
D'CPUTIME	SIMP. VAR.	DOUBLE	Q +007	
D'ELAPSEDTIME	SIMP. VAR.	DOUBLE	Q +011	
DUPLICATIVE	DEFINE			LISTTEXTRELATED.(0:1)
ELAPSEDTIME	SIMP. VAR.	INTEGER	Q +011	
END'COMMAND	LABEL		PB+2420	
ENDOFTEXT	SIMP. VAR.	LOGICAL	Q +052	
EOF	SIMP. VAR.	DOUBLE	Q +044	
FILECODE	SIMP. VAR.	INTEGER	Q +033	
FINOUTPUTCOM	LABEL		PB+1255	
FIRSTLINE	SIMP. VAR.	LOGICAL	Q +047	
FIXEDFORMAT	DEFINE			0
FLIMIT	SIMP. VAR.	DOUBLE	Q +042	
FOPTIONS	SIMP. VAR.	INTEGER	Q +070	
GETDESC	LABEL		PB+1641	
I	SIMP. VAR.	INTEGER	Q +073	
INFILES	POINTER	BYTE	Q +062	
INITINFILES	ARRAY	BYTE	DB+000	
INPUT'COMMAND	LABEL		PB+773	
INPUTFILES	POINTER	INTEGER	Q +054	
INTERACTIVE	DEFINE			LISTTEXTRELATED.(15:1)
KEY'COMMAND	LABEL		PB+1320	
KEYS	POINTER	INTEGER	Q +055	
KEYSONLY	SIMP. VAR.	LOGICAL	Q +051	
L	SIMP. VAR.	INTEGER	Q +071	
LASTKEY	POINTER	INTEGER	Q +056	
LASTKEYCHECK	LABEL		PB+1703	
LISTFILE	SIMP. VAR.	INTEGER	Q +014	
LISTRECLEN	SIMP. VAR.	INTEGER	Q +021	
LISTTEXTRELATED	SIMP. VAR.	LOGICAL	Q +046	
N	SIMP. VAR.	INTEGER	Q +072	
NEXTCOMMAND	LABEL		PB+626	
NUMCOMPARES	SIMP. VAR.	INTEGER	Q +005	
NUMINPUTFILES	SIMP. VAR.	INTEGER	Q +001	
NUMRECS	SIMP. VAR.	DOUBLE	Q +040	
OUTPUT'COMMAND	LABEL		PB+1124	
OUTPUT1	LABEL		PB+1165	
OUTPUTBLOCKFAC	SIMP. VAR.	INTEGER	Q +030	
OUTPUTFILE	SIMP. VAR.	INTEGER	Q +017	
OUTPUTFILECODE	SIMP. VAR.	INTEGER	Q +027	
OUTPUTFILENAME	ARRAY	BYTE	Q +057	
OUTPUTFNAMELEN	SIMP. VAR.	INTEGER	Q +024	
OUTPUTFOPTIONS	SIMP. VAR.	INTEGER	Q +025	
OUTPUTNUMBER	SIMP. VAR.	INTEGER	Q +020	
OUTPUTRECLEN	SIMP. VAR.	INTEGER	Q +023	
PARAM	SIMP. VAR.	LOGICAL	Q +050	
PR	SUBROUTINE		PB+025	
PRINTERROR	SUBROUTINE		PB+057	
PRINTIOERROR	SUBROUTINE		PB+070	
PROMPT	SUBROUTINE		PB+076	
PROMPTFILE	SIMP. VAR.	INTEGER	Q +016	
READTEXT	SUBROUTINE		PB+140	
RECLEN	SIMP. VAR.	INTEGER	Q +031	
RECORDFORMAT	DEFINE			(8:2)

RENAME	LABEL		PB+3231
RESET'COMMAND	LABEL		PB+1722
SCANIT	SUBROUTINE		PB+176
SPACEAVAIL	SIMP. VAR.	INTEGER	Q +004
STATS	ARRAY	LOGICAL	Q +001
SYNTAXERROR	SUBROUTINE		PB+123
TEMPFILENAME	ARRAY	BYTE	Q +060
TEMPINFILES	POINTER	BYTE	Q +063
TEMPNUMINFILES	SIMP. VAR.	INTEGER	Q +026
TEMPNUMRECS	SIMP. VAR.	DOUBLE	Q +074
TESTDEC	LABEL		PB+1657
TEXTFILE	SIMP. VAR.	INTEGER	Q +015
TEXTRECLEN	SIMP. VAR.	INTEGER	Q +022
TOKEN	POINTER	BYTE	Q +066
TOKEN'	POINTER	BYTE	Q +065
TOTALKEYS	SIMP. VAR.	INTEGER	Q +032
TYPES	ARRAY	BYTE	PB+000
USERNUMRECS	SIMP. VAR.	DOUBLE	Q +036
VARIABLEFORMAT	DEFINE		
VERIFY'COMMAND	LABEL		PB+1740
WUF	ARRAY	LOGICAL	Q +053

```
00000  020102 054524 042440 020040 044516 052040 020040 020122   00010  042501 046040 020120 040503 045505 042040 042111 051520
00020  046101 054520 040503 045505 042052 041461 101702 004500   00030  051703 061421 141603 041421 051702 041414 022000 141207
00040  041414 041453 041703 000600 000000 140005 041453 041702   00050  000600 000000 141204 041414 000000 000000 032001 000600
00060  041461 041703 000000 004300 177461 170002 140441 032001   00070  041702 000000 041701 170002 140415 032002 041416 022000
00100  141222 041416 041453 041461 101704 021320 000000 141213   00110  041414 061416 141503 000600 051414 041416 021036 170002
00120  140430 000000 032001 041701 170002 140446 041446 037401   00130  013602 000000 041456 003300 027401 172002 032001 000467
00140  000600 041415 041453 041422 000000 141606 041415 021037   00150  170002 140461 000000 141311 041446 013317 051446 025001
00160  051452 162461 000304 140012 004306 166461 041446 026401   00170  013604 177461 170002 140546 000717 032000 041464 021040
00200  020121 051465 011504 041465 051464 140100 152465 022046   00210  141535 021401 177465 021040 020122 011530 173461 170003
00220  010201 140002 037076 021002 020043 041447 013703 021001   00230  140002 021002 004300 177461 170002 140537 000600 051447
00240  170002 140501 041461 051464 140446 152465 022054 141204   00250  152465 022073 141505 041465 003300 051464 140027 041465
00260  004500 020234 051464 152464 022054 141217 152464 022073   00270  141214 152464 022000 141211 152464 022046 141206 152464
00300  022040 141203 003345 140422 004000 041465 003400 051466   00310  041464 101465 162466 032000 035017 025001 035003 021007
00320  035011 000700 035004 000700 035003 000700 035024 171700   00330  022401 051453 035052 171700 010201 051457 035022 171700
00340  010201 051460 035021 041453 010201 051461 000000 000645   00350  051401 051462 173457 170003 010201 140005 047525 052120
00360  052524 026440 021007 020043 173461 170003 010201 140004   00370  046111 051524 026440 021005 020043 000600 041461 040013
00400  021001 035013 040011 000000 051414 141210 000600 021001   00410  170004 140721 000517 016000 000000 041414 035002 171467
00420  171421 035017 034022 000000 041421 022000 141305 041421   00430  010001 002400 051421 041467 037417 022000 141511 000600
00440  051414 021002 170004 140764 000011 100000 000032 173461   00450  170003 010201 140004 052105 054124 026440 021005 020043
00460  000600 041461 021057 035014 040010 000062 051415 141210   00470  000600 021003 170004 142002 014000 177373 000030 041415
00500  035001 171470 171467 171422 035017 034020 000063 041422   00510  022000 141305 041422 010001 002400 051422 021001 041467
00520  037417 021003 003243 012604 140010 000013 100000 021004   00530  170003 142001 177325 000035 000600 041415 041414 000000
00540  051446 041446 037401 013704 041414 051416 140036 173461   00550  170003 010201 140005 050122 047515 050124 026440 021007
00560  020043 000600 041461 040053 021001 035013 040051 000102   00570  051416 141513 000600 041415 041416 000036 013606 041416
00600  000700 000000 000600 051416 041422 026120 141613 000600   00610  021002 101422 010101 171700 004500 051453 010201 051461
00620  035000 171700 003300 051455 171700 051456 041461 051464   00630  021001 051447 040006 053453 170006 142004 002414 016000
00640  023000 177335 141505 041452 017602 140417 001552 041466   00650  170003 010201 140004 002511 047120 052524 021006 020243
00660  145205 041466 170004 010201 140006 000106 003117 052524   00670  050125 052040 021007 020243 145205 041466 170004 010201
00700  140004 000223 001513 042531 021004 020243 145205 041466   00710  170004 010201 140005 000405 002522 042523 042524 021006
00720  020243 145205 041466 170004 010201 140006 000774 003126   00730  042522 044506 054440 021007 020243 145205 041466 170004
00740  010201 140004 000776 001505 047104 021004 020243 145207   00750  041000 013716 041466 170004 010201 140005 001442 002504
00760  042502 052507 021006 020243 141503 000000 140540 021005   00770  170003 142001 177131 000600 051426 041462 051463 170003
01000  142001 177175 141207 152465 022054 141204 152465 022073   01010  141505 021011 170003 142001 177107 152466 022043 141305
```

```
01020  021007 170003 142001 177100 120426 041463 152466 002134   01030  051463 024440 041463 010101 001700 141313 000600 041463
01040  010101 000000 004000 141205 021015 170003 142001 177054   01050  041463 152466 003343 177463 041462 177463 002100 020062
01060  041465 152466 020062 170003 010201 140002 026040 021001   01070  020043 170003 142001 177103 152465 022054 145207 152465
01100  022000 141206 021010 170004 142002 177672 177015 025401   01110  171000 010201 003700 177462 041463 101462 020062 003300
01120  051462 041426 051401 140675 170003 142001 177050 141505   01130  021011 170003 142001 176770 152465 022054 141504 000600
01140  051450 140034 021004 051450 152466 022043 141305 021007   01150  170003 142001 176751 173460 041465 152466 003345 051472
01160  020063 170003 142001 177013 145207 152465 022054 141206   01170  021010 170004 142002 000062 176727 170003 142001 176777
01200  141505 021011 170003 142001 176717 041466 170003 010201   01210  140003 001513 042531 021004 020243 141505 041450 036401
01220  051450 140030 041450 036402 051450 000700 041465 152466   01230  000000 161474 141210 141303 021013 140002 021012 170003
01240  142001 176662 151474 000710 141405 021013 170003 142001   01250  176653 170003 142001 176723 145511 041450 037401 051451
01260  041450 026741 013704 151474 140003 177700 000700 161436   01270  021007 051424 041450 026721 013710 173457 173460 041472
01300  004500 051424 020063 140013 173457 170003 010201 140005   01310  047525 052120 052524 026440 021007 020043 142001 177307
01320  170003 142001 176654 141505 021022 170003 142001 176574   01330  000600 041465 152466 000000 141210 141303 021020 140002
01340  021017 170003 142001 176560 004500 022000 141405 021020   01350  170003 142001 176551 170003 142001 176621 152465 022054
01360  141205 021010 170003 142001 176537 170003 142001 176607   01370  141505 021011 170003 142001 176527 152465 004000 141503
01400  000600 140044 000600 041465 152466 000052 141210 141303   01410  021021 140002 021022 170003 142001 176506 004500 022000
01420  141405 021021 170003 142001 176477 170003 142001 176547   01430  152465 022054 141203 000600 140247 170003 142001 176537
01440  141505 021011 170003 142001 176457 041466 170003 010201   01450  140003 001511 047124 021004 020243 141507 004500 022000
01460  141502 022402 021001 140156 041466 170003 010201 140005   01470  003104 047525 041114 042440 021007 020243 141507 004500
01500  022000 141502 022404 021001 140135 041466 170003 010201   01510  140004 002122 042501 046040 021005 020243 141507 004500
01520  022000 141502 022404 021002 140115 041466 170003 010201   01530  140004 002114 047516 043440 021005 020243 141507 004500
01540  022000 141502 022406 021002 140075 004500 022000 141505   01550  021023 170003 142001 176350 041466 170003 010201 140004
01560  002102 054524 042440 021005 020243 141503 000600 140052   01570  041466 170003 010201 140005 003120 040503 045505 042040
01600  021007 020243 141503 021003 140035 041466 170003 010201   01610  140005 003504 044523 050114 040531 021010 020243 141503
01620  021004 140020 041466 170003 010201 140005 003520 040503   01630  045505 042052 021010 020243 141503 021005 140003 000600
01640  140017 170003 142001 176333 152465 022054 141535 170003   01650  142001 176325 141505 021011 170003 142001 176245 041466
01660  170003 010201 140004 002104 042523 041440 021005 020243   01670  141205 021024 170003 142001 176227 040004 002000 170004
01700  142002 000400 176274 152465 022073 145207 152465 022000   01710  141206 021014 170004 142002 177404 176206 171700 051456
01720  142001 176705 170003 142001 176252 141205 021014 170003   01730  142001 176172 041455 003445 051456 027401 142001 176667
01740  170003 142001 176234 141205 021014 170003 142001 176154   01750  041461 170003 142001 176052 173461 170003 010201 140010
01760  044516 050125 052040 043111 046105 051440 036440 021016   01770  020042 021072 051472 041462 051464 021001 051473 041473
02000  061401 145417 041464 021054 020161 101464 051471 041473   02010  061401 141602 120471 041471 061472 141324 170004 142002
02020  000032 176004 173461 170003 010201 140010 020040 020040   02030  020040 020040 020040 020040 020040 021016 020042 021072
02040  051472 041464 041471 020061 051464 041472 101471 051472   02050  120473 140452 170003 142001 175751 173461 170003 010201
02060  140010 047525 052120 052524 020106 044514 042440 036440   02070  021016 020042 173457 041424 003400 020062 041451 013710
02100  170003 010201 140003 026113 042531 021004 020042 170003   02110  142001 175714 151436 000710 145305 173461 170004 010201
02120  140014 000046 047125 046502 042522 020117 043040 051105   02130  041517 051104 051440 036440 021024 020043 171436 025001
02140  021064 000600 025001 000600 021424 177461 171472 000000   02150  177461 021040 020121 021507 177461 101701 051472 021424
02160  177461 003200 041472 020062 170003 142001 175637 041456   02170  061455 141623 173461 170003 010201 140011 047117 020113
02200  042531 051440 042516 052105 051105 042040 021017 020042   02210  170003 142001 175613 140177 173461 170003 010201 140026
02220  045505 054440 050117 051511 052111 047516 020040 020114   02230  042516 043524 044040 020040 020040 052131 050105 020040
02240  020040 040523 041457 042105 051503 021052 020042 170003   02250  142001 175554 000600 051473 171473 021003 041456 101455
02260  050002 140131 131473 177455 000600 021011 000706 041461   02270  171472 000122 041473 003343 177455 000600 021012 000706
02300  021411 177461 171472 000012 021423 177461 170003 010201   02310  140004 020040 020040 020040 021006 020042 041473 022402
02320  004300 047455 037777 023407 004300 172010 010201 003700   02330  021007 020042 170004 010201 140004 175443 020040 020040
02340  021004 020042 041473 022402 004300 047455 022377 141311   02350  170003 010201 140003 042105 051503 021004 020042 140010
02360  170003 010201 140003 020101 051503 021004 020042 041473   02370  022000 141515 170003 010201 140010 020040 020050 046501
02400  045117 051040 045505 054451 021016 020042 170003 142001   02410  175415 052527 041461 170003 142001 175410 142001 176207
02420  170003 142001 175554 141205 021014 170003 142001 175474   02430  041401 022000 141505 021030 170003 142001 175465 041456
02440  061455 141605 021006 170003 142001 175456 000600 041456   02450  101455 003443 040007 101701 067455 141606 021032 170004
02460  142002 077777 175441 077455 033003 141653 051432 171700   02470  003300 051454 041462 051464 000645 051423 051425 000700
02500  161440 000600 051473 041473 061401 145624 041464 021054   02510  020161 101464 051472 000600 041464 021003 035014 040013
02520  000000 141231 004000 173461 041464 041472 020062 170005   02530  142003 000166 014000 175272 000600 021016 170003 142001
02540  175330 121473 141104 000700 000000 140404 041446 037401   02550  017630 000000 004500 035001 171470 171467 171431 035003
02560  171433 035001 171444 171442 035002 171434 035005 034012   02570  000000 021001 041467 037417 021003 003243 012605 140013
02600  176026 000013 105440 004500 021025 170003 142001 175261   02610  120473 140450 041434 041431 002340 051435 041431 022000
02620  141305 041431 010001 002400 051431 041473 022000 141510   02630  041425 041470 026721 027321 051425 041433 051427 041470
```

```
02640  026602 022000 141507 041473 022000 141310 041431 061423    02650  141205 041425 021001 027202 051425 041431 061423 141603
02660  041431 051423 041473 022000 141504 041435 051430 140006    02670  041430 061435 141203 000600 051430 151440 151442 000710
02700  141505 034002 140004 000000 023420 151444 001100 161440    02710  041464 071472 003300 051464 120473 140612 017053 151436
02720  000710 141303 151436 161440 041451 013712 041425 000600    02730  027202 051425 041432 002400 051423 000600 051430 000600
02740  041457 041425 021001 041423 035003 041430 035001 151440    02750  035001 021040 041427 040435 000234 051417 141207 000600
02760  021026 170003 142001 175105 140623 041417 035002 171467    02770  035020 034012 000000 041467 037417 022000 141514 041417
03000  021027 170005 142003 000011 000000 175063 041417 000700    03010  000000 140650 171700 022406 022415 051001 100000 171473
03020  000000 041401 041454 041417 041451 041456 003300 101455    03030  024003 041455 035004 171401 035001 040014 000000 000600
03040  171473 000021 041417 041425 037403 022000 141504 021011    03050  140003 007702 000600 000600 000044 145220 041417 171472
03060  035003 021030 000000 041472 022144 145511 041446 037401    03070  017707 173461 170006 010201 140017 000252 000243 000210
03100  050125 051107 042440 047514 042040 047525 052120 052524    03110  020106 044514 042440 021026 020043 041417 021426 177461
03120  035022 034014 000130 021462 000500 156461 022040 141243    03130  000400 177461 170005 010201 140005 000014 000000 020077
03140  020040 021003 020042 170003 142001 174731 170004 142001    03150  174770 000136 145541 173461 021040 020121 051464 152464
03160  037537 021131 005700 145535 041417 041461 035022 034023    03170  000046 000600 041461 040021 021100 035013 040017 000000
03200  051472 141524 041472 021004 000600 000131 145210 021033    03210  170011 142006 000014 000000 002001 016000 177624 174640
03220  000011 041472 000700 000016 140005 021034 170003 142001    03230  174627 173461 170003 010201 140021 042516 052105 051040
03240  047105 053440 047101 046505 020106 047522 020117 052524    03250  050125 052040 043111 046105 035040 021040 020042 170003
03260  142001 174615 170003 142001 174654 141522 173461 021040    03270  020121 051464 152464 004000 141505 041417 041464 000000
03300  145204 021035 170004 142002 177536 174552 140455 120420    03310  041420 022144 141627 173461 170003 010201 140006 047525
03320  052120 052524 026455 026440 021011 020043 000600 041420    03330  021012 021406 177461 000000 004000 041417 041461 000040
03340  140676 041417 021031 170003 142001 174523 140051 041420    03350  022000 145105 173461 170004 010201 140026 000041 047525
03360  052120 052524 020106 044514 042440 041514 047523 042504    03370  020127 044524 044040 043111 046105 047101 046505 020117
03400  052524 050125 052040 021047 020042 051464 000600 041420    03410  021012 041464 000057 071464 170003 142001 174407 041461
03420  170003 142001 174403 173461 170003 010201 140020 020040    03430  020040 020040 020040 020040 020040 020040 020040 020040
03440  020040 051524 040524 044523 052111 041523 021036 020042    03450  170003 142001 174353 041461 170003 142001 174347 041461
03460  022462 051464 173453 170002 140015 047125 046502 042522    03470  020117 043040 044516 050125 052040 043111 046105 051440
03500  036440 021014 020003 171401 000600 021034 000600 025001    03510  000600 021427 177461 171473 000000 041464 170003 142001
03520  174305 173453 170002 140013 047125 046502 042522 020117    03530  043040 051105 041517 051104 051440 036440 021012 020003
03540  171402 025001 021040 000600 025001 000600 021423 177461    03550  171473 000035 041464 170003 142001 174250 173453 170002
03560  140015 051105 041517 051104 020123 044532 042440 024111    03570  047040 041131 052105 051451 020075 021014 020003 041423
03600  002400 051423 171423 000600 021033 000600 025001 000600    03610  021430 177461 171473 000042 041464 170003 142001 174206
03620  173453 170002 140017 051520 040503 042440 040526 040511    03630  046101 041114 042440 024111 047040 053517 051104 051451
03640  020075 021016 020003 171404 000600 021027 000600 025001    03650  000600 021434 177461 171473 000041 041464 170003 142001
03660  174145 173453 170002 140013 047125 046502 042522 020117    03670  043040 041517 046520 040522 042523 020075 021012 020003
03700  171405 025001 021037 000600 025001 000600 021424 177461    03710  171473 000035 041464 170003 142001 174110 173453 170002
03720  140013 041520 052440 052111 046505 020050 046511 047125    03730  052105 051451 020075 021012 020003 151407 003000 034020
03740  005400 161407 171407 021001 021036 021002 025001 000600    03750  021424 177461 171473 000042 041464 170005 142003 041765
03760  030000 174044 173453 170002 140015 042514 040520 051505    03770  042040 052111 046505 020050 046511 047125 052105 051451
04000  020075 021014 020003 151411 003000 034020 005400 161411    04010  171411 021001 021032 021002 025001 000600 021430 177461
04020  171473 000046 041464 170005 142003 041765 030000 173776    04030  031400
```

```
01482000  00000 1      GO TO START;
01483000  00001 1   BUGGER;
01484000  00001 1      DEBUGGING:=TRUE;
01485000  00003 1      DEBUG;
01486000  00004 1   START;
01487000  00004 1      MERGEMAIN;
01488000  00005 1   END.

 00000  140004 025001 051000 000000 000000 000000
```

IDENTIFIER	CLASS	TYPE	ADDRESS	
ASCII	PROCEDURE	INTEGER		
BINARY	PROCEDURE	LOGICAL		
BUGGER	ENTRY		PB+001	
DASCII	PROCEDURE	INTEGER		
DATELINE	PROCEDURE			
DBINARY	PROCEDURE	DOUBLE		
DEBUG	PROCEDURE			
DEBUGGING	SIMP. VAR.	LOGICAL	DB+000	
DLSIZE	PROCEDURE	INTEGER		
FCHECK	PROCEDURE			
FCLOSE	PROCEDURE			
FGETINFO	PROCEDURE			
FOPEN	PROCEDURE	INTEGER		
FREAD	PROCEDURE	INTEGER		
FRELATE	PROCEDURE	LOGICAL		
FRENAME	PROCEDURE			
FWD	DEFINE			OPTION FORWARD
FWRITE	PROCEDURE			
GETERRORMESS	PROCEDURE	INTEGER		
INEXT'	PROCEDURE			
MERGE	PROCEDURE			
MERGEERRORMESS	PROCEDURE			
MERGEEXITPARAM	EQUATE			VALUE = %15
MERGEMAIN	PROCEDURE			
MERGETITLE	PROCEDURE			
NUMRECSPTR	POINTER	DOUBLE	DB+001	
NUMSTATS	EQUATE			VALUE = %12
PRINT	PROCEDURE			
PRINT'FILE'INFO	PROCEDURE			
PROCTIME	PROCEDURE	DOUBLE		
RESETCONTROL	PROCEDURE			
S0	SIMP. VAR.	INTEGER	S -000	
S1	SIMP. VAR.	INTEGER	S -001	
START	LABEL		PB+004	
TERMINATE	PROCEDURE			
TERMINATE'	PROCEDURE			
TIMER	PROCEDURE	DOUBLE		
TRAP	PROCEDURE			
VERSION	DEFINE			"HP32214B.00.00 MERGE/3000 "
X	SIMP. VAR.	INTEGER	XREG	
XCONTRAP	PROCEDURE			
ZSIZE	PROCEDURE	INTEGER		

```
PRIMARY DB STORAGE=%0021    SECONDARY DB STORAGE=%00000
NO. ERRORS=0001             NO. WARNINGS=000
PROCESSOR TIME=0:01:131     ELAPSED TIME=0:04:19
```

```
 END OF COMPILE
!EOJ

CPU (SEC) = 191
ELAPSED (MIN) = 19
THU, JUN  3, 1976, 11:53 PM
END OF JOB
```