NOTICE

THIS DOCUMENT HAS BEEN REPRODUCED FROM MICROFICHE. ALTHOUGH IT IS RECOGNIZED THAT CERTAIN PORTIONS ARE ILLEGIBLE, IT IS BEING RELEASED IN THE INTEREST OF MAKING AVAILABLE AS MUCH INFORMATION AS POSSIBLE

"Made available under NASA sponsorship in the interest of early and wide dissemination of Earth Resources Survey Program information and without liability for any use made thereof."

T77-10766 NMF
80-10218
JSC-12537
NASA CR-
160641

AS BUILT DESIGN SPECIFICATION FOR THE YIELD ESTIMATION SUBSYSTEM (YES) MONTHLY YIELD DATA BASE AND SUPPORTING PROGRAMS

Job Order 74-963

AD 63-1347-4963-01

(E80-10218) AS-BUILT DESIGN SPECIFICATION FOR THE YIELD ESTIMATION SUBSYSTEM (YES) MONTHLY YIELD DATA BASE AND SUPPORTING PROGRAMS (Lockheed Electronics Co.) 124 p HC A06/MF A01 CSCL 05B G3/43

N80-29795

Unclas 00218

Prepared By

Lockheed Electronics Company, Inc.
Systems and Services Division
Houston, Texas

Contract NAS 9-15200

For

EARTH OBSERVATIONS DIVISION

National Aeronautics and Space Administration
LYNDON B. JOHNSON SPACE CENTER
Houston, Texas

February 1977

LEC-10034

JSC-12537

AS BUILT DESIGN SPECIFICATION FOR THE YIELD ESTIMATION SUBSYSTEM (YES) MONTHLY YIELD DATA BASE AND SUPPORTING PROGRAMS

Job Order 74-963

AD 63-1347-4963-01

PREPARED BY

D. Cook
C. Slemons

APPROVED BY

Philip L Krumm

P. L. Krumm, Supervisor
Software Development Section

Prepared By

Lockheed Electronics Company, Inc.

For

Earth Observations Division

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
LYNDON B. JOHNSON SPACE CENTER
HOUSTON, TEXAS

February 1977

LEC-10034

CONTENTS

Section Page

Appendix

1. SCOPE

This document describes the monthly weather and yield data base and associated computer programs installed on the 360/195 complex at Suitland, Maryland. The system is in support of Yield Estimation efforts of LACIE.

2. APPLICABLE DOCUMENTS

AD 63-1347-4963-01

AD 63-1347-4963-04

AD-04 requires specification for the India data base. This is not available at the time of preparation of this document. Documentation for India will be delivered separately.

3. SYSTEM DESCRIPTION

The monthly yield data base system consists of three components. The first is the computer hardware necessary to support the system. This is described in section 3.1. The second is a data base structure. This is described in section 3.2. The third is a set of support programs. This is described in section 3.3.

3.1 HARDWARE DESCRIPTION

These programs and data are resident on the IBM 360/195 complex at Suitland, Maryland. They should be transferable to any IBM 360-370 series machine with sufficient disk to handle the data base and main memory to support the PL/I optimizing compiler.

3.2 DATA BASE STRUCTURE

The data base (Monthly Yield Data Base) is a tree structure, nodes being countries, regions, districts, etc. Nodes are referred to as levels in the remainder of this document. The basic unit of information is a block. Blocks are of four types: Control, Directory, Data Descriptor and Data, and Model Definition, each with a corresponding PL/I structure given in appendix A.

3.2.1 DATA BASE STORAGE REQUIREMENTS

The data base currently occupies 288 6440 byte blocks, partitioned into three data sets: USA 114 blocks, USSR/Canada 114 blocks, Argentina/Australia 60 blocks.

3.2.2 CONTROL BLOCKS

There is only one control block on a file. It is the first block to be defined and contains information on the block type of every other block in the file. It also contains the location of the directory entry for every level-one region (usually a country).

Control block information is divided into eleven sections, some of which are arrays with subsections.

1. The first section is the file identification name which is a name up to eight characters in length describing the file.

2. The second section gives the number of passwords which are available to use the programs accessing the file.

3. The third section is an array of one to eight passwords, each up to eight characters in length. Any one of the passwords can be used to access the programs. The number of passwords in this section should equal the number given in section 2.

4. The fourth section gives the number of levels in which the data is arranged.

5. The fifth section is an array of one to eight level names, each up to 24 characters in length. The levels refer to the organization of the data. The smaller the level number, the larger the region; the larger the level number, the smaller the region. For example, level one is probably a country, whereas level four may be a crop reporting district. Data are collected at the smaller regions (higher level numbers) and may or may not be aggregated up to lower level numbers. The number of level names in this section should equal the number given in section 4.

6. The sixth section gives the number of codes, not to exceed 32, for variables which are in the data blocks.

7. The seventh section is an array with subsections giving information on each of the codes. The six subsections are repeated for each of the codes, the number of which should equal the number in section 6.

   a. The code number identifies the variable, for example, precipitation.

   b. The unit number identifies how the variable is measured, for example, millimeters.

   c. The base is the number of digits allowed for an observation.

   d. The scale is the power of ten by which the observation is multiplied. This may be simply the number of decimal places in the observation; it eliminates keypunching the decimal points.

   e. The code name is a name up to 24 characters in length associated with the code number.

   f. The unit name is a name up to 24 characters in length associated with the unit number.

8. The eighth section gives the number of level-one regions on the file. This will probably be the number of countries, and cannot exceed 24.

9. The ninth section is an array with subsections giving information on each of the level-one regions. The five subsections are repeated for each level-one region, the number of which should equal the number in section 8.

   a. The code number identifies the level-one region.

   b. The number of directories is the current count of directory entries on the file for that level-one region and all higher level regions within that level-one region.

c. The record number is the location on the file of the directory block containing the directory entry for the level-one region.

d. The displacement number is the position in the directory block where the directory entry for the level-one region begins.

e. The level-one region name is the name up to 24 characters in length for that region.

10. The tenth section gives the number, not exceeding 601, of records or blocks which the file can contain, excluding the control block. Each block is 6440 bytes long.

11. The eleventh section is an array with subsections giving information on each of the records on the file. The three subsections are repeated for each record, the number of which should be equal to the number in section 10.

a. The record type identifies each record according to what kind of block it contains.

1) A type of 0 (zero) means blank or no information recorded on the record.

2) A type of -1 (negative one) means the record contains directory entries.

3) A type of +1 (positive one) means the record contains a data descriptor and data.

4) A type of +2 (positive two) means the record contains a model definition block.

b. The free space is the number of bytes on the record which are blank.

c. The location is the position on the record where the free space begins.

3.2.3 DIRECTORY BLOCKS

There is a directory entry for every level, sublevel, subsublevel, etc., to a maximum of eight levels. The entries contain information which gives the location of other entries at the same level and at the next higher and next lower levels, and also information which gives the location in the file of the entry's data descriptor and model definition. Directory entries are grouped together in directory blocks, with the number of blocks dependent on the number of reporting districts.

A directory entry is divided into fifteen sections, one of which is an array with three subsections. A directory block contains up to 84 directory entries, each 76 bytes long, for a level-one region. More than one directory block may be needed for a level-one region, but a directory block does not contain directory entries for more than one level-one region.

1. The first section is the level number for the entry. It ranges from one to the maximum number of levels defined in the fourth section of the control block.

2. The second section is the code number for the entry. It is a unique number only within that particular sublevel. For example, there could be a code number of 10 for more than one level-three entry provided each of them is associated with a different level-two region.

3. The third section is the latitude for the region. It is a positive number for regions in the northern hemisphere and negative for those in the southern hemisphere. For large areas it is the latitude of some central point.

4. The fourth section is the longitude for the region. It is a positive number in the western hemisphere and negative in the eastern hemisphere. For large areas it is the longitude of some central point.

5. The fifth section is the name of the region to which the directory entry pertains.

6. The sixth section is the location on the file of the directory block which contains the directory entry for the "parent" of the current entry. The "parent" of any entry is the entry with the next smallest level number and of which the original entry is a part. For example, the Black Lands is a level-four region whose parent is the level-three region, Texas. The parent of Texas is the level-two region, the Great Plains, whose parent is the level-one region, the United States. Level-one regions have no parent, so the location is coded as -1 (negative one).

7. The seventh section is the position within the directory block where the parent's directory entry begins. The directory block location is given in section 6; if the directory block location is a -1 (negative one), this position is set to a +1 (positive one).

8. The eighth section is the location on the file of the directory block which contains the directory entry for the "brother" of the current entry. The "brother" of any entry is the entry with the same level number, the same parent, and the next largest code number. For example, the brother of the Black Lands with code number 40 is East Texas North with code number 51. Both are at level four and have Texas as their parent. The brother of East Texas North is East Texas South which has the code number 52. The last entry under a given parent has no brother, so the location is coded as a -1 (negative one).

9. The ninth section is the position within the directory block where the brother's directory entry begins. The directory block location is given in section 8; if the directory block location is -1 (negative one), this position is set to a +1 (positive one).

10. The tenth section is the location on the file of the directory block which contains the directory entry for the "child" of the current entry. The child of any entry is the entry with the next largest level number and the smallest code number of all entries which are a part of the current entry. For example, the North High Plains, which is at level four and has a code number of 11, is the child of Texas. The entries with the highest level numbers have no children, so the location is coded as a -1 (negative one).

11. The eleventh section is the position within the directory block where the child's directory entry begins. The directory block location is given in section 10; if the directory block location is a -1 (negative one), then this position is set to a +1 (positive one).

12. The twelfth section is the location on the file of the block which contains the data descriptor entry, followed immediately by the data associated with the directory entry. If there are no data for the entry, this location is coded as a -1 (negative one).

13. The thirteenth section is the position within the data descriptor and data block where the data descriptor entry begins. The data block location is given in section 12; if the data block location is a -1 (negative one), then this position is coded as a +1 (positive one).

14. The fourteenth section is a ten-digit code number which is unique for every directory entry. It is made up of the code numbers for all lower level regions of which the particular region is a part, and the region's own code number. The first two digits contain the level-one region code, the second two contain the level-two region code, etc. When the region's own code is reached, the remaining digits are coded as zeros. For example, the United States would be

coded as 0300000000, the Great Plains as 0301000000, Texas as 0301480000, and the Black Lands as 0301484000.

15. The fifteenth section is an array with subsections giving information on the model definition blocks for up to four different crops.

    a. The crop code identifies the crop whose yield the model is estimating.

    b. The model record number gives the location on the file of the model definition block for the particular crop and region.

    c. The model displacement number gives the position in the block where the model definition begins.

3.2.4 DATA DESCRIPTOR AND DATA BLOCKS

There is a data descriptor entry preceding the data for every region for which data is available. It contains information about the region and completely describes the amount, type, and format of the data that follows. The data include historic weather and yield measurements for a particular region.

A data descriptor entry, which is 336 bytes long, is divided into fifteen sections, one of which is an array with five subsections. The data descriptor entry immediately precedes the data for all years from a certain region. In many cases, there will be only one region's descriptor and data on a 6440-byte record. However, if there are a limited number of variables recorded and/or a limited number of years available, a second region's descriptor and data may be started in the middle of the record at byte 3221.

1. The first section is the identification number. It is the same ten-digit code number which is given in section fourteen of the region's directory entry and has been previously described in part 3.2.3.

2. The second section is the World Meteorological Organization's code number for the region. If the region has no WMO number, this section is coded as zero.

3. The third section is the latitude of the region and is identical to the third section of the region's directory entry.

4. The fourth section is the longitude of the region and is identical to the fourth section of the region's directory entry.

5. The fifth section is the elevation of the region. If unknown, this is coded as zero.

6. The sixth section is the total number of years of data from the particular region which a record (or half a record) could contain. It will depend on the amount of data recorded for each year, which will vary according to country.

7. The seventh section is the current count of the number of years of data from the particular region that the record contains.

8. The eighth section is the length in bytes needed to store one year's data. This should be the same for regions within a country, but will vary between countries.

9. The ninth section is the location on the file of the data block that contains the first chronological year's data for the region. In most cases this should be the same record location as the data descriptor entry's location. Also in most cases, the first chronological year and the first physical year in the data block are the same.

10. The tenth section is the position in the data block where the first chronological year's data begin.

11. The eleventh section is the location on the file of the data block that contains the last chronological year's data for the region. In most cases the last chronological year and the last physical year in the data block are the same.

12. The twelfth section is the position in the data block where the last chronological year's data begin.

13. The thirteenth section is reserved space, eighteen bytes long. It is coded as blank and can be used later if needed.

14. The fourteenth section is the number of codes, not to exceed twelve, for variables used in the data which follow.

15. The fifteenth section is an array with subsections giving information on each of the codes. The five subsections are repeated for each of the codes, the number of which should equal the number in section 14. The codes in the data descriptor entry should be a subset of the codes in the control block.

    a. The code number identifies the variable. The code number table is given in appendix B.

    b. The number of elements is the number of times the variable is recorded in a year. For example, if precipitation is recorded on a monthly basis, the number of elements is twelve.

    c. The element size is the length in bytes of a single observation of the variable. For example, the precipitation for a given month uses two bytes of storage.

    d. The number of subcodes is the number of subdivisions into which the variable is broken down. For example, the variable production can be broken down into production for spring wheat and for winter wheat.

    e. An array of one to eight code numbers identifies the subdivisions of the variable. The number of codes in this array should equal the number given in part d above.

The data which are stored in the data descriptor and data block will vary from country to country. However, for all countries the data for a region are grouped according to year and begin immediately after the region's data descriptor entry. Also for all countries, the first eight bytes of each year's data will contain the same variables.

1. The first variable is the year in which the data were recorded.

2. The second variable is the location on the file of the data block that contains the next chronological year's data for the region.

3. The third variable is the position in the data block where the next chronological year's data begin.

4. The fourth variable is two bytes of reserved space which is coded as blank and can be used later if needed.

3.2.5 MODEL DEFINITION BLOCKS

There is a separate model definition block for every district requiring a unique yield model. It contains the information needed to run the appropriate model for that district.

3.3 SUPPORTING PROGRAMS

There are three classes of programs supporting the data base:

1. Initialization and Definition Programs
   (INITIAL and YESM001, 3.3.1 to 3.3.10) These prepare the data base and subsections of the data base for data entry.

2. Data Entry Programs
   (Loaders and UPDDATA, 3.3.11 to 3.3.12) These programs load data into the data base.

3. Listing Programs
(LISTJOB, YESLS02, YESLS04, 3.3.13 to 3.3.15) These programs list data stored in the data base.

3.3.1 DATA BASE INITIALIZATION PROGRAM (INITIAL)

INITIAL prepares the data base for entry of directory and data by setting size information parameters and filling the data area with zeros.

3.3.1.1 Linkages

None.

3.3.1.2 Interfaces

INITIAL must be run first.

3.3.1.3 Inputs

The file name via JCL. The file size encoded at line 430. (See listing.)

3.3.1.4 Outputs

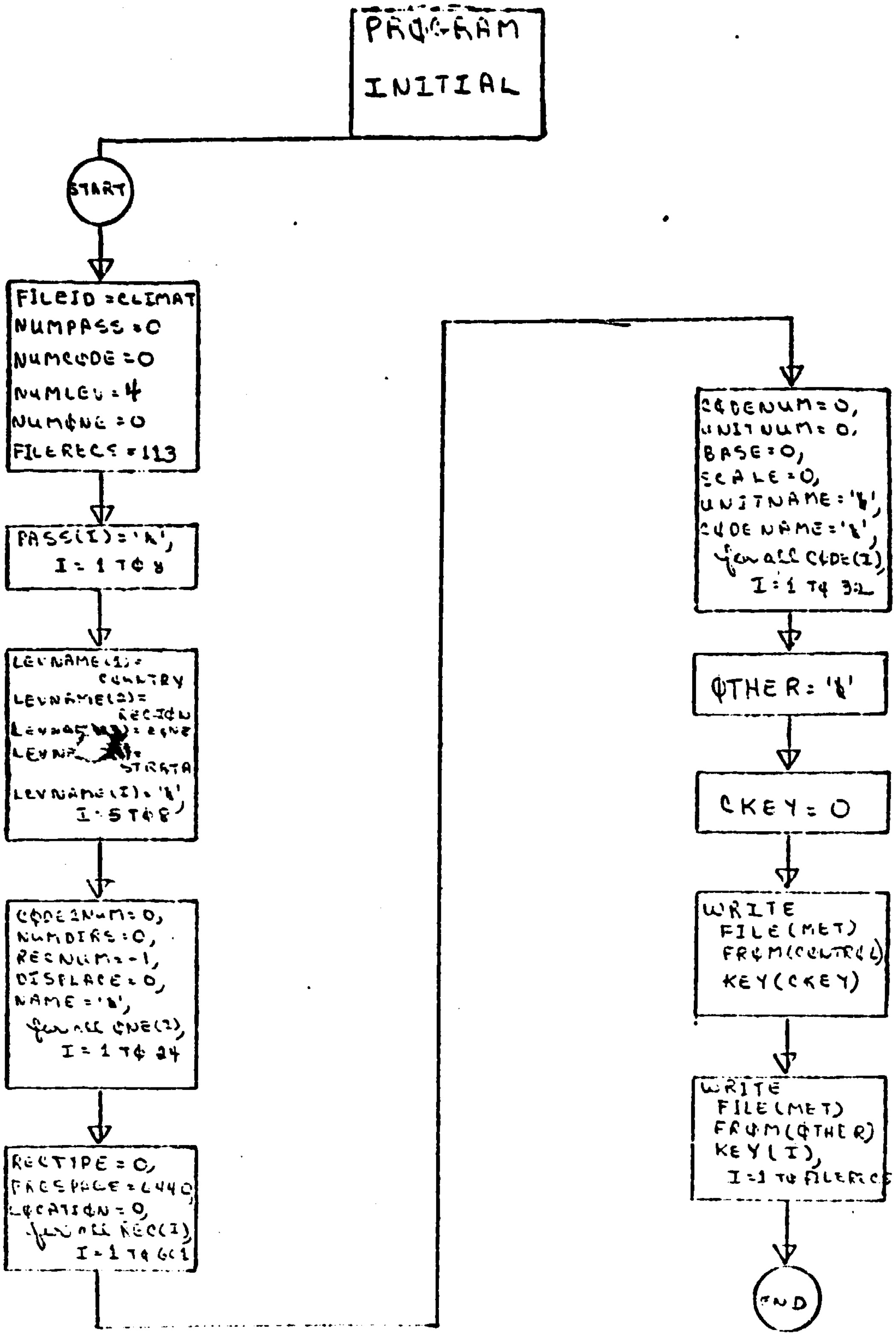
Data base prepared for subsequent processing.

3.3.1.5 Flow Chart

Next page.

3.3.1.6 Listing

Follows flow chart.


PRØGRAM INITIAL
START
FILEID = CLIMAT
NUMPASS = 0
NUMCØDE = 0
NUMLEV = 4
NUMØNE = 0
FILERECS = 113
PASS(I) = 'b̸', I = 1 TØ 8
LEVNAME(1) = CØUNTRY
LEVNAME(2) = REGIØN
LEVNAME(3) = ZØNE
LEVNAME(4) = STRATA
LEVNAME(I) = 'b̸', I = 5 TØ 8
CØDEINUM = 0, NUMDIRS = 0, RECNUM = -1, DISPLACE = 0, NAME = 'b̸', for all ØNE(I), I = 1 TØ 24
RECTYPE = 0, FREESPACE = 6440, LØCATIØN = 0, for all REC(I), I = 1 TØ 661
CØDENUM = 0, UNITNUM = 0, BASE = 0, SCALE = 0, UNITNAME = 'b̸', CØDENAME = 'b̸', for all CØDE(I), I = 1 TØ 32
ØTHER = 'b̸'
CKEY = 0
WRITE FILE(MET) FRØM(CØNTRØL) KEY(CKEY)
WRITE FILE(MET) FRØM(ØTHER) KEY(I), I = 1 TØ FILERECS
END

ORIGINAL PAGE IS
OF POOR QUALITY

ORIGINAL PAGE IS
OF POOR QUALITY

ORIGINAL PAGE IS
OF POOR QUALITY

```
RUN NO. 15       DATE  11/12/75    TIME  0910             LISTING OF MODULE INITAL

DESCRIPTION            DATA BASE PGM

MASTER FILE            W.EDS.CCEA.LEC.LTBR
ADDED TO MASTER        10/13/75
LAST DATE COPIED         NONE
LAST UPDATE              NONE

PASSWORD               JVGV
PROGRAMMER             LEC
LANGUAGE               PLI
PROC PARAMETER         $NOJCL

 INITAL:  PROC OPTIONS(MAIN);
   DCL 1 CONTROL,
            2 FILEID CHAR(8),
            2 NUMPASS FIXED BIN(15,0),
            2 PASS(8) CHAR(8),
            2 NUMLEV FIXED BIN(15,0),
            2 LEVNAME(4) CHAR(24),
            2 NUMCODE FIXED BIN(15,0),
            2 CODE(12),
              3 CODENUM FIXED BIN(15,0),
              3 UNITNUM FIXED BIN(15,0),
              3 BASE FIXED BIN(15,0),
              3 SCALE FIXED BIN(15,0),
              3 CODENAME CHAR(24),
              3 UNITNAME CHAR(24),
            2 NUMONE FIXED BIN(15,0),
            2 ONE(24),
              3 CODEINUM FIXED BIN(15,0),
              3 NUMDISS FIXED BIN(15,0),
              3 RECNUM FIXED BIN(15,0),
              3 DISPLACE FIXED BIN(15,0),
              3 NAME CHAR(24),
            2 FILERECS FIXED BIN(15,0),
            2 REC(501),
              3 RECTYPE FIXED BIN(15,0),
              3 FREESPACE FIXED BIN(15,0),
              3 LOCATION FIXED BIN(15,0);
     DCL OTHER CHAR(544);
     DCL I FIXED BIN(15,0);
     DCL (CKEY,DKEY) FIXED BIN(10,0);
    DCL MET FILE RECORD DIRECT KEYED ENV(REGIONAL(1)) OUTPUT;
     DCL PI POINTER;
     DCL D CHAR(8) BASED(PI);
     DCL PW CHAR(8);
     DCL B BIT(64) BASED(PI);
      OPEN FILE(MET) ;
      ALLOCATE D SET(PI);
       CONTROL.FILEID='CLIMAT';
      CONTROL.NUMPASS=0;
      CONTROL.NUMCODE=9;
      CONTROL.NUMLEV=4;
      CONTROL.NUMONE=0;
      CONTROL.FILERECS=75;      /*   CANADA & RUSSIA COMBINED  */
```

```
RUN NO. 15     DATE 11/12/76     TIME 0910          LISTING OF MODULE INITAL
       DO I = 1 TO 8;
          CONTROL.PASS(I)=' ';
       END;
       DO I = 1 TO 8;
          CONTROL.LEVNAME(I)=' ';
       END;
       CONTROL.LEVNAME(1)='COUNTRY';
       CONTROL.LEVNAME(2)='REGION';
       CONTROL.LEVNAME(3)='ZONE';
       CONTROL.LEVNAME(4)='STRATA';
       DO I = 1 TO 24;
          CONTROL.ONE(I).CODENUM=0;
          CONTROL.ONE(I).NUMDIRS=0;
          CONTROL.ONE(I).NAME=' ';
          CONTROL.ONE(I).RECNUM=-1;
          CONTROL.ONE(I).DISPLACE=0;
       END;
       DO I = 1 TO 601;
          CONTROL.REC(I).RECTYPE=0;
          CONTROL.REC(I).FRESPACE=6440;
          CONTROL.REC(I).LOCATION=0;
       END;
       DO I = 1 TO 32;
          CONTROL.CODE(I).CODENUM=0;
          CONTROL.CODE(I).UNITNAME=' ';
          CONTROL.CODE(I).UNITNUM=0;
          CONTROL.CODE(I).BASE=0;
          CONTROL.CODE(I).CODENAME=' ';
          CONTROL.CODE(I).SCALE=0;
       END;
       OTHER=' ';
       CKEY=0;
       DO I = 1 TO CONTROL.NUMPASS;
          PUT SKIP EDIT('ENTER PASSWORD :')(A);
          GET EDIT(PW)(A(3));
          D=PW;
          P= - B;
          CONTROL.PASS(I)=D;
       END;
       WRITE FILE(MET) FROM(CONTROL) KEYFROM(CKEY);
       DO I = 1 TO CONTROL.FILERECS;
        DKEY=I;
        WRITE FILE(MET) FROM(OTHER) KEYFROM(DKEY);
      END;
       FREE D;
       CLOSE FILE(MET);
       PUT SKIP EDIT('**** END OF JOB ****')(A);
    END INITAL;
```

3.3.2 CONTROL, DIRECTORY, AND DATA DESCRIPTOR ENTRY (YESM001)

YESM001 is used to enter control directory and data descriptive information prior to data entry.

3.3.2.1 Linkages

YESM001 calls YESX002, YESPC01, YESDF01, YESDE01, YESLS01, and YESUD01. YESDE01, YESLS01 and YESUD01 are dummy programs.

3.3.2.2 Interfaces

INITIAL must be run before YESM001.

3.3.2.3 Inputs

See 4.1.2.1. 4.1.2.1

ORIGINAL PAGE IS
OF POOR QUALITY

3.3.2.4 Outputs

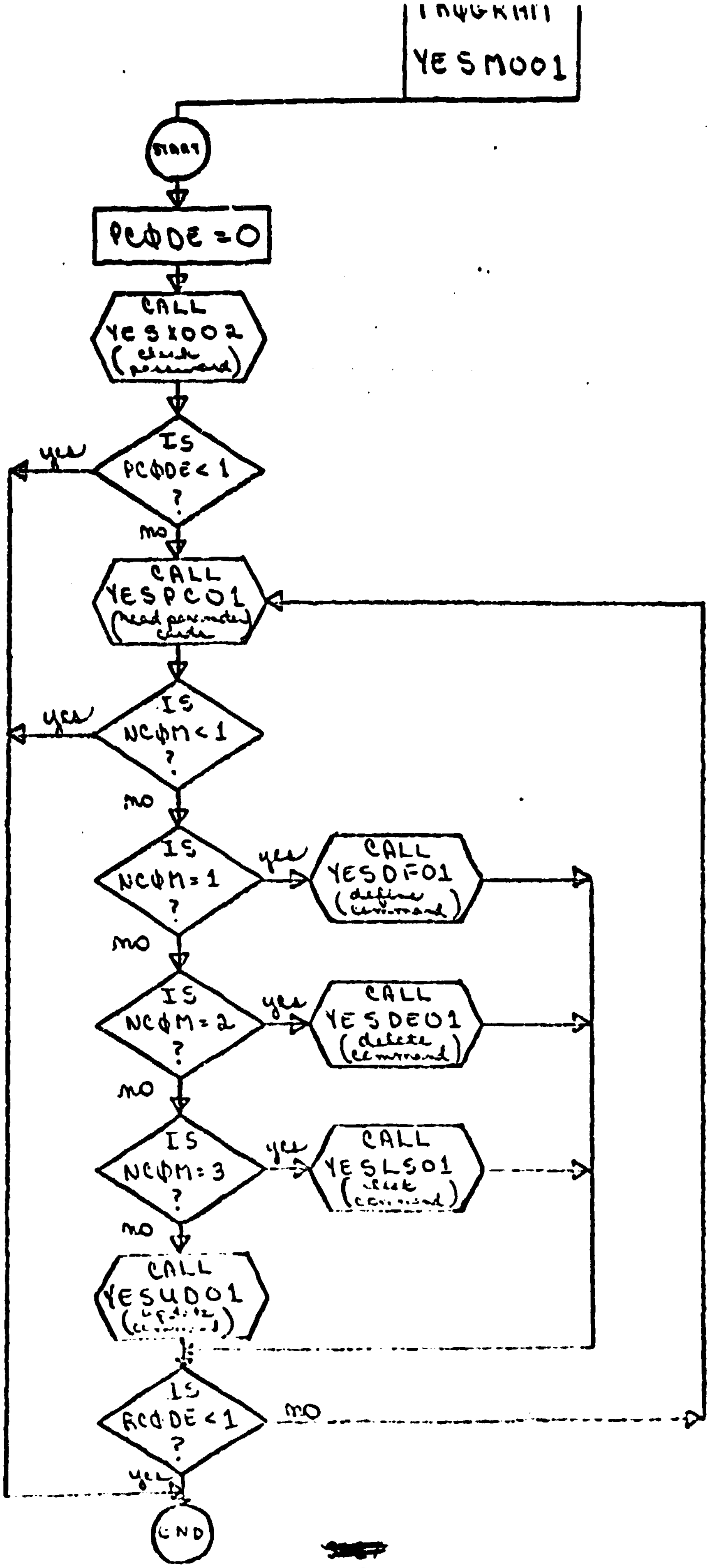
Directory descriptor and control entries in the data base.

3.3.2.5 Flow Chart

Next page.

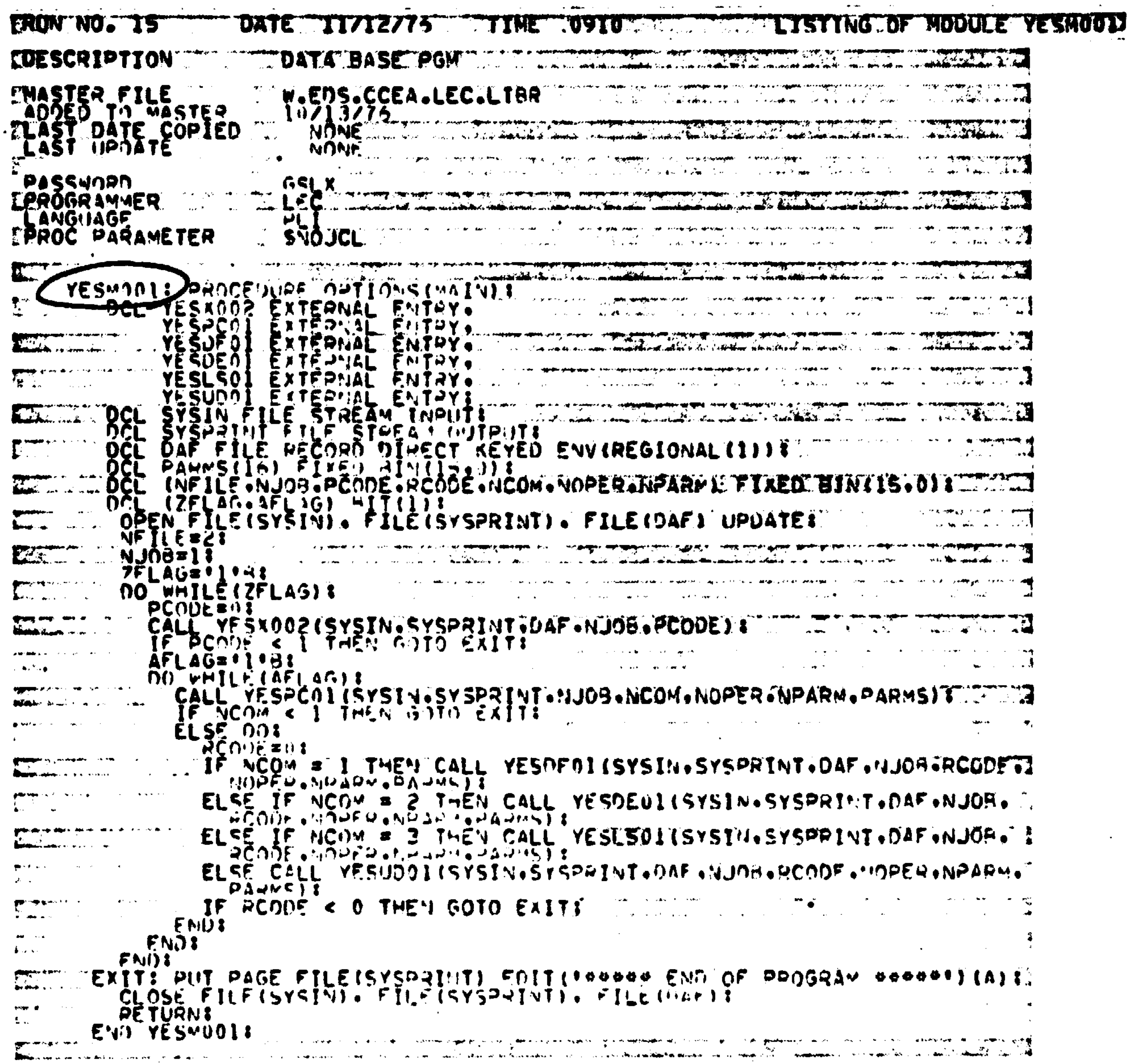
3.3.2.6 Listing

Follows flow chart.


PROGRAM
YESMOO1
START
PCØDE = O
CALL YESXOO2 (check password)
IS PCØDE < 1 ?
yes
no
CALL YESPCO1 (read parameter data)
IS NCØM < 1 ?
yes
no
IS NCØM = 1 ?
yes
CALL YESDFO1 (define command)
no
IS NCØM = 2 ?
yes
CALL YESDEO1 (delete command)
no
IS NCØM = 3 ?
yes
CALL YESLSO1 (list command)
no
CALL YESUDO1 (update command)
IS RCØDE < 1 ?
no
yes
END

[redacted]

ORIGINAL PAGE IS
OF POOR QUALITY



```
RUN NO. 15        DATE  11/12/75     TIME  0910                LISTING OF MODULE YESM001

DESCRIPTION              DATA BASE PGM

MASTER FILE              W.EDS.CCEA.LEC.LIBR
 ADDED TO MASTER         10/13/75
LAST DATE COPIED           NONE
LAST UPDATE                NONE

 PASSWORD                GSLX
PROGRAMMER               LEC
 LANGUAGE                PLI
PROC PARAMETER           SNOJCL

   YESM001: PROCEDURE OPTIONS(MAIN);
         DCL YESX002 EXTERNAL ENTRY,
             YESPC01 EXTERNAL ENTRY,
             YESDF01 EXTERNAL ENTRY,
             YESDE01 EXTERNAL ENTRY,
             YESLS01 EXTERNAL ENTRY,
             YESUD01 EXTERNAL ENTRY;
         DCL SYSIN FILE STREAM INPUT;
         DCL SYSPRINT FILE STREAM OUTPUT;
         DCL DAF FILE RECORD DIRECT KEYED ENV(REGIONAL(1));
         DCL PARMS(16) FIXED BIN(15,0);
         DCL (NFILE,NJOB,PCODE,RCODE,NCOM,NOPER,NPARM) FIXED BIN(15,0);
         DCL (ZFLAG,AFLAG) BIT(1);
          OPEN FILE(SYSIN), FILE(SYSPRINT), FILE(DAF) UPDATE;
          NFILE=2;
          NJOB=1;
          ZFLAG='1'B;
          DO WHILE(ZFLAG);
             PCODE=0;
             CALL YESX002(SYSIN,SYSPRINT,DAF,NJOB,PCODE);
             IF PCODE < 1 THEN GOTO EXIT;
             AFLAG='1'B;
             DO WHILE(AFLAG);
                CALL YESPC01(SYSIN,SYSPRINT,NJOB,NCOM,NOPER,NPARM,PARMS);
                IF NCOM < 1 THEN GOTO EXIT;
                ELSE DO;
                   RCODE=0;
                   IF NCOM = 1 THEN CALL YESDF01(SYSIN,SYSPRINT,DAF,NJOB,RCODE,
                      NOPER,NPARM,PARMS);
                   ELSE IF NCOM = 2 THEN CALL YESDE01(SYSIN,SYSPRINT,DAF,NJOB,
                      RCODE,NOPER,NPARM,PARMS);
                   ELSE IF NCOM = 3 THEN CALL YESLS01(SYSIN,SYSPRINT,DAF,NJOB,
                      RCODE,NOPER,NPARM,PARMS);
                   ELSE CALL YESUD01(SYSIN,SYSPRINT,DAF,NJOB,RCODE,NOPER,NPARM,
                      PARMS);
                   IF RCODE < 0 THEN GOTO EXIT;
                END;
             END;
          END;
       EXIT: PUT PAGE FILE(SYSPRINT) EDIT('***** END OF PROGRAM *****')(A);
          CLOSE FILE(SYSIN), FILE(SYSPRINT), FILE(DAF);
          RETURN;
       END YESM001;
```

3.3.3 PASSWORD VALIDATION SUBROUTINE (YESX002)

YESX002 is a subroutine called by YESM001 to validate the users password.

3.3.3.1 Linkages

None.

3.3.3.2 Interfaces

YESX002 searches the password section of the control block.

3.3.3.3 Inputs

Card containing password.

3.3.3.4 Outputs

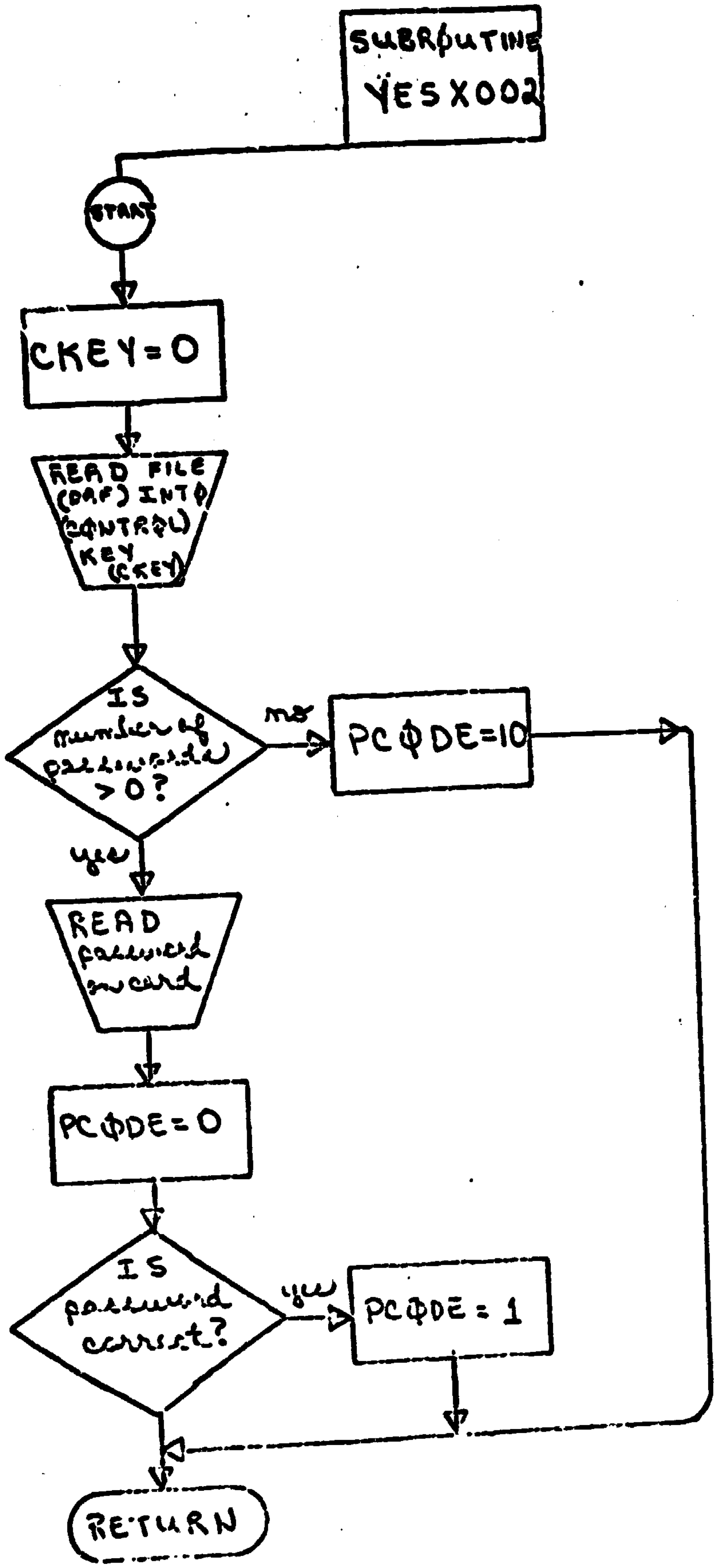
Code allowing or disallowing data base access.

3.3.3.5 Flow Chart

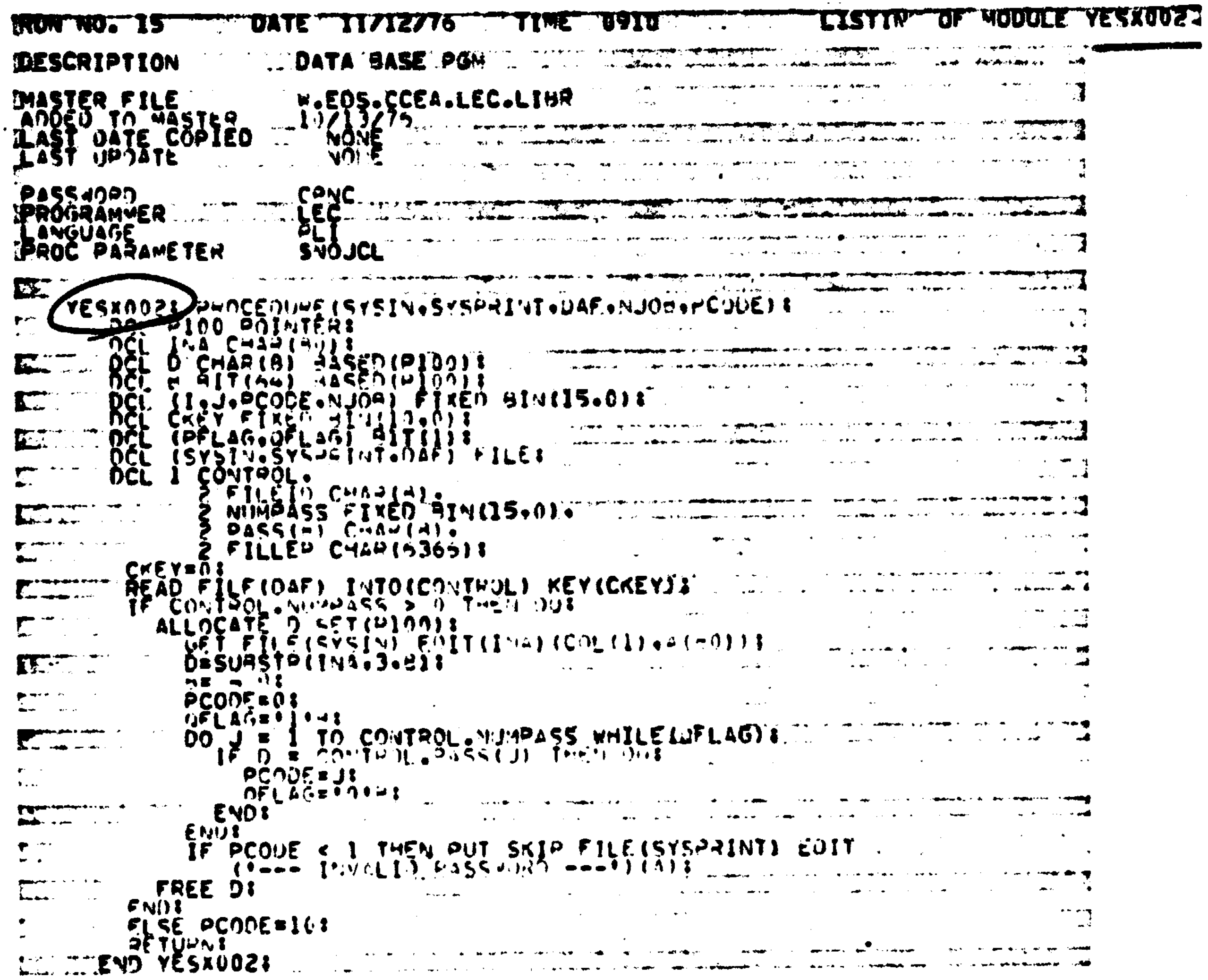
Next page.

3.3.3.6 Listing

Follows flow chart.


SUBRØUTINE YESX002
START
CKEY=0
READ FILE (DAF) INTØ (CØNTRØL) KEY (CKEY)
IS number of passwords >0?
no
PCØDE=10
yes
READ password on card
PCØDE=0
IS password correct?
yes
PCØDE = 1
RETURN

ORIGINAL PAGE IS
OF POOR QUALITY



```
RUN NO. 15        DATE  11/12/76    TIME  0910        LISTING OF MODULE YESX002

DESCRIPTION            DATA BASE PGM

MASTER FILE            W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER        10/13/75
LAST DATE COPIED         NONE
LAST UPDATE              NONE

PASSWORD               CONC
PROGRAMMER             LEC
LANGUAGE               PLI
PROC PARAMETER         SNOJCL

 YESX002: PROCEDURE(SYSIN,SYSPRINT,DAF,NJOB,PCODE);
     DCL P100 POINTER;
     DCL INA CHAR(80);
     DCL D CHAR(8) BASED(P100);
     DCL E BIT(64) BASED(P100);
     DCL (I,J,PCODE,NJOB) FIXED BIN(15,0);
     DCL CKEY FIXED BIN(31,0);
     DCL (PFLAG,OFLAG) BIT(1);
     DCL (SYSIN,SYSPRINT,DAF) FILE;
     DCL 1 CONTROL,
           2 FILEID CHAR(8),
           2 NUMPASS FIXED BIN(15,0),
           2 PASS(*) CHAR(8),
           2 FILLER CHAR(6365);
      CKEY=0;
      READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
      IF CONTROL.NUMPASS > 0 THEN DO;
        ALLOCATE D SET(P100);
          GET FILE(SYSIN) EDIT(INA)(COL(1),A(80));
          D=SUBSTR(INA,3,8);
          E = ¬E;
          PCODE=0;
          OFLAG='1'B;
          DO J = 1 TO CONTROL.NUMPASS WHILE(OFLAG);
            IF D = CONTROL.PASS(J) THEN DO;
              PCODE=J;
              OFLAG='0'B;
            END;
          END;
          IF PCODE < 1 THEN PUT SKIP FILE(SYSPRINT) EDIT
             ('--- INVALID PASSWORD ---')(A);
        FREE D;
      END;
      ELSE PCODE=10;
      RETURN;
    END YESX002;
```

3.3.4 COMMAND CARD DECODING (YESPC01)

YESPC01 is called by YESM001 to decode a command card.

3.3.4.1 Linkages

None.

3.3.4.2 Interfaces

None.

3.3.4.3 Inputs

A command card (see section 4).

3.3.4.4 Outputs

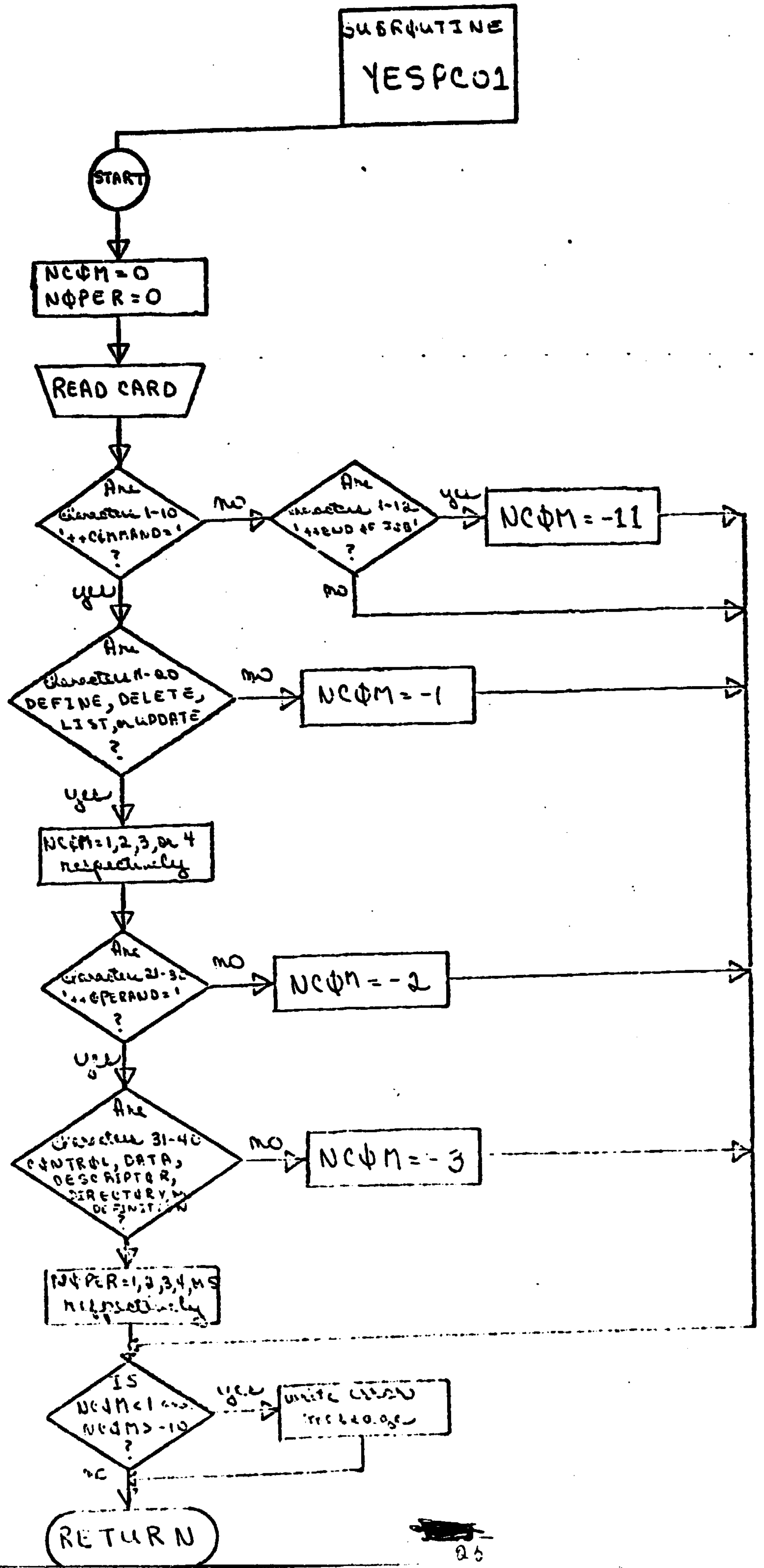
The card is parsed and results returned to YESM001.

3.3.4.5 Flow Chart

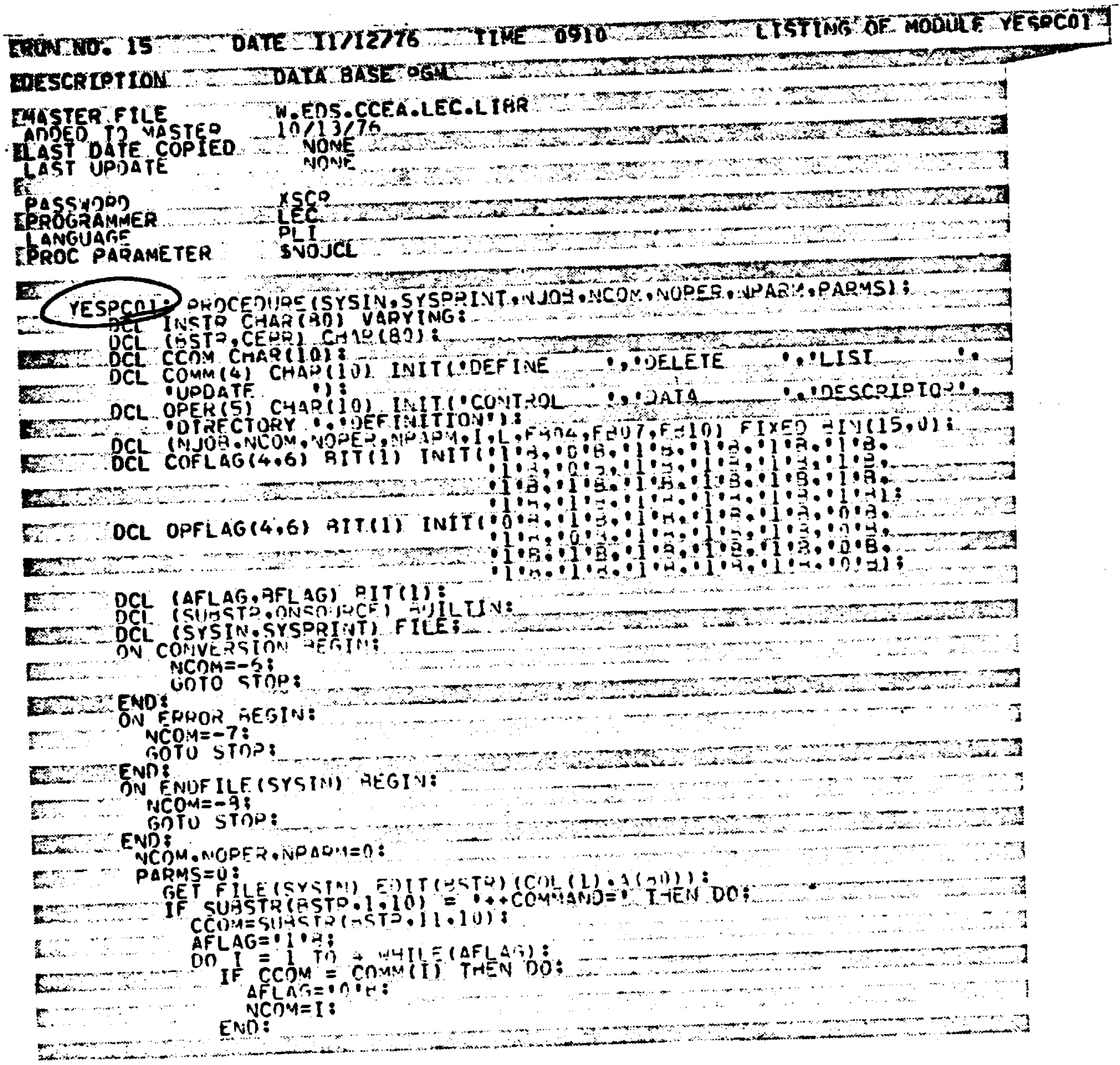
Next page.

3.3.4.6 Listing

Follows flow chart.

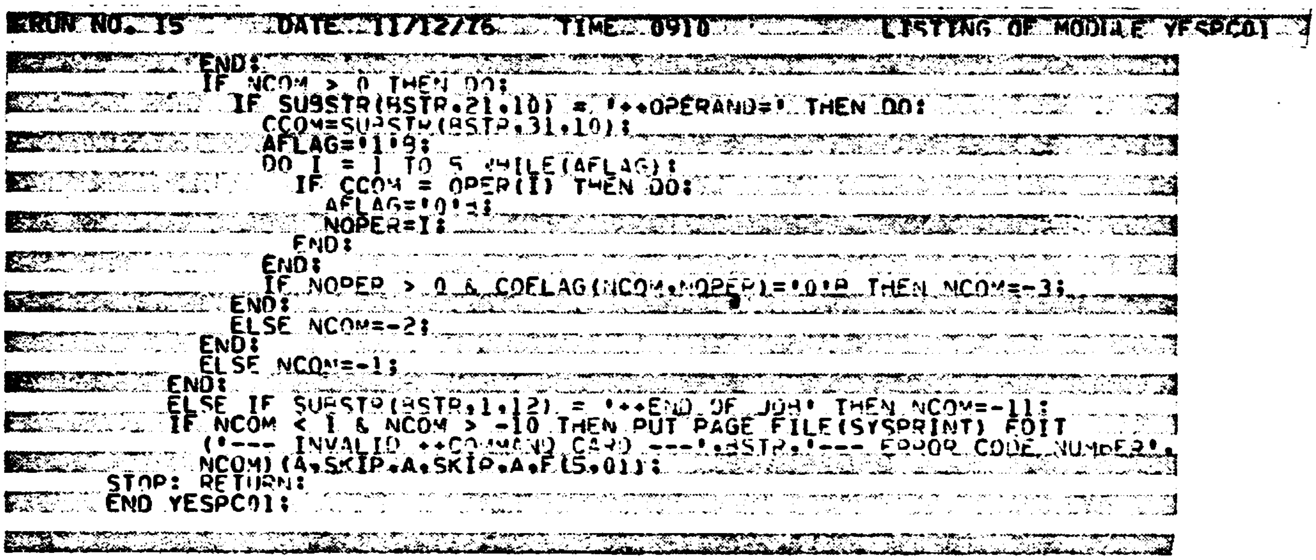

SUBROUTINE
YESPC01
START
NCΦM=0
NΦPER=0
READ CARD
Are characters 1-10 '++COMMAND=' ?
no
Are characters 1-12 '++END OF JOB' ?
yes
NCΦM=-11
no
yes
Are characters 11-20 DEFINE, DELETE, LIST, or UPDATE ?
no
NCΦM=-1
yes
NCΦM=1,2,3,or 4 respectively
Are characters 21-30 '++OPERAND=' ?
no
NCΦM=-2
yes
Are characters 31-40 CONTROL, DATA, DESCRIPTOR, DIRECTORY, or DEFINITION ?
no
NCΦM=-3
NΦPER=1,2,3,4,or 5 respectively
IS NCΦM<1 and NCΦM>-10 ?
yes
write error message
no
RETURN

ORIGINAL PAGE IS
OF POOR QUALITY



```
RUN NO. 15        DATE  11/12/76    TIME  0910            LISTING OF MODULE YESPC01

DESCRIPTION            DATA BASE PGM

MASTER FILE             W.EDS.CCEA.LEC.LIBR
 ADDED TO MASTER        10/13/76
LAST DATE COPIED          NONE
 LAST UPDATE              NONE

 PASSWORD               XSCR
PROGRAMMER              LEC
 LANGUAGE               PLI
PROC PARAMETER          $NOJCL

  YESPC01: PROCEDURE(SYSIN,SYSPRINT,NJOB,NCOM,NOPER,NPARM,PARMS);
        DCL INSTR CHAR(80) VARYING;
        DCL (BSTR,CERR) CHAR(80);
        DCL CCOM CHAR(10);
        DCL COMM(4) CHAR(10) INIT('DEFINE    ','DELETE    ','LIST      ',
            'UPDATE    ');
        DCL OPER(5) CHAR(10) INIT('CONTROL   ','DATA      ','DESCRIPTOR',
            'DIRECTORY ','DEFINITION');
        DCL (NJOB,NCOM,NOPER,NPARM,I,L,F804,F807,F810) FIXED BIN(15,0);
        DCL COFLAG(4,6) BIT(1) INIT('1'B,'0'B,'1'B,'1'B,'1'B,'1'B,
                                    '1'B,'0'B,'1'B,'1'B,'1'B,'1'B,
                                    '1'B,'1'B,'1'B,'1'B,'1'B,'1'B,
                                    '1'B,'1'B,'1'B,'1'B,'1'B,'1'B);
        DCL OPFLAG(4,6) BIT(1) INIT('0'B,'1'B,'1'B,'1'B,'1'B,'0'B,
                                    '1'B,'0'B,'1'B,'1'B,'1'B,'0'B,
                                    '1'B,'1'B,'1'B,'1'B,'1'B,'0'B,
                                    '1'B,'1'B,'1'B,'1'B,'1'B,'0'B);
        DCL (AFLAG,BFLAG) BIT(1);
        DCL (SUBSTR,ONSOURCE) BUILTIN;
        DCL (SYSIN,SYSPRINT) FILE;
        ON CONVERSION BEGIN;
              NCOM=-6;
              GOTO STOP;
        END;
        ON ERROR BEGIN;
           NCOM=-7;
           GOTO STOP;
        END;
        ON ENDFILE(SYSIN) BEGIN;
           NCOM=-8;
           GOTO STOP;
        END;
         NCOM,NOPER,NPARM=0;
          PARMS=0;
            GET FILE(SYSIN) EDIT(BSTR)(COL(1),A(80));
            IF SUBSTR(BSTR,1,10) = '++COMMAND=' THEN DO;
               CCOM=SUBSTR(BSTR,11,10);
               AFLAG='1'B;
               DO I = 1 TO 4 WHILE(AFLAG);
                  IF CCOM = COMM(I) THEN DO;
                     AFLAG='0'B;
                     NCOM=I;
                  END;
```

ORIGINAL PAGE IS
OF POOR QUALITY



RUN NO. 15   DATE 11/12/76   TIME 0910   LISTING OF MODULE YESPC01

```
      END;
      IF NCOM > 0 THEN DO;
        IF SUBSTR(BSTR,21,10) = '++OPERAND=' THEN DO;
          CCOM=SUBSTR(BSTR,31,10);
          AFLAG='1'B;
          DO I = 1 TO 5 WHILE(AFLAG);
            IF CCOM = OPER(I) THEN DO;
              AFLAG='0'B;
              NOPER=I;
            END;
          END;
          IF NOPER > 0 & COFLAG(NCOM,NOPER)='0'B THEN NCOM=-3;
        END;
        ELSE NCOM=-2;
      END;
      ELSE NCOM=-1;
    END;
    ELSE IF SUBSTR(BSTR,1,12) = '++END OF JOB' THEN NCOM=-11;
    IF NCOM < 1 & NCOM > -10 THEN PUT PAGE FILE(SYSPRINT) EDIT
      ('--- INVALID ++COMMAND CARD ---',BSTR,'--- ERROR CODE NUMBER',
      NCOM)(A,SKIP,A,SKIP,A,F(5,0));
STOP: RETURN;
END YESPC01;
```

3.3.5 SELECTION OF TYPE OF DEFINITION (YESDF01)

YESDF01 is called by YESM002 tc select the type of definition to be entered.

3.3.5.1 Linkages

YESDF01 calls YESDF02, YESDF03, YESDF04, and YESDF05. YESDF05 is a dummy subroutine.

3.3.5.2 Interfaces

YESDF01 operates on a code produced by YESPC01.

3.3.5.3 Inputs

See 3.3.5.2.

3.3.5.4 Outputs

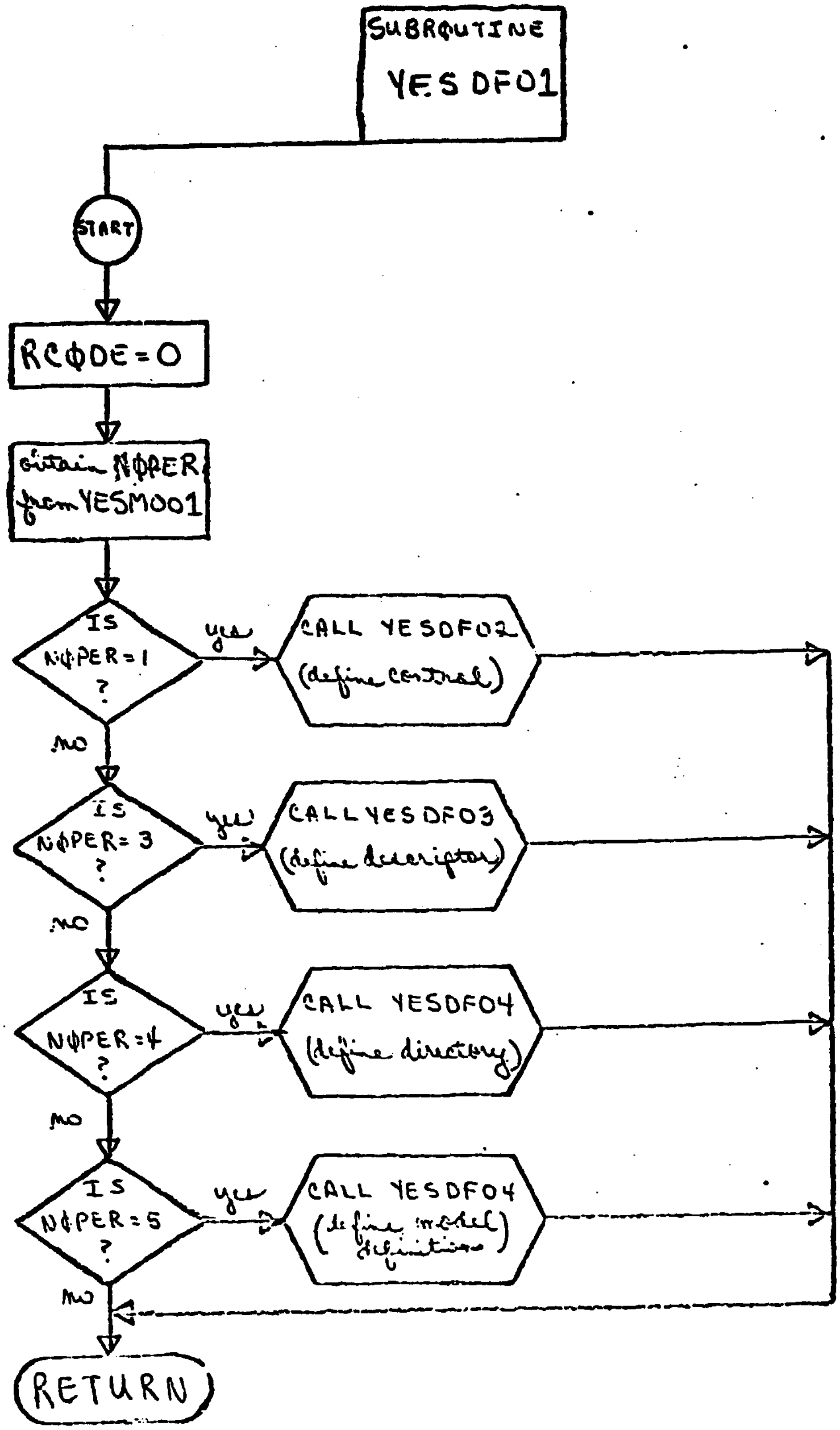
Indirectly – via the called routines.

3.3.5.5 Flow Chart

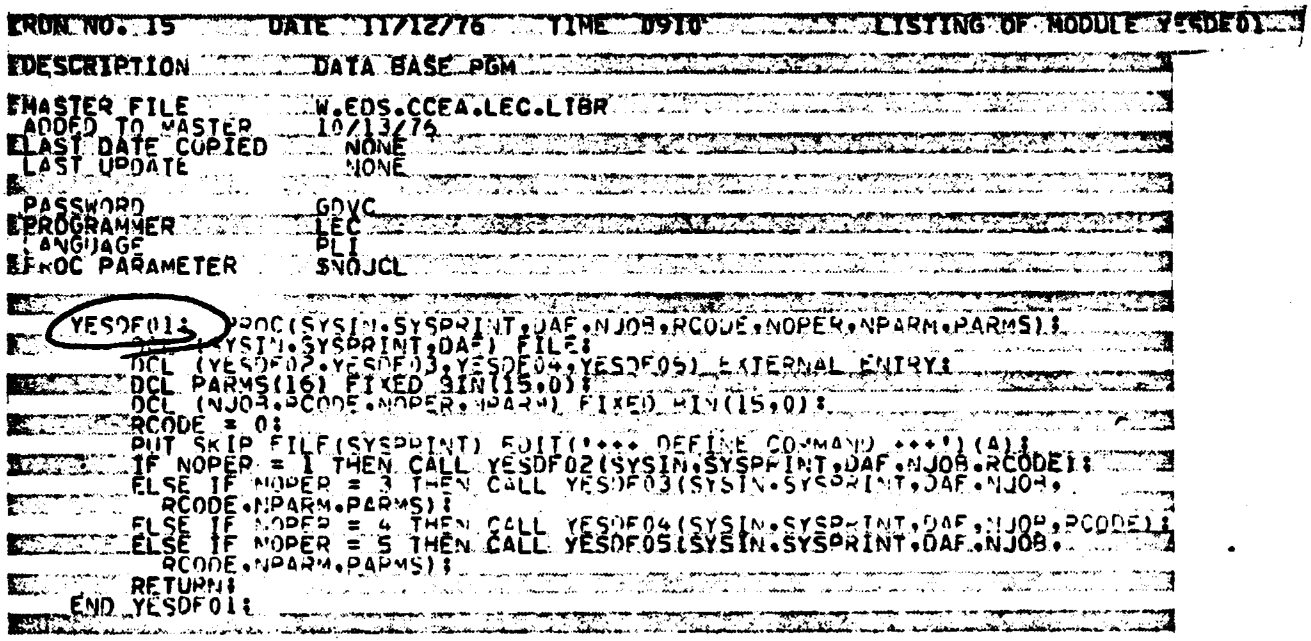
Next page.

3.3.5.6 Listing

Follows flow chart.


SUBROUTINE
YESDF01
START
RCφDE=0
obtain NφPER from YESM001
IS NφPER=1 ?
yes
CALL YESDF02 (define control)
no
IS NφPER=3 ?
yes
CALL YESDF03 (define descriptor)
no
IS NφPER=4 ?
yes
CALL YESDF04 (define directory)
no
IS NφPER=5 ?
yes
CALL YESDF04 (define model definitions)
no
RETURN

ORIGINAL PAGE IS
OF POOR QUALIT



```
RUN NO. 15      DATE 11/12/76     TIME 0910                LISTING OF MODULE YESDF01

DESCRIPTION           DATA BASE PGM

MASTER FILE           W.EDS.CCEA.LEC.LIBR
 ADDED TO MASTER      10/13/76
LAST DATE COPIED        NONE
 LAST UPDATE            NONE

 PASSWORD             GDVC
PROGRAMMER            LEC
 LANGUAGE             PLI
PROC PARAMETER        $NOJCL

   YESDF01:  PROC(SYSIN.SYSPRINT.DAF.NJOB.RCODE.NOPER.NPARM.PARMS);
         DCL (SYSIN.SYSPRINT.DAF) FILE;
         DCL (YESDF02.YESDF03.YESDF04.YESDF05) EXTERNAL ENTRY;
         DCL PARMS(16) FIXED BIN(15.0);
         DCL (NJOB.RCODE.NOPER.NPARM) FIXED BIN(15.0);
         RCODE = 0;
         PUT SKIP FILE(SYSPRINT) EDIT('*** DEFINE COMMAND ***')(A);
         IF NOPER = 1 THEN CALL YESDF02(SYSIN.SYSPRINT.DAF.NJOB.RCODE);
         ELSE IF NOPER = 3 THEN CALL YESDF03(SYSIN.SYSPRINT.DAF.NJOB.
            RCODE.NPARM.PARMS);
         ELSE IF NOPER = 4 THEN CALL YESDF04(SYSIN.SYSPRINT.DAF.NJOB.RCODE);
         ELSE IF NOPER = 5 THEN CALL YESDF05(SYSIN.SYSPRINT.DAF.NJOB.
            RCODE.NPARM.PARMS);
         RETURN;
   END YESDF01;
```

3.3.6 CONTROL BLOCK DEFINITION PROGRAM (YESDF02)

YESDF02 enters control block information in the data base.

3.3.6.1 Linkages

None.

3.3.6.2 Interfaces

None.

3.3.6.3 Inputs

Control block definition cards.

3.3.6.4 Outputs

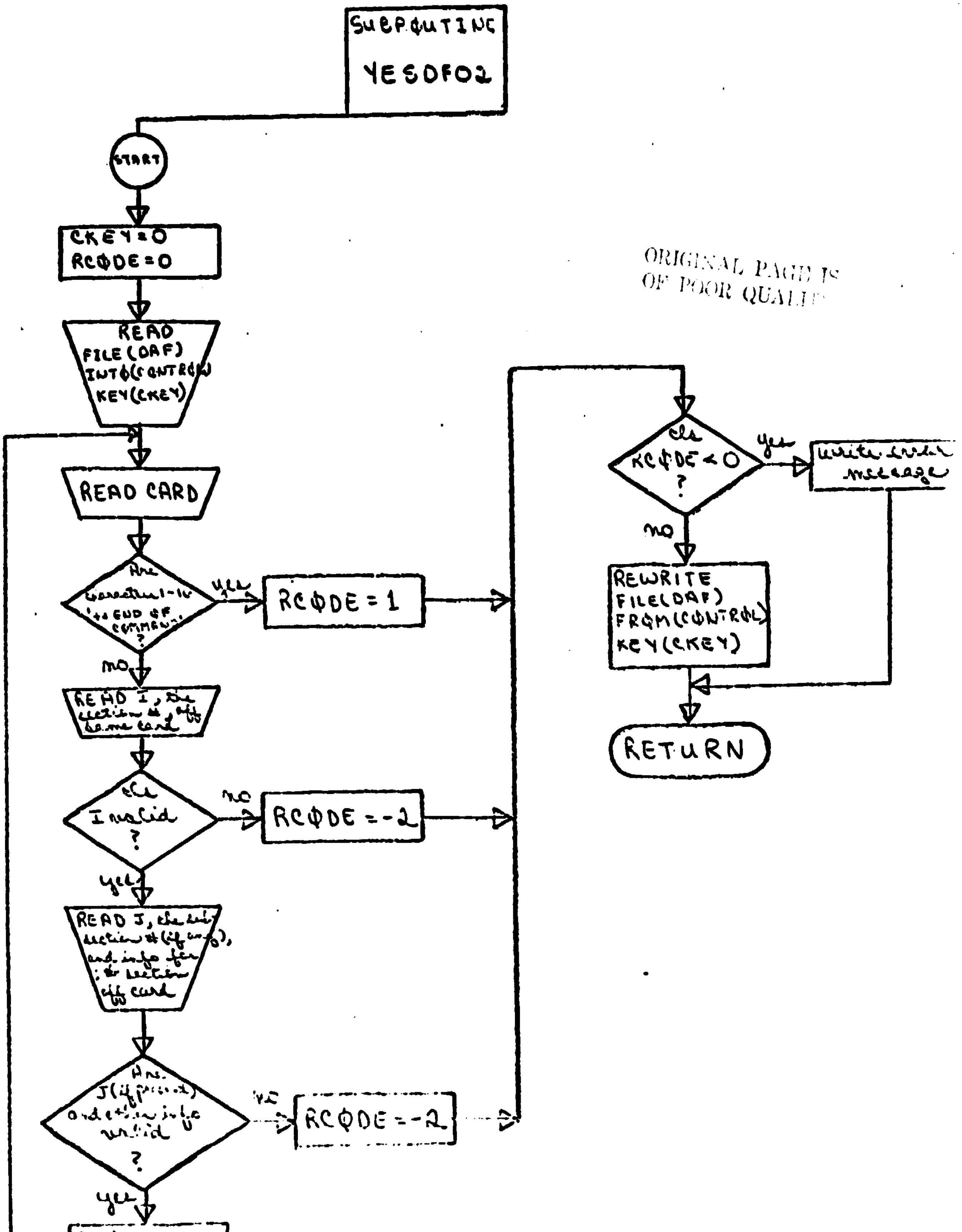
Defined control block to data base.

3.3.6.5 Flow Chart

Next page.

3.3.6.6 Listing

Follows flow chart.


SUBRØUTINE
YESDF02
START
CKEY=0
RCØDE=0
READ FILE(DAF) INTØ(CONTROL) KEY(CKEY)
READ CARD
Are columns 1-16 '** END OF COMMAND' ?
yes
RCØDE=1
no
READ I, the section # off the card
Is I valid ?
no
RCØDE=-2
yes
READ J, the sub-section # (if any), and info for the section off card
Are J (if present) and other info valid ?
no
RCØDE=-2
yes
Assign new values for the section and the sub-section (if any)
Is RCØDE<0 ?
yes
write error message
no
REWRITE FILE(DAF) FRØM(CØNTRØL) KEY(CKEY)
RETURN

ORIGINAL PAGE IS
OF POOR QUALITY

ORIGINAL PAGE IS
OF POOR QUALITY

```
RUN NO. 15       DATE  11/12/76     TIME  0910              LISTING OF MODULE YESDF02
DESCRIPTION            DATA BASE PGM

MASTER FILE            #.EDS.CCFA.LEC.LIBR
ADDED TO MASTER        10/13/75
LAST DATE COPIED        NONE
LAST UPDATE             NONE

PASSWORD               XPCB
PROGRAMMER             LEC
LANGUAGE               PLI
PROC PARAMETER         SNOJCL

 YESDF02:PROCEDURE(SYSIN.SYSPRINT.DAF.NJOB.RCODE);
 /*                                                                       */
 /*THIS PROGRAM IS CALLED BY YESDF01 TO DEFINE THE CONTROL BLOCK          */
 /*                                                                       */
   DCL 1 CONTROL.
           2 FILEID CHAR(8).
           2 NUMPASS FIXED BIN(15.0).
           2 PASS(8) CHAR (8).
           2 NUMLEV FIXED BIN(15.0).
           2 LEVNAME(9) CHAR(24).
           2 NUMCODE FIXED BIN(15.0).
           2 CODE(32).
             3 CODENUM FIXED BIN(15.0).
             3 UNITNUM FIXED BIN(15.0).
             3 BASE FIXED BIN(15.0).
             3 SCALE FIXED BIN(15.0).
             3 CODENAME CHAR(24).
             3 UNITNAME CHAR(24).
           2 NUMONE FIXED BIN(15,0).
           2 ONE(24).
             3 CODEINUM FIXED BIN(15.0).
             3 NUMDIRS FIXED BIN(15.0).
             3 RECNUM FIXED BIN(15.0).
             3 DISPLACE FIXED BIN(15.0).
             3 NAME CHAR(24).
           2 FILERECS FIXED BIN(15,0).
           2 REC(601).
             3 RECTYPE FIXED BIN(15.0).
             3 FRESPACE FIXED BIN(15.0).
             3 LOCATION FIXED BIN(15.0);
   DCL CKEY FIXED BIN(10.0);
   DCL PI POINTER;
   DCL D CHAR(8) BASED(PI);
   DCL B BIT(64) BASED(PI);
   DCL AFLAG BIT(1);
   DCL (I.J.NJOB.RCODE.XNPASS.XNLEV.XNCODE.
      CNO.UNO.XBASE.YSCALE.XONE.CINO.NDO.RECNO.
      XDIS.RT.FS.LOC.FR) FIXED BIN(15.0);
   DCL (XFLEID.PW) CHAR(8);
   DCL (XLNAM.CNAM.UNAM.XNAM) CHAR(24);
   DCL INSTR CHAR(80);
   DCL (SYSIN.SYSPRINT.DAF) FILE;
   DCL SUBSTR BUILTIN;
  PUT SKIP FILE(SYSPRINT) EDIT ('***DEFINE CONTROL PROGRAM***') (A);
  ON CONVERSION BEGIN;
    PUT SKIP FILE(SYSPRINT) EDIT ('***INVALID INPUT CARD***') (A);
    PUT SKIP FILE(SYSPRINT) EDIT (INSTR) (A(50));
    RCODE=-1;
    GOTO STOP;
  END;
  ON ENDFILE(SYSIN) BEGIN;
    PUT SKIP FILE (SYSPRINT) EDIT
     ('****ERROR - END OF FILE SYSIN ENCOUNTERED****') (A);
      RCODE=-1;
      GOTO STOP;
  END;
  CKEY=0;
   READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
     RCODE=0;
     AFLAG='1'B;
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
       DO WHILE(AFLAG);
          GET FILE(SYSIN) EDIT(INSTR) (COL(1).A(80));
          IF SUBSTR(INSTR.1.15)='**END_OF_COMMAND' THEN DO;
             AFLAG='0'B;
             RCODE=1;
          END;
          ELSE DO;
             GET STRING(INSTR) EDIT (I) (X(5).F(2.0));
             IF I<0 | I>11 THEN DO;
                PUT SKIP FILE(SYSPRINT) EDIT ('****'.I.
                 ' IS AN INVALID SECTION NUMBER****')(A.F(4.0).A);
                AFLAG='0'B; RCODE=-2;
             END;
              ELSE IF I=1 THEN DO;
              GET STRING(INSTR) EDIT(XFLEID) (R(F12));
              CONTROL.FILEID=XFLEID;
             END;
             ELSE IF I=2 THEN DO;
                GET STRING(INSTR) EDIT(XNPASS) (R(F11));
                   IF XNPASS<1 | XNPASS>9 THEN DO;
                     PUT SKIP FILE(SYSPRINT) EDIT('****'.XNPASS.
                        'IS AN INVALID NUMBER OF PASSWORDS****')(A.F(4.0).A);
                      AFLAG='0'B; RCODE=-2;
                  END;
                CONTROL.NUMPASS=XNPASS;
             END;
             ELSE IF I=3 THEN DO;
                ALLOCATE D SET(PI);
                GET STRING(INSTR) EDIT(J.PW) (R(F31));
                   IF J<1 | J>CONTROL.NUMPASS THEN DO;
                     IF J<1 THEN PUT SKIP FILE(SYSPRINT) EDIT('****'.J.
                       'IS AN INVALID NUMBER OF PASSWORDS****')
                          (A.F(4.0).A);
                  ELSE PUT SKIP FILE(SYSPRINT) EDIT ('****'.J.
                    'IS LARGER THAN CONTROL.NUMPASS****')
                     (A.F(4.0).A);
                    AFLAG='0'B; RCODE=-2;
                  END;
                D=PW;
                B=-B;
                CONTROL.PASS(J)=D;
             END;
             ELSE IF I=4 THEN DO;
                GET STRING(INSTR) EDIT(XNLEV) (R(F11));
                   IF XNLEV<1 | XNLEV>9 THEN DO;
                     PUT SKIP FILE(SYSPRINT) EDIT('****'.XNLEV.
                        'IS AN INVALID NUMBER OF LEVELS****')(A.F(4.0).A);
                     AFLAG='0'B; RCODE=-2;
                  END;
                CONTROL.NUMLEV=XNLEV;
             END;
             ELSE IF I=5 THEN DO;
                GET STRING(INSTR) EDIT(J.XLNAM)(R(F32));
                   IF J<1 | J>CONTROL.NUMLEV THEN DO;
                   IF J<1 THEN PUT SKIP FILE(SYSPRINT) EDIT ('****'.J.
              'IS AN INVALID NUMBER OF LEVELS****')(A.F(4.0).A);
                     ELSE PUT SKIP FILE(SYSPRINT) EDIT('****'.J.
                       'IS LARGER THAN CONTROL.NUMLEV****')(A.F(4.0).A);
                     AFLAG='0'B; RCODE=-2;
                  END;
                CONTROL.LEVNAME(J)=XLNAM;
             END;
             ELSE IF I=6 THEN DO;
                GET STRING(INSTR) EDIT(XNCODE) (R(F11));
                   IF XNCODE<0 | XNCODE>32 THEN DO;
                     PUT SKIP FILE(SYSPRINT) EDIT('****'.XNCODE.
                       'IS AN INVALID NUMBER OF CODES****')(A.F(4.0).A);
                     AFLAG='0'B; RCODE=-2;
                  END;
                CONTROL.NUMCODE=XNCODE;
             END;
             ELSE IF I=7 THEN DO;
                GET STRING(INSTR) EDIT(J.CNO.UNO.XBASE.XSCALE.
                   CNAM.UNAM)(R(F41));
                   IF J<0 | J>CONTROL.NUMCODE THEN DO;
                     IF J<0 THEN PUT SKIP FILE(SYSPRINT) EDIT ('****'.J.
                       'IS AN INVALID NUMBER OF CODES****')(A.F(4.0).A);
                     ELSE PUT SKIP FILE(SYSPRINT) EDIT('****'.J.
                       'IS LARGER THAN CONTROL.NUMCODE****')(A.F(4.0).A);
                     AFLAG='0'B; RCODE=-2;
                  END;
```

```
            CONTROL.CODE(J).CODENUM=CNO;
            CONTROL.CODE(J).UNITNUM=UNO;
            CONTROL.CODE(J).BASE=XBASE;
            CONTROL.CODE(J).SCALE=XSCALE;
            CONTROL.CODE(J).CODENAME=CNAM;
            CONTROL.CODE(J).UNITNAME=UNAM;
         END;
         ELSE IF I=8 THEN DO;
            GET STRING(INSTR) EDIT(XONE) (R(F11));
               IF XONE<0 | XONE>24 THEN DO;
                  PUT SKIP FILE(SYSPRINT) EDIT('***',XONE,
                     'IS AN INVALID NUMBER OF ONES***')(A,F(4,0),A);
                  AFLAG='0'B; RCODE=-2;
               END;
            CONTROL.NUMONE=XONE;
         END;
         ELSE IF I=9 THEN DO;
            GET STRING(INSTR) EDIT(J,C1NO,NOD,RECNO,XDIS,XNAM)
               (R(F42));
               IF J<1 | J>CONTROL.NUMONE THEN DO;
                  IF J<1 THEN PUT SKIP FILE(SYSPRINT) EDIT ('***',
                     J,'IS AN INVALID NUMBER OF ONES***')(A,F(4,0),A);
                  ELSE PUT SKIP FILE(SYSPRINT) EDIT('***',J,
                     'IS LARGER THAN CONTROL.NUMONE***')(A,F(4,0),A);
                  AFLAG='0'B; RCODE=-2;
               END;
            CONTROL.ONE(J).CODE1NUM=C1NO;
            CONTROL.ONE(J).NUMDIRS=NOD;
            CONTROL.ONE(J).RECNUM=RECNO;
            CONTROL.ONE(J).DISPLACE=XDIS;
            CONTROL.ONE(J).NAME=XNAM;
         END;
         ELSE IF I=10 THEN DO;
            GET STRING(INSTR) EDIT(FR) (R(F11));
               IF FR<0 | FR>60) THEN DO;
                  PUT SKIP FILE(SYSPRINT) EDIT('***',FR,
                     'IS AN INVALID NUMBER OF FILERECS***')(A,F(4,0),A);
                  AFLAG='0'B; RCODE=-2;
               END;
            CONTROL.FILERECS=FR;
         END;
         ELSE DO;
            GET STRING(INSTR) EDIT(J,RT,FS,LOC) (R(F43));
               IF J<0 | J>CONTROL.FILERECS THEN DO;
                  IF J<0 THEN PUT SKIP FILE(SYSPRINT) EDIT('***',J,
                     'IS AN INVALID NUMBER OF FILERECS***')(A,F(4,0),A);
               ELSE PUT SKIP FILE(SYSPRINT) EDIT('***',J,
                     'IS LARGER THAN CONTROL.FILERECS***')(A,F(4,0),A);
               AFLAG='0'B; RCODE=-2;
               END;
            CONTROL.REC(J).RECTYPE=RT;
            CONTROL.REC(J).FREESPACE=FS;
            CONTROL.REC(J).LOCATION=LOC;
         END;
      END;
    END;
    IF AFLAG='0'B & RCODE<0 THEN PUT SKIP FILE(SYSPRINT)
      EDIT (INSTR) (A);
    IF RCODE<0 THEN GOTO STOP;
    REWRITE FILE(DAF) FROM(CONTROL) KEY(CKEY);
    PUT SKIP FILE(SYSPRINT) EDIT
     ('***CONTROL BLOCK DEFINED OR UPDATED***') (A);
    F12: FORMAT(X(11),A(8));
    F11: FORMAT(X(11),F(4,0));
    F31: FORMAT(X(7),F(3,0),X(1),A(8));
    F32: FORMAT(X(7),F(3,0),X(1),A(24));
    F41: FORMAT(X(7),F(3,0),4 F(5,0),X(1),A(24),X(1),A(24));
    F42: FORMAT(X(7),F(3,0),4 F(5,0),X(1),A(24));
    F43: FORMAT(X(7),F(3,0),3 F(5,0));
 FREE D;
 STOP: RETURN;
 END YFSUF02;
```

3.3.7 DATA DESCRIPTOR ENTRY (YESDF03)

YESDF03 enters data descriptors into the data base.

3.3.7.1 Linkages

YESDFO3 calls GETDIR.

3.3.7.2 Interfaces

A valid directory entry must exist before invocation.

3.3.7.3 Inputs

Data descriptor cards.

3.3.7.4 Outputs

New data definitions in data base.

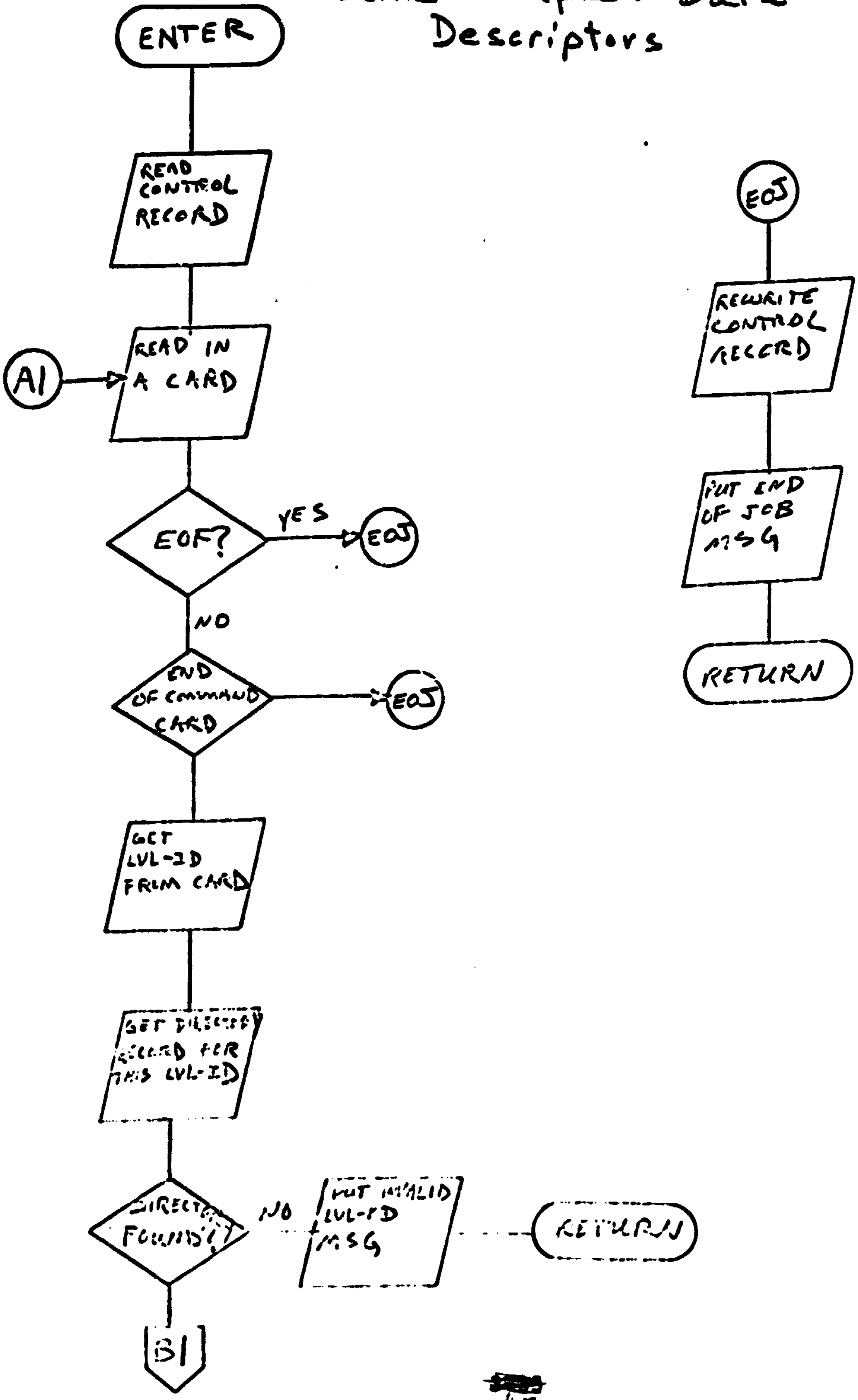
3.3.7.5 Flow Chart

Next page.

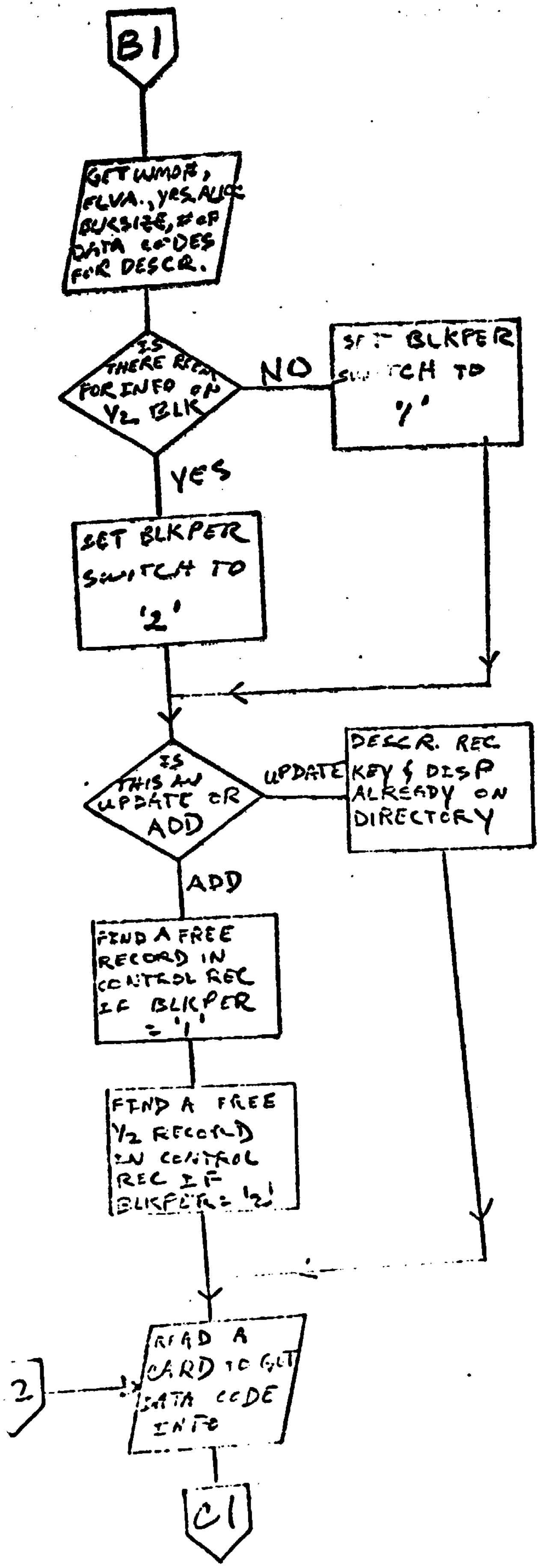
3.3.7.6 Listing

Follows flow chart.

YESDF03
Define or Update Data Descriptors

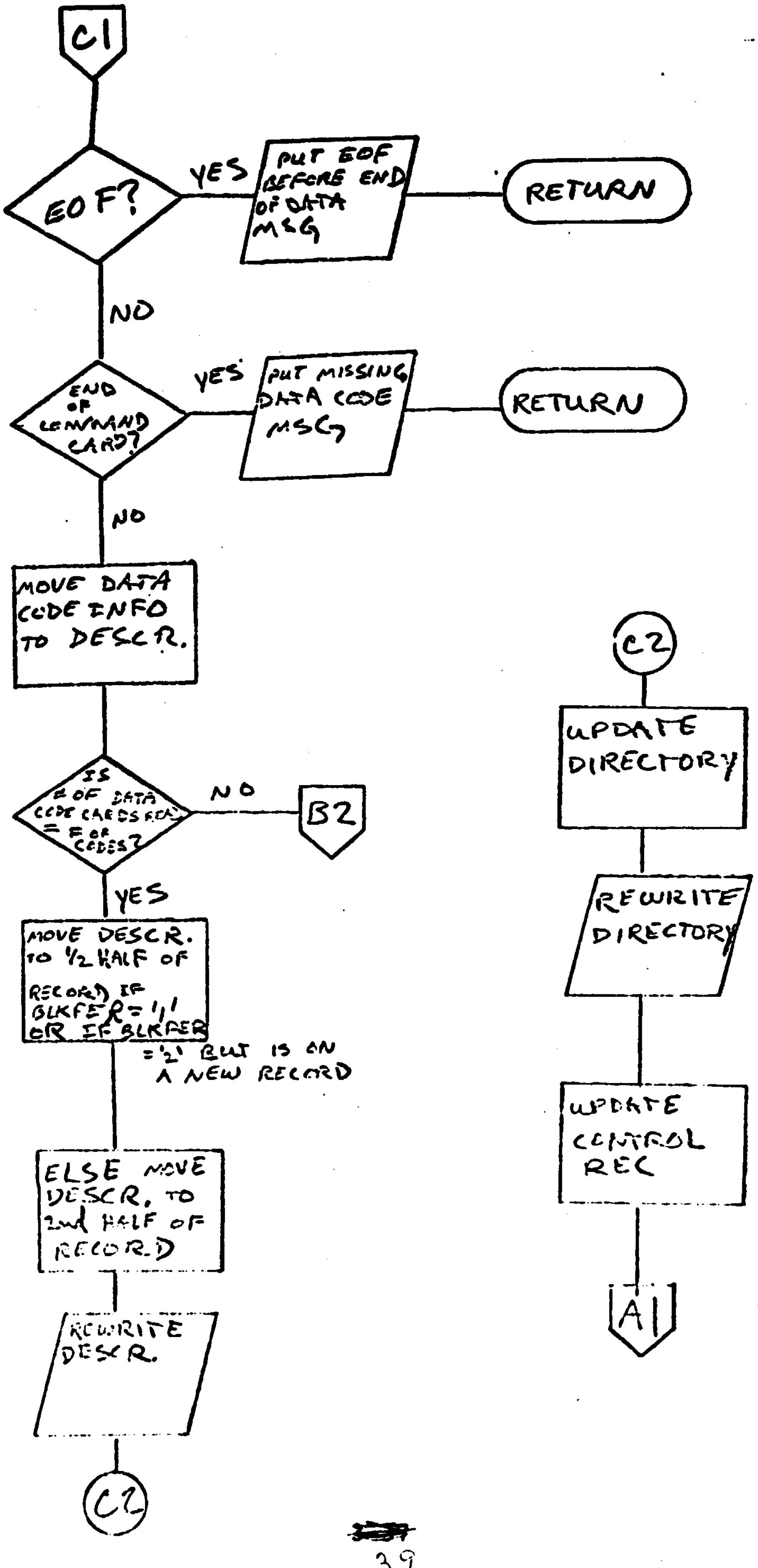


YESDF03, con't.


B1
GET WMO#, FLVA., YES.ALLC BLKSIZE, # OF DATA CODES FOR DESCR.
IS THERE ROOM FOR INFO ON 1/2 BLK
NO
SET BLKPER SWITCH TO '1'
YES
SET BLKPER SWITCH TO '2'
IS THIS AN UPDATE OR ADD
UPDATE
DESCR. REC. KEY & DISP ALREADY ON DIRECTORY
ADD
FIND A FREE RECORD IN CONTROL REC IF BLKPER = '1'
FIND A FREE 1/2 RECORD IN CONTROL REC IF BLKPER = '2'
READ A CARD TO GET DATA CODE INFO
2
C1

ORIGINAL PAGE IS OF POOR QUALITY

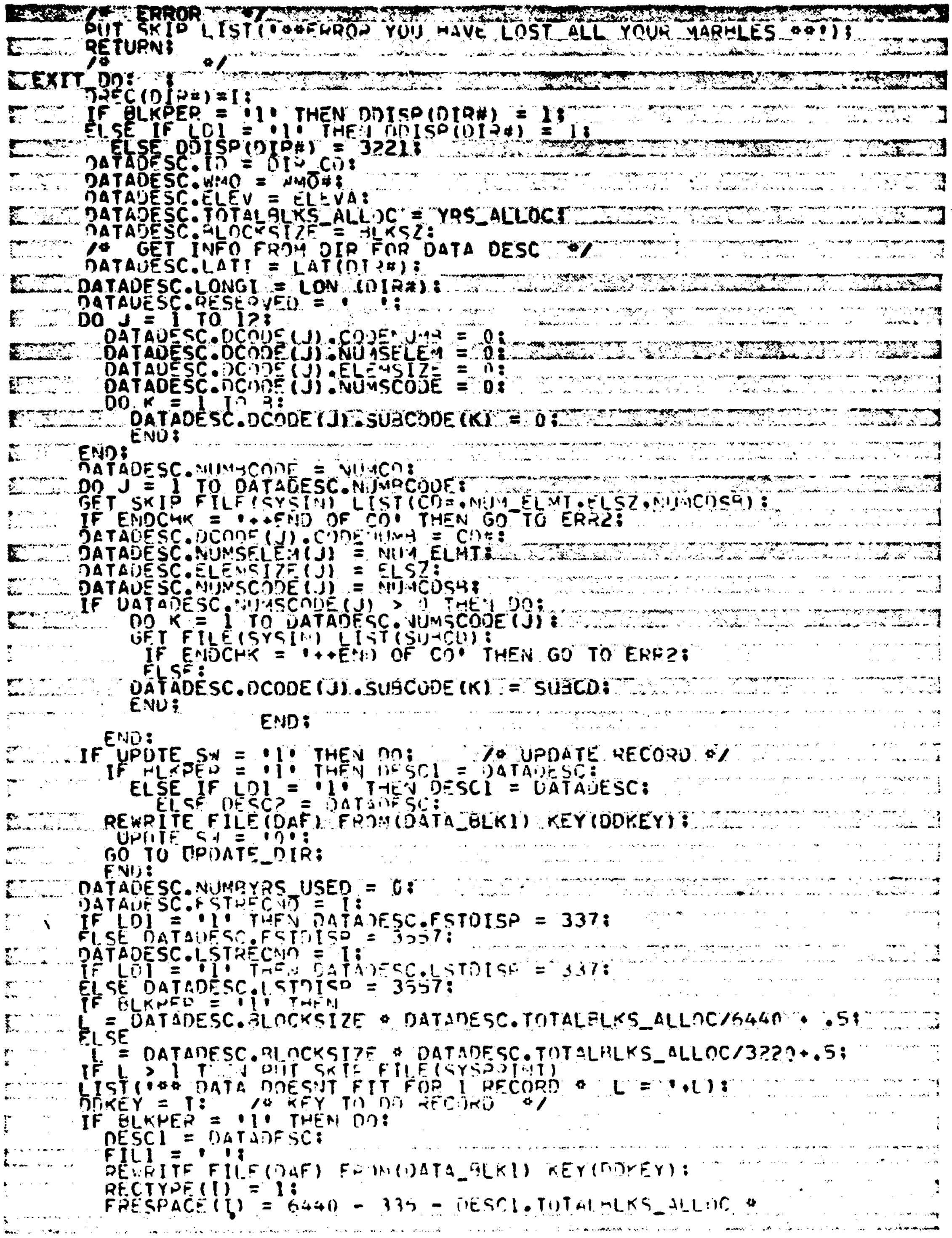
ORIGINAL PAGE IS
OF POOR QUA


C1
EOF?
YES
PUT EOF BEFORE END OF DATA MSG
RETURN
NO
END OF COMMAND CARD?
YES
PUT MISSING DATA CODE MSG
RETURN
NO
MOVE DATA CODE INFO TO DESCR.
IS # OF DATA CODE CARDS RCA = # OF CODES?
NO
B2
YES
MOVE DESCR. TO 1/2 HALF OF RECORD IF BLKFER = '1' OR IF BLKFER = '2' BUT IS ON A NEW RECORD
ELSE MOVE DESCR. TO 2nd HALF OF RECORD
REWRITE DESCR.
C2
C2
UPDATE DIRECTORY
REWRITE DIRECTORY
UPDATE CONTROL REC
A1

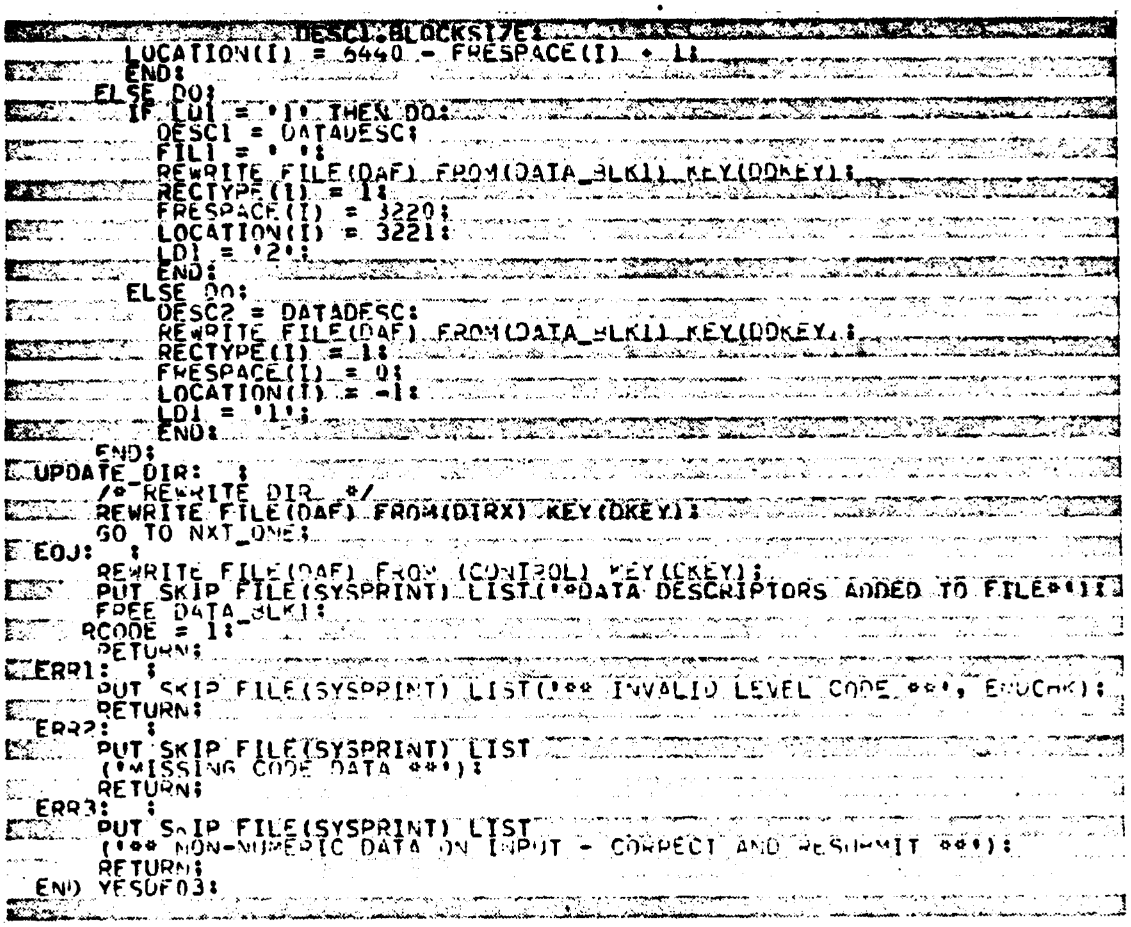
ORIGINAL PAGE IS
OF POOR QUALITY

```
RUN NO. 15        DATE 11/12/75    TIME 0910                LISTING OF MODULE YESDF03
DESCRIPTION             DATA BASE PGM

MASTER FILE             W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER         10/13/76
LAST DATE COPIED          NONE
LAST UPDATE               NONE

PASSWORD                RXHC
PROGRAMMER              LEC
LANGUAGE                PLI
PROC PARAMETER          SNOJCL

  YESDF03:  PROC(SYSIN,SYSPRINT,DAF,NJOB,RCODE);
  /*                                                                    */
  /*  THIS PGM IS CALLED BY YESDF01 TO DEFINE DATA DESCRITORS,          */
  /*                                                                    */
       DCL (SYSIN,SYSPRINT,DAF) FILE;
       DCL GETDIR EXTERNAL ENTRY;
     ON ENDFILE(SYSIN)  GO TO EOJ;
       DCL LVL_ID_CD   PIC '(11)9';
       DCL ENDCHK CHAR(11);
       DCL CKEY FIXED BIN(15) INIT(0);
       DCL PC     FIXED BIN(15) INIT(0);
       DCL DKEY FIXED BIN(15);
       DCL (DDKEY,DIR#) FIXED BIN(15);
       DCL (WMO#,ELEVA,YRS_ALLOC,BLKSZ,NUMCD)  FIXED DEC(5,0);
       DCL (CD#,NUM_ELMT,ELSZ,NUMCDS9,SUBCD)   FIXED DEC(5,0);
       DCL (NJOB,RCODE) FIXED BIN(15);
       DCL REC_BLK CHAR(6440);
       DCL DIR_CD FIXED BIN(31);
     DCL 1 CONTROL,
             2 FILEID CHAR(8),
             2 NUMPASS FIXED BIN(15,0),
             2 PASS(8) CHAR(8),
             2 NUMLEV FIXED BIN(15,0),
             2 LEVNAME(8) CHAR(24),
             2 NUMCODE FIXED BIN(15,0),
             2 CODE(32),
               3 CODENUM FIXED BIN(15,0),
               3 UNITNUM FIXED BIN(15,0),
               3 BASE FIXED BIN(15,0),
               3 SCALE FIXED BIN(15,0),
               3 CODENAME CHAR(24),
               3 UNITNAME CHAR(24),
             2 NUMONE FIXED BIN(15,0),
             2 ONE(24),
               3 CODEINUM FIXED BIN(15,0),
               3 NUMDIRS FIXED BIN(15,0),
               3 RECNUM FIXED BIN(15,0),
               3 DISPLACE FIXED BIN(15,0),
               3 NAME CHAR(24),
             2 FILERECS FIXED BIN(15,0),
             2 REC(601),
               3 RECTYPE FIXED BIN(15,0),
               3 FRESPACE FIXED BIN(15,0),
               3 LOCATION FIXED BIN(15,0);
     DCL 1 DIRX,
             2 DIR (84),
             3 LEVNUM FIXED BIN(15,0),
             3 CODENUMS FIXED BIN(15,0),
             3 LAT FIXED BIN(15,0),
             3 LON FIXED BIN(15,0),
             3 DIRNAME CHAR(24),
             3 PREC FIXED BIN(15,0),
             3 PDISP FIXED BIN(15,0),
             3 BREC FIXED BIN(15,0),
             3 BDISP FIXED BIN(15,0),
             3 CREC FIXED BIN(15,0),
             3 CDISP FIXED BIN(15,0),
             3 DREC FIXED BIN(15,0),
             3 DDISP FIXED BIN(15,0),
             3 LEVCODE FIXED BIN(31,0),
           3 MODEL(4),
              4 CROP FIXED BIN(15,0),
              4 MREC FIXED BIN(15,0),
             4 MDISP FIXED BIN(15,0),
       2 FILLER CHAR (56);
     DCL 1 DATADESC,
             2 ID                 FIXED BIN(31,0),
             2 WMO                FIXED BIN(31,0),
             2 LATI               FIXED BIN(15,0),
             2 LONGI              FIXED BIN(15,0),
             2 ELEV               FIXED BIN(15,0),
             2 TOTALBLKS_ALLOC    FIXED BIN(15,0),
             2 NUMBYRS_USED       FIXED BIN(15,0),
             2 BLOCKSIZE          FIXED BIN(15,0),
             2 FSTRECNO           FIXED BIN(15,0),
             2 FSTDISP            FIXED BIN(15,0),
             2 LSTRECNO           FIXED BIN(15,0),
             2 LSTDISP            FIXED BIN(15,0),
             2 RESERVED           CHAR(18),
             2 NUMBCODE           FIXED BIN(15,0),
             2 DCODE(12),
               3 CODENUMB            FIXED BIN(15,0),
               3 NUMBELEM            FIXED BIN(15,0),
               3 ELEMSIZE            FIXED BIN(15,0),
               3 NUMSCODE            FIXED BIN(15,0),
               3 SUBCODE(8)          FIXED BIN(15,0),
          2  FILLER2 CHAR(2424);
     DCL 1 DATA_BLK1 BASED(D),
             2  DESC1 LIKE DATADESC,
             2  FIL1  CHAR(3220);
     DCL 1 DATA_BLK2 BASED(D),
             2  FIL2  CHAR(3220),
             2  DESC2 LIKE DATADESC;
     DCL BLKPER CHAR(1) INIT('1');
     DCL LD1     CHAR(1) INIT('1');
     DCL UPDTE_SW  CHAR(1) INIT('0');   /* 0=ADD  1=UPDATE  */
     ALLOCATE DATA_BLK1;
   ON CONVERSION GO TO ERR3;
    RCODE= -1;
     DATADESC.FILLER2 = '     ';
     READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
 NXT_ONE:  ;
     GET SKIP FILE(SYSIN) EDIT(ENDCHK)(A(11));
     IF ENDCHK = '**END OF CO'  THEN GO TO EOJ;
     LVL_ID_CD = TRANSLATE(ENDCHK,'0',' ');
     DIR_CD = LVL_ID_CD;
     CALL GETDIR(DIR_CD,DIRX,CONTROL,DAF,DKEY,DIR#,RC);
     IF RC = -1 THEN DO;
       PUT SKIP DATA(LVL_ID_CD,DIR#,RC);
       GO TO ERR1;
       END;
     GET FILE(SYSIN) LIST(WMO#,ELEVA,YRS_ALLOC,BLKSZ,NUMCD);
     IF (YRS_ALLOC * BLKSZ + 366) < 3220 THEN BLKPER = '2';
     ELSE BLKPER = '1';
     IF DREC(DIR#) ¬= -1 THEN DO;
        /*  RECORD ALREADY ASSIGNED  */
        /*  THIS IS AN UPDATE        */
       UPDTE_SW = '1';
      I = DREC(DIR#);
     IF RECTYPE(I) ¬= 1 THEN DO;
       PUT SKIP LIST('**WARNING -- CONTROL BLOCK WAS NOT UPDATED
 THIS UPDATE WILL TRY TO CORRECT THE CONTROL BLOCK **');
       RECTYPE(I) = 1;
       END;
       DDKEY = I;
       READ FILE(DAF) INTO(DATA_BLK1) KEY(DDKEY);
       IF BLKPER = '1' THEN  DATADESC = DESC1;
         ELSE  IF LD1 = '1' THEN  DATADESC = DESC1;
               ELSE DATADESC = DESC2;
       GO TO EXIT_DO;
       END;
     /*  OTHERWISE                                      */
     /*  LOOK IN CONTROL BLK FOR 1ST AVAILABLE RECORD   */
     DO I = 1 TO FILERECS;
     IF RECTYPE(I) = 1 THEN
       IF BLKPER = '2' THEN
         IF FRESPACE(I) = 3220 THEN GO TO EXIT_DO;
     IF RECTYPE(I) = 0 THEN GO TO EXIT_DO;
     END;
```

ORIGINAL PAGE IS
OF POOR QUALITY



```
      /*  ERROR     */
      PUT SKIP LIST('**ERROR YOU HAVE LOST_ALL YOUR MARBLES **');
      RETURN;
      /*          */
EXIT_DO: ;
      DREC(DIR#)=I;
      IF BLKPER = '1' THEN DDISP(DIR#) = 1;
      ELSE IF LD1 = '1' THEN DDISP(DIR#) = 1;
        ELSE DDISP(DIR#) = 3221;
      DATADESC.ID = DIR_CO;
      DATADESC.WMO = WMO#;
      DATADESC.ELEV = ELEVA;
      DATADESC.TOTALBLKS_ALLOC = YRS_ALLOC;
      DATADESC.BLOCKSIZE = BLKSZ;
      /*  GET INFO FROM DIR FOR DATA DESC   */
      DATADESC.LATI = LAT(DIR#);
     DATADESC.LONGI = LON (DIR#);
     DATADESC.RESERVED = '  ';
     DO J = 1 TO 12;
        DATADESC.DCODE(J).CODENUMB = 0;
        DATADESC.DCODE(J).NUMSELEM = 0;
        DATADESC.DCODE(J).ELEMSIZE = 0;
        DATADESC.DCODE(J).NUMSCODE = 0;
        DO K = 1 TO 8;
          DATADESC.DCODE(J).SUBCODE(K) = 0;
          END;
     END;
     DATADESC.NUMBCODE = NUMCO;
     DO J = 1 TO DATADESC.NUMBCODE;
     GET SKIP FILE(SYSIN) LIST(CO#,NUM_ELMT,ELSZ,NUMCDSB);
     IF ENDCHK = '++END OF CO' THEN GO TO ERR2;
     DATADESC.DCODE(J).CODENUMB = CO#;
     DATADESC.NUMSELEM(J) = NUM_ELMT;
     DATADESC.ELEMSIZE(J) = ELSZ;
     DATADESC.NUMSCODE(J) = NUMCDSB;
     IF DATADESC.NUMSCODE(J) > 0 THEN DO;
          DO K = 1 TO DATADESC.NUMSCODE(J);
          GET FILE(SYSIN) LIST(SUBCD);
           IF ENDCHK = '++END OF CO' THEN GO TO ERR2;
           ELSE;
          DATADESC.DCODE(J).SUBCODE(K) = SUBCD;
          END;
                    END;
        END;
     IF UPDTE_SW = '1' THEN DO;      /* UPDATE RECORD */
        IF BLKPER = '1' THEN DESC1 = DATADESC;
          ELSE IF LD1 = '1' THEN DESC1 = DATADESC;
            ELSE DESC2 = DATADESC;
        REWRITE FILE(DAF) FROM(DATA_BLK1) KEY(DDKEY);
         UPDTE_SW = '0';
        GO TO UPDATE_DIR;
        END;
     DATADESC.NUMBYRS_USED = 0;
     DATADESC.FSTRECNO = I;
     IF LD1 = '1' THEN DATADESC.FSTDISP = 337;
     ELSE DATADESC.FSTDISP = 3557;
     DATADESC.LSTRECNO = I;
     IF LD1 = '1' THEN DATADESC.LSTDISP = 337;
     ELSE DATADESC.LSTDISP = 3557;
     IF BLKPER = '1' THEN
     L = DATADESC.BLOCKSIZE * DATADESC.TOTALBLKS_ALLOC/6440 + .5;
     ELSE
      L = DATADESC.BLOCKSIZE * DATADESC.TOTALBLKS_ALLOC/3220+.5;
     IF L > 1 THEN PUT SKIP FILE(SYSPRINT)
     LIST('** DATA DOESNT FIT FOR 1 RECORD *  L = ',L);
     DDKEY = I:    /* KEY TO DD RECORD   */
     IF BLKPER = '1' THEN DO;
        DESC1 = DATADESC;
        FIL1 = ' ';
        REWRITE FILE(DAF) FROM(DATA_BLK1) KEY(DDKEY);
        RECTYPE(I) = 1;
        FRESPACE(I) = 6440 - 335 - DESC1.TOTALBLKS_ALLOC *
```



```
                        DESC1.BLOCKSIZE;
          LOCATION(I) = 6440 - FRESPACE(I) + 1;
          END;
       ELSE DO;
          IF LDI = '1' THEN DO:
             DESC1 = DATADESC;
             FIL1 = ' ';
             REWRITE FILE(DAF) FROM(DATA_BLK1) KEY(DDKEY);
             RECTYPE(I) = 1;
             FRESPACE(I) = 3220;
             LOCATION(I) = 3221;
             LDI = '2';
             END;
          ELSE DO;
             DESC2 = DATADESC;
             REWRITE FILE(DAF) FROM(DATA_BLK1) KEY(DDKEY);
             RECTYPE(I) = 1;
             FRESPACE(I) = 0;
             LOCATION(I) = -1;
             LDI = '1';
             END;
       END;
 UPDATE_DIR:  ;
       /* REWRITE DIR  */
       REWRITE FILE(DAF) FROM(DIRX) KEY(DKEY);
       GO TO NXT_ONE;
 EOJ:  ;
       REWRITE FILE(DAF) FROM (CONTROL) KEY(CKEY);
       PUT SKIP FILE(SYSPRINT) LIST('*DATA DESCRIPTORS ADDED TO FILE*');
       FREE DATA_BLK1;
     RCODE = 1;
       RETURN;
 ERR1:  ;
       PUT SKIP FILE(SYSPRINT) LIST('** INVALID LEVEL CODE **', ENUCHK);
       RETURN;
 ERR2:  ;
       PUT SKIP FILE(SYSPRINT) LIST
       ('MISSING CODE DATA **');
       RETURN;
 ERR3:  ;
       PUT SKIP FILE(SYSPRINT) LIST
       ('** NON-NUMERIC DATA ON INPUT - CORRECT AND RESUBMIT **');
       RETURN;
 END YESDF03;
```

3.3.8 RECOVER DIRECTORY FROM THE DATA BASE

GETDIR is used by YESM001 and the loaders (3.3) to recover directory blocks from the data base.

3.3.8.1 Linkages

None.

3.3.8.2 Interfaces

None.

3.3.8.3 Inputs

Level identification number.

3.3.8.4 Outputs

Directory block, or error indication.

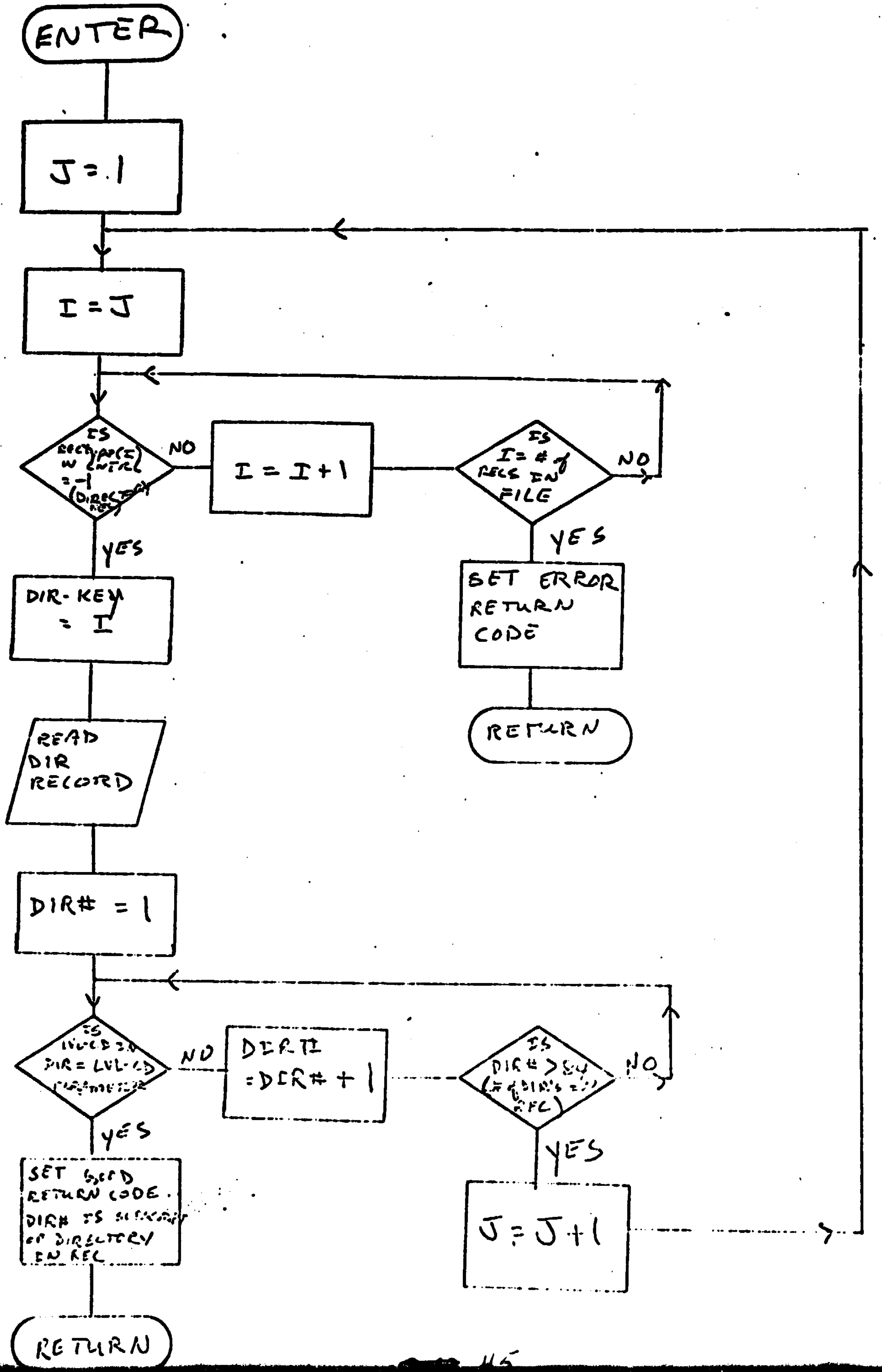
3.3.8.5 Flow Chart

Next page.

3.3.8.6 Listing

Follows flow chart.

GETDIR


ENTER
J = 1
I = J
IS RECTYPE(I) W ENTRL = -1 (DIRECTORY KEY)
NO
I = I + 1
IS I = # of RECS IN FILE
NO
YES
SET ERROR RETURN CODE
RETURN
YES
DIR-KEY = I
READ DIR RECORD
DIR# = 1
IS LVL-CD IN DIR = LVL-CD PARAMETER
NO
DIR# = DIR# + 1
IS DIR# > 64 (# of DIR's in REC)
NO
YES
J = J + 1
YES
SET GOOD RETURN CODE. DIR# IS SUBSCRIPT OF DIRECTORY IN REC
RETURN

```
RUN NO. 15        DATE 11/12/76    TIME  0910             LISTING OF MODULE GETDIR

DESCRIPTION             DATA BASE ROM

MASTER FILE             W.EOS.CCEA.LEC.LIBR
 ADDED TO MASTER        10/13/75
LAST DATE COPIED          NONE
 LAST UPDATE              NONE

 PASSWORD               GHAK
PROGRAMMER              LEC
 LANGUAGE               PLI
PROC PARAMETER          $NOJCL
```

ORIGINAL PAGE IS OF POOR QUALITY

```
   GETDIR:  PROC(DIR_CD,DIRX,CONTROL,DAF,DKEY,DIR#,RC);
        DCL DAF FILE;
        DCL (DIR#,CKEY,DKEY,RC) FIXED BIN(15);
        DCL DIR_CD FIXED BIN(31);
      DCL 1 CONTROL,
              2 FILEID CHAR(8),
              2 NUMPASS FIXED BIN(15,0),
              2 PASS(8) CHAR(8),
              2 NUMLEV FIXED BIN(15,0),
              2 LEVNAME(8) CHAR(24),
              2 NUMCODE FIXED BIN(15,0),
              2 CODE(32),
                 3 CODENUM FIXED BIN(15,0),
                 3 UNITNUM FIXED BIN(15,0),
                 3 BASE FIXED BIN(15,0),
                 3 SCALE FIXED BIN(15,0),
                 3 CODENAME CHAR(24),
                 3 UNITNAME CHAR(24),
              2 NUMONE FIXED BIN(15,0),
              2 ONE(24),
                 3 CODENUM FIXED BIN(15,0),
                 3 NUMDIRS FIXED BIN(15,0),
                 3 RECNUM FIXED BIN(15,0),
                 3 DISPLACE FIXED BIN(15,0),
                 3 NAME CHAR(24),
              2 FILERECS FIXED BIN(15,0),
              2 REC(501),
                 3 RECTYPE FIXED BIN(15,0),
                 3 FRESPACE FIXED BIN(15,0),
                 3 LOCATION FIXED BIN(15,0);
      DCL 1 DIRX,
              2 DIR (54),
              3 LEVNUM FIXED BIN(15,0),
              3 CODENUMB FIXED BIN(15,0);
              3 LAT FIXED BIN(15,0),
              3 LON FIXED BIN(15,0),
              3 DIRNAME CHAR(24),
              3 PREC FIXED BIN(15,0),
              3 PDISP FIXED BIN(15,0),
              3 BREC FIXED BIN(15,0),
              3 BDISP FIXED BIN(15,0),
              3 CREC FIXED BIN(15,0),
              3 CDISP FIXED BIN(15,0),
              3 DREC FIXED BIN(15,0),
              3 DDISP FIXED BIN(15,0),
              3 LEVCODE FIXED BIN(31,0),
              3 MODEL(4),
                 4 CROP FIXED BIN(15,0),
                 4 MREC FIXED BIN(15,0),
               4 MDISP FIXED BIN(15,0),
           2 FILLER CHAR (56);
        DCL 1 REC_BLK LIKE DIRX;
        ON CONVERSION BEGIN;
        PUT SKIP LIST(DIR_CD,DIR#,RC,DKEY,ONE(1),DIRX.DIR(DIR#));
        END;
   /*                  */
        J=1;
   LP1: ;
        DO I=J TO FILERECS;
        IF RECTYPE(I) = -1 THEN GOTO XIT1;
        END;
        RC = -1;   /*  ERROR  */
        RETURN;
   /*     */
   XIT1:  ;
        DKEY = I;
        READ FILE(DAF) INTO(REC_BLK) KEY(DKEY);
        DIRX = REC_BLK;
   LP2:  ;
        DO DIR# = 1 TO 54;
        IF DIRX.LEVCODE(DIR#) = DIR_CD THEN GOTO FOUND_DIR;
        END;
        J = J + 1;
        GOTO LP1;
   FOUND_DIR:  ;
        RC = 1;
        RETURN;
   END GETDIR;
```

3.3.9 DIRECTORY BLOCK ENTRY ROUTINE (YESDF04)

YESDF04 places directory blocks in the data base.

3.3.9.1 Linkages

None.

3.3.9.2 Interfaces

None.

3.3.9.3 Inputs

Directory definition cards.

3.3.9.4 Outputs

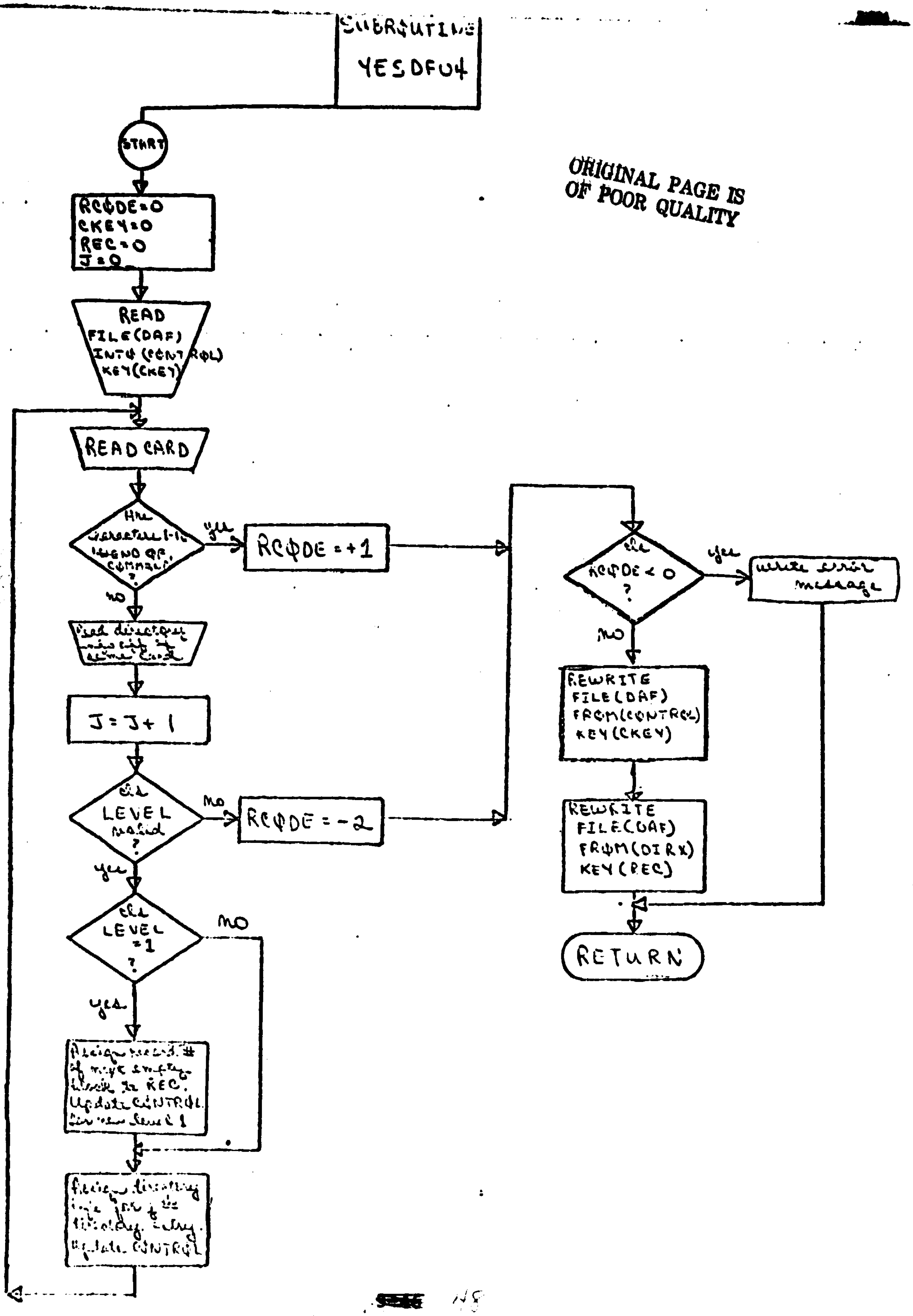
Directory definition on the data base.

3.3.9.5 Flow Chart

Next page.

3.3.9.6 Listing

Follows flow chart.



ORIGINAL PAGE IS
OF POOR QUALITY

```
RUN NO. 15        DATE  11/12/76     TIME  0910            LISTING OF MODULE YESDF04
DESCRIPTION              DATA BASE PGM

MASTER FILE              W.EDS.CCEA.LEC.LIBR
 ADDED TO MASTER         10/13/75
LAST DATE COPIED           NONE
 LAST UPDATE               NONE

 PASSWORD                HDMH
PROGRAMMER               LEC
 LANGUAGE                PLI
PROC PARAMETER           $NOJCL

 YESDF04: PROCEDURE(SYSIN,SYSPRINT,DAF,IJOB,RCODE);
       /* THIS PROGRAM IS CALLED BY YESDF01 TO DEFINE DIRECTORY ENTRIES     */
       /* THIS PROGRAM ASSUMES A 6440-BYTE DIRECTORY BLOCK CAN CONTAIN       */
       /*   84 DIRECTORY ENTRIES, EACH 76 BYTES LONG.                        */
       /*                                                                    */
      DCL 1 CONTROL,
              2 FILEID CHAR(8),
              2 NUMPASS FIXED BIN(15,0),
              2 PASS(8) CHAR(8),
              2 NUMLEV FIXED BIN(15,0),
              2 LEVNAME(8) CHAR(24),
              2 NUMCODE FIXED BIN(15,0),
              2 CODE(32),
                3 CODENUM FIXED BIN(15,0),
                3 UNITNUM FIXED BIN(15,0),
                3 BASE FIXED BIN(15,0),
                3 SCALE FIXED BIN(15,0),
                3 CODENAME CHAR(24),
                3 UNITNAME CHAR(24),
              2 NUMONE FIXED BIN(15,0),
              2 ONE(24),
                3 CODENUM FIXED BIN(15,0),
                3 NUMDIRS FIXED BIN(15,0),
                3 RECNUM FIXED BIN(15,0),
                3 DISPLACE FIXED BIN(15,0),
                3 NAME CHAR(24),
              2 FILERECS FIXED BIN(15,0),
              2 REC(501),
                3 RECTYPE FIXED BIN(15,0),
                3 FRESPACE FIXED BIN(15,0),
                3 LOCATION FIXED BIN(15,0);
      DCL 1 DIRX BASED(P),
              2 DIR (84),
              3 LEVNUM FIXED BIN(15,0),
              3 CODENUMP FIXED BIN(15,0),
              3 LAT FIXED BIN(15,0),
              3 LON FIXED BIN(15,0),
              3 DIRNAME CHAR(24),
              3 PREC FIXED BIN(15,0),
              3 PDISP FIXED BIN(15,0),
              3 BREC FIXED BIN(15,0),
              3 BDISP FIXED BIN(15,0),
              3 CREC FIXED BIN(15,0),
              3 CDISP FIXED BIN(15,0),
              3 DREC FIXED BIN(15,0),
              3 DDISP FIXED BIN(15,0),
              3 LEVCODE FIXED BIN(31,0),
              3 MODEL(4),
                4 CROP FIXED BIN(15,0),
                4 MREC FIXED BIN(15,0),
               4 MDISP FIXED BIN(15,0),
           2 FILLER CHAR (56);
      DCL  (REC,CKEY,LASTKEY) FIXED BIN(10,0);
      DCL  (I,J,K,JJ,XLEV,XCODE,XLAT,XLON,XP,XB,XC,RCODE,RSIZE,RTYPE,L,
        II) FIXED BIN(15,0);
      DCL  (XL,D) FIXED BIN(31,0);
      DCL  INSTR CHAR(80);
      DCL  XNAME CHAR(24);
      DCL  (AFLAG,BFLAG,CFLAG) BIT(1);
      DCL  (SYSIN,SYSPRINT,DAF) FILE;
      DCL  SUBSTR BUILTIN;
      DCL  BLANK CHAR(6440) BASED(P);
      DCL  P POINTER;
   PUT SKIP FILE(SYSPRINT) EDIT ('**DEFINE DIRECTORY PROGRAM**') (A);
   ON CONVERSION BEGIN;
    PUT SKIP FILE(SYSPRINT) EDIT ('***INVALID INPUT CARD***') (A);
    PUT SKIP FILE(SYSPRINT) EDIT (INSTR) (A(80));
    RCODE=-1;
    GOTO STOP;
   END;
```

```
ON ENDFILE (SYSIN) BEGIN;
  PUT SKIP FILE(SYSPRINT) EDIT
    ('***ERROR - END OF FILE SYSIN ENCOUNTERED***') (A);
  RCODE=-1;
  GOTO STOP;
END;
ALLOCATE BLANK SET(P);
BLANK=' ';
DIRX.FILLER=' ';
I.J.K.RCODE.JJ.O=0;
CKEY.REC.LASTKEY=0;
RTYPE=-1;
RSIZE=76;
READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
AFLAG='1'B;
DO WHILE(AFLAG);
  GET FILE(SYSIN) EDIT(INSTR) (COL(1).A(80));
  IF SUBSTR(INSTR.1.16)='**END OF COMMAND' THEN DO;
    AFLAG='0'B;
    RCODE=1;
  END;
  ELSE DO;
     GET STRING(INSTR) EDIT (XLEV.XCODE.XLAT.XLON.XNAME.XA.XB.XC.XL)
       (A(2).F(2.0).F(4.0).X(1).2 F(5.0).X(1).A(24).3 F(4.0).X(14).
       F(10.0));
 /* CHECK FOR INVALID INPUT CARDS */
     IF XLEV < 1 | XLEV > CONTROL.NUMLEV THEN DO;
       AFLAG='0'B;
       RCODE=-2;
       PUT SKIP FILE(SYSPRINT) EDIT
         ('***INVALID LEVEL NUMBER ON INPUT CARD***') (A);
       PUT SKIP FILE(SYSPRINT) EDIT (INSTR) (A(80));
     END;
     ELSE IF J=0 & XLEV>1 THEN DO;
       AFLAG='0'B; RCODE=-2;
       PUT SKIP FILE(SYSPRINT) EDIT
        ('***FIRST INPUT CARD ISNOT A LEVEL-ONE DIRECTORY ENTRY***') (A);
       PUT SKIP FILE(SYSPRINT) EDIT (INSTR) (A(80));
     END;
     ELSE IF J>0 & XLEV=1 THEN DO;
       AFLAG='0'B; RCODE=-2;
       PUT SKIP FILE(SYSPRINT) EDIT
        ('***ERROR - SECOND LEVEL-ONE DIRECTORY ENTRY READ***') (A);
       PUT SKIP FILE(SYSPRINT) EDIT (INSTR) (A(80));
     END;
     ELSE DO;
       IF XLEV = 1 THEN DO;
         J=1;
         BFLAG='1'B;
          /* FIND A CONTIGUOUS SET OF EMPTY BLOCKS FOR THE NEW       */
          /*    DIRECTORY ENTRIES                                     */
          /* THEN UPDATE CONTROL BLOCK INFO FOR THE FIRST OF THE      */
          /*    EMPTY BLOCKS                                          */
         DO L = 1 TO CONTROL.FILEBLCS WHILE(BFLAG);
           IF CONTROL.REC(L).RECTYPE = 0 THEN DO;
            CFLAG='1'B;
            DO K=1 TO 11 WHILE (CFLAG);
              IF CONTROL.REC(L+K).RECTYPE = ¬ 0 THEN CFLAG='0'B;
            END;
            IF CFLAG THEN DO;
             BFLAG='0'B;
              /* UPDATE CONTROL BLOCK INFO FOR A NEW LEVEL-ONE ENTRY */
             CONTROL.NUMONE.I = CONTROL.NUMONE + 1;
             CONTROL.ONE(I).CODE = XCODE;
             CONTROL.ONE(I).NAME = XNAME;
             CONTROL.ONE(I).[illegible] = 1;
             CONTROL.ONE(I).RECNUM = L;
             CONTROL.ONE(I).DISPLACE = 1;
             CONTROL.REC(L).RECTYPE = RTYPE;
             CONTROL.REC(L).FREESPACE = CONTROL.REC(L).FREESPACE - RSIZE;
             CONTROL.REC(L).LOCATION = CONTROL.REC(L).LOCATION+RSIZE+1;
              REC=L;
            END;
           END;
         END;
         IF BFLAG THEN DO;
           AFLAG='0'B; RCODE=-2;
           PUT SKIP FILE(SYSPRINT) EDIT
             ('***FEWER THAN 11 BLOCKS REMAINING ON FILE***') (A);
         END;
        END;
        ELSE DO;
          J=J+1;
          I=REC;
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
          IF J=85 THEN DO;
           /* IF INPUT CARD IS THE 85TH TO BE READ IN, THEN REWRITE  */
           /*    FILE WITH PREVIOUS 84 DIRECTORY ENTRIES IN A BLOCK   */
            REWRITE FILE(DAF) FROM(DIRX) KEY(REC);
            REC,L=REC+1;
            CONTROL.REC(L).RECTYPE=-1;
            BLANK=' ';
            J=1;
          END;
           /* UPDATE CONTROL BLOCK INFO */
          CONTROL.REC(L).FREESPACE = CONTROL.REC(L).FREESPACE - RSIZE;
          CONTROL.REC(L).LOCATION = CONTROL.REC(L).LOCATION + RSIZE;
          CONTROL.ONE(I).NUMDIRS = CONTROL.ONE(I).NUMDIRS +1;
        END;
         /* DEFINE DIRECTORY INFO */
        DIRX.DIR(J).LEVNUM = XLEV;
        DIRX.DIR(J).CODENUMB = XCODE;
        DIRX.DIR(J).LAT = XLAT;
        DIRX.DIR(J).LON = XLON;
        DIRX.DIR(J).DIRNAME = XNAME;
        DIRX.DIR(J).LEVCODE = XL;
        DO K = 1 TO 4;
          DIRX.DIR(J).MODEL(K).CPNP = 0;
          DIRX.DIR(J).MODEL(K).MREC = -1;
          DIRX.DIR(J).MODEL(K).MDISP = 1;
        END;
        IF XP=-1 THEN DO;
          DIRX.DIR(J).PREC = XP;
          DIRX.DIR(J).PDISP = 1;
        END;
        IF XP>0 THEN DO;
          D = ((XP-1)*RSIZE)+1;
 /* 6384 = 76*84, WHICH ARE THE SIZE OF ONE DIRECTORY ENTRY AND THE   */
 /*  MAXIMUM NUMBER OF ENTRIES ONE BLOCK CAN CONTAIN                   */
          II = FLOOR(D/6384);
          DIRX.DIR(J).PREC = CONTROL.ONE(I).RECNUM + II;
          DIRX.DIR(J).PDISP = D - (II*6384);
        END;
        IF XR=-1 THEN DO;
          DIRX.DIR(J).RREC = XR;
          DIRX.DIR(J).RDISP =1;
        END;
        IF XR>0 THEN DO;
          D = ((XR-1)*RSIZE)+1;
          II = FLOOR(D/6384);
          DIRX.DIR(J).RREC = CONTROL.ONE(I).RECNUM + II;
          DIRX.DIR(J).RDISP = D - (II*6384);
        END;
        IF XC=-1 THEN DO;
          DIRX.DIR(J).CREC = XC;
          DIRX.DIR(J).CDISP = 1;
        END;
        IF XC>0 THEN DO;
          D = ((XC-1)*RSIZE)+1;
          II = FLOOR(D/6384);
          DIRX.DIR(J).CREC = CONTROL.ONE(I).RECNUM + II;
          DIRX.DIR(J).CDISP = D - (II*6384);
        END;
        DIRX.DIR(J).DREC = -1;
        DIRX.DIR(J).DDISP = 1;
      END;
    END;
  END;
  IF REC>0 & XCODE>=1 THEN DO;
    REWRITE FILE(DAF) FROM(DIRX) KEY(REC);
    REWRITE FILE(DAF) FROM(CONTROL) KEY(CREC);
  END;
  PUT SKIP FILE(SYSPRINT) EDIT ('***', CONTROL.ONE(I).NUMDIRS,
    ' DIRECTORY ENTRIES WRITTEN TO FILE***') (A,F(5,0),A);
  FREE BLANK;
  STOP; RETURN;
  END YESUFU4;
```

3.3.10 SUBPROGRAM STUBS

The subprogram stubs YESDF05, YESLS01, YESUD01, YESDE02 are dummy subroutines.

3.3.10.1 Linkages

N/A.

3.3.10.2 Interfaces

N/A.

3.3.10.3 Inputs

N/A.

3.3.10.4 Outputs

The message: DUMMY CALL TO YESXX00X.

3.3.10.5 Flow Chart

N/A.

3.3.10.6 Listing

Next page.

ORIGINAL PAG
OF POOR QUALIT

```
RUN NO. 15      DATE  11/12/76     TIME  0910            LISTING OF MODULE YESDF05

DESCRIPTION             DATA BASE PGM

MASTER FILE             W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER         10/13/76
LAST DATE COPIED          NONE
LAST UPDATE               NONE

PASSWORD                KLCL
PROGRAMMER              LEC
LANGUAGE                PLI
PROC PARAMETER          $NOJCL

     YESDF05:  PROC(SYSIN,SYSPRINT);
          DCL (SYSIN,SYSPRINT) FILE;
          PUT SKIP FILE(SYSPRINT) LIST('**DUMMY CALL YESDF05**');
          RETURN;
     END YESDF05;
```

```
RUN NO. 15      DATE  11/12/76     TIME  0910            LISTING OF MODULE YESLS01

DESCRIPTION             DATA BASE PGM

MASTER FILE             W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER         10/13/76
LAST DATE COPIED          NONE
LAST UPDATE               NONE

PASSWORD                VTCF
PROGRAMMER              LEC
LANGUAGE                PLI
PROC PARAMETER          $NOJCL

     YESLS01:  PROC(SYSIN,SYSPRINT);
          DCL (SYSIN,SYSPRINT) FILE;
          PUT SKIP FILE(SYSPRINT) LIST('**DUMMY CALL TO YESLS01**');
          RETURN;
     END YESLS01;
```

```
RUN NO. 15      DATE  11/12/76     TIME  0910            LISTING OF MODULE YESUD01

DESCRIPTION             DATA BASE PGM

MASTER FILE             W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER         10/13/76
LAST DATE COPIED          NONE
LAST UPDATE               NONE

PASSWORD                BLPV
PROGRAMMER              LEC
LANGUAGE                PLI
PROC PARAMETER          $NOJCL

     YESUD01:  PROC(SYSIN,SYSPRINT);
          DCL (SYSIN,SYSPRINT) FILE;
          PUT SKIP FILE(SYSPRINT) LIST('**DUMMY CALL TO YESUD01**');
          RETURN;
     END YESUD01;
```

```
RUN NO. 15        DATE  11/12/76     TIME  0910                  LISTING OF MODULE YESDE01

DESCRIPTION            DATA BASE PGM

MASTER FILE             W.EDS.CCEA.LEC.LIBR
 ADDED TO MASTER        10/13/75
LAST DATE COPIED          NONE
 LAST UPDATE              NONE

 PASSWORD               PLBX
PROGRAMMER              LEC
 LANGUAGE               PLI
PROC PARAMETER          SNOJCL

    YESDE01:  PROC(SYSIN.SYSPRINT);
         DCL (SYSIN.SYSPRINT) FILE;
         PUT SKIP FILE(SYSPRINT) LIST('**DUMMY CALL TO YESDE01**');
         RETURN;
    END YESDE01;
```

...AL PAGE IS
OF POOR QUALITY

ORIGINAL PAGE IS
OF POOR QUALITY

3.3.11 UPDATING THE DATA BASE (UPDDATA)

UPDDATA will enter or modify data in the data base.

3.3.11.1 Linkages

None.

3.3.11.2 Interfaces

A directory must have been defined for the area to be updated.

3.3.11.3 Inputs

Update data cards.

3.3.11.4 Outputs

Updated data base.

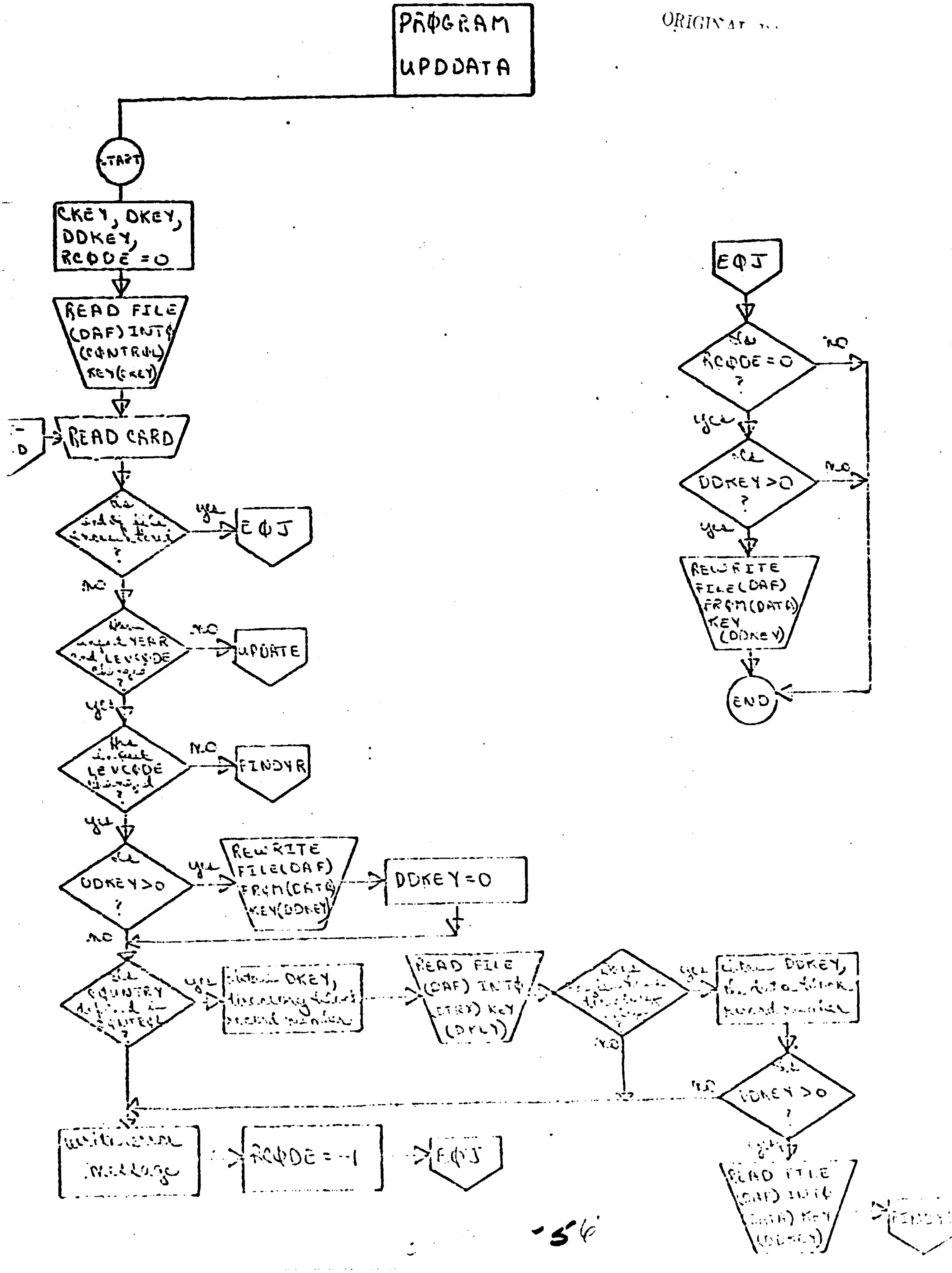
3.3.11.5 Flow Chart

Next page.

3.3.11.6 Listing

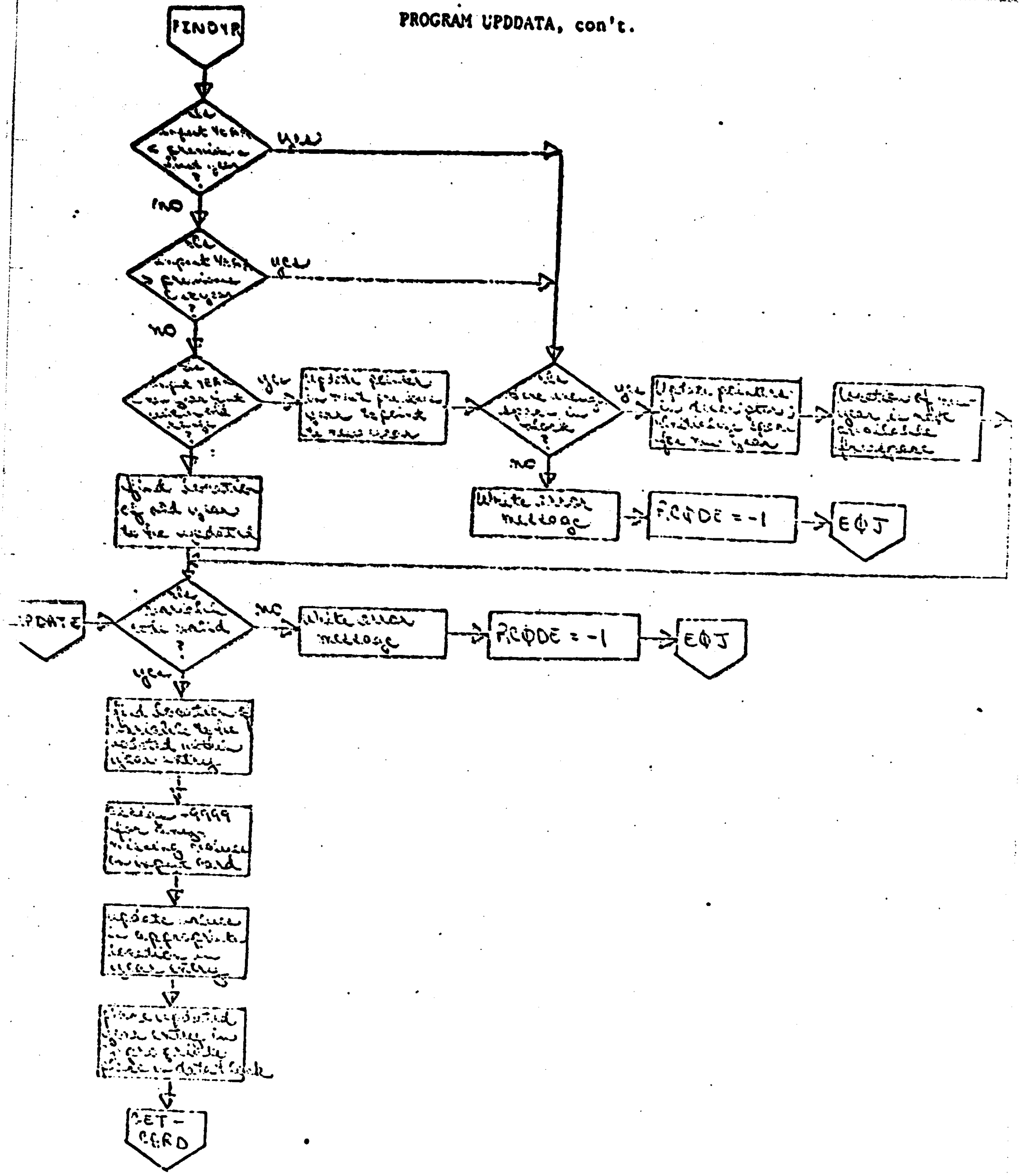
Follows flow chart.

ORIGINAL ...



PRΦGRAM UPDDATA
START
CKEY, DKEY, DDKEY, RCΦDE = 0
READ FILE (DAF) INTΦ (CΦNTRΦL) KEY(CKEY)
READ CARD
yes
EΦJ
NO
the input YEAR and LEVCΦDE changed ?
NO
UPDATE
yes
the input LEVCΦDE changed ?
NO
FINDYR
yes
Is DDKEY>0 ?
yes
REWRITE FILE(DAF) FRΦM(DATA) KEY(DDKEY)
DDKEY=0
NO
Is COUNTRY defined in CNTRY ?
yes
DKEY,
READ FILE (DAF) INTΦ (ETRY) KEY (DKEY)
yes
DDKEY,
NO
NO
Is DDKEY > 0 ?
yes
READ FILE (DAF) INTΦ (DATA) KEY (DDKEY)
FINDYR
message
RCΦDE = -1
EΦJ
EΦJ
Is RCΦDE = 0 ?
NO
yes
Is DDKEY>0 ?
NO
yes
REWRITE FILE(DAF) FRΦM(DATA) KEY (DDKEY)
END

PROGRAM UPDDATA, con't.


FINDYR
Is input YEAR < previous first year ?
yes
no
Is input YEAR > previous last year ?
yes
no
yes
Update pointer in that previous year to point to new year
Are there any spaces in block ?
yes
Update pointers in directory; available space for new year
Location of new year is next available freespace
no
Write error message
RCØDE = -1
EØJ
Find location of old year to be updated
UPDATE
Is variable code valid ?
no
Write error message
RCØDE = -1
EØJ
yes
Find location of variable to be updated within year entry
-9999
Update values in appropriate location in year entry
GET-CARD

```
   STMT LEV NT

                         /*  UNPRINTABLE CHARACTER AND APPEARS AS A BLANK             */
                         /*                                                           */
     14   1   0   DCL ZERO CHAR(128) VAR INIT((128)' ');
     15   1   0   DCL YRDATA CHAR(124) VAR;
     16   1   0   DCL METDATA(12) FIXED BIN(15,0) BASED(P1);
     17   1   0   DCL MET CHAR(24) BASED(P1);
     18   1   0   DCL YLDDATA(4) FIXED BIN(31,0) BASED(P2);
     19   1   0   DCL YLD CHAR(16) BASED(P2);
     20   1   0   DCL PNTRDATA(4) FIXED BIN(15,0) BASED(P3);
     21   1   0   DCL PNTR CHAR(8) BASED(P3);
     22   1   0   DCL AFLAG BIT(1);
     23   1   0   DCL (CKEY,DKEY,DDKEY,DNUM,DDDISP,I,J,FSTYR,LSTYR,YRSLEFT,DIFF,DSP,
                        SPACE,LENGTH,ELEMENTS,BEFORE,      NEWYR,RCODE,NXTDISP,FORMERYR,
                        FORMERDSP)
                        FIXED BIN(15,0);
     24   1   0   DCL LEVS FIXED BIN(31,0);
     25   1   0   DCL (P1,P2,P3,Q,R) POINTER;
     26   1   0   DCL Q1 POINTER;
     27   1   0   ON ENDFILE(CARDS) GOTO EOJ;
     28   1   0   CKEY,DKEY,DDKEY,DDDISP,DNUM,LEVS,NEWYR,RCODE,FSTYR,LSTYR,YRSLEFT=0;
     29   1   0   DIFF,DSP,LENGTH,ELEMENTS,BEFORE=0;
     30   1   0   READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
     31   1   0   ALLOCATE INPUT;
     32   1   0   ALLOCATE INSTR;
     33   1   0   GETCARD: GET FILE(CARDS) EDIT(INSTR) (COL(1),A(80));
     34   1   0     GET STRING(INSTR) EDIT (COUNTRY) (X(70),F(2,0));
     35   1   0     GET STRING(INSTR) EDIT(INPUT.INYR,INPUT.INCD,DEF.INLVLCD)
                        (X(3),F(4,0),F(3,0),X(60),F(10,0));
     36   1   0     IF INPUT.INCD<100 THEN GET STRING(INSTR) EDIT
                        ((INPUT.INMON(I) DO I=1 TO 12)) (X(10),12 F(5,0));
     37   1   0     IF INPUT.INCD>100 THEN GET STRING(INSTR) EDIT
                        ((DEF.INAGDTA(I) DO I=1 TO 6)) (X(10),6 F(10,0));
                         /*                                                           */
                         /* CHECK IF INPUT ENTRY AND YEAR ARE THE SAME AS THOSE ON    */
                         /*  THE PREVIOUS CARD. IF BOTH ARE EQUAL THEN GO TO SECTION*/
                         /*  WHICH CHECKS FOR NEW VARIABLE CODE.  IF ONLY YEAR IS     */
                         /*  CHANGED THEN GO TO SECTION WHICH FINDS NEW YEAR BLOCK. */
                         /*  IF BOTH ARE CHANGED AND THIS ISNOT THE FIRST INPUT CARD*/
                         /*  THEN REWRITE UPDATED DATA BLOCK AND START PROCESS OVER */
                         /*                                                           */
     38   1   0     IF DEF.INLVLCD=LEVS & INPUT.INYR=NEWYR THEN GOTO UPDATE;
     39   1   0     IF DEF.INLVLCD=LEVS THEN GOTO FINDYR;
     40   1   0     IF DDKEY>0 THEN DO;
     41   1   1       REWRITE FILE(DAF) FROM(DATA) KEY(DDKEY);
     42   1   1       PUT SKIP(2) FILE(SYSPRINT) EDIT ('*** FILE UPDATED FOR REGION',
                        LEVS) (A,F(11,0));
     43   1   1       DKEY,DDKEY=0;
     44   1   1       FREE DUMMY;
     45   1   1     END;
     46   1   0     LEVS=DEF.INLVLCD;  NEWYR=INPUT.INYR;
     48   1   0     ALLOCATE DUMMY;
     49   1   0     AFLAG='1'B;
                         /*                                                           */
                         /* FIND DIRECTORY BLOCK FOR COUNTRY DEFINED ON INPUT CARD    */
                         /*                                                           */
     50   1   0     DO I=1 TO CONTROL.NUMONE WHILE(AFLAG);
     51   1   1       IF COUNTRY=CONTROL.ONE(I).CODE1NUM THEN DO;
     52   1   2         AFLAG='0'B;
     53   1   2         DKEY=CONTROL.ONE(I).RECNUM;
     54   1   2         DNUM=CONTROL.ONE(I).NUMDIRS;
     55   1   2       END;
     56   1   1     END;
     57   1   0     IF DKEY=0 THEN DO;
     58   1   1       PUT SKIP(2) FILE(SYSPRINT) EDIT ('***NO DIRECTORY BLOCK FOUND FOR',
                        ' COUNTRY CODE ',COUNTRY      ,' ***') (A,A,F(3,0),A);
     59   1   1       RCODE=-1;
     60   1   1       GOTO EOJ;
     61   1   1     END;
     62   1   0     READ FILE(DAF) INTO(DIRX) KEY(DKEY);
     63   1   0     J=0;
     64   1   0     AFLAG='1'B;
                         /*                                                           */
                         /* FIND DATA DESCRIPTOR AND DATA BLOCK FOR INPUT ENTRY       */
                         /*                                                           */
     65   1   0     DO I=1 TO DNUM WHILE(AFLAG);
     66   1   1       J=J+1;
     67   1   1       IF J=85 THEN DO;
     68   1   2         DKEY=DKEY+1;
     69   1   2         READ FILE(DAF) INTO(DIRX) KEY(DKEY);
     70   1   2         J=1;
     71   1   2       END;
```

```
 72    1   1         IF DEF.INLVLCO=DIR(J).LEVCODE THEN DO;
 73    1   2           AFLAG='0'B;
 74    1   2           DDKEY=DIR(J).DREC;
 75    1   2           DDDISP=DIR(J).DDISP;
 76    1   2         END;
 77    1   1       END;
 78    1   0       IF DDKEY=0 THEN DO;
 79    1   1         PUT SKIP(2) FILE(SYSPRINT) EDIT ('***NO DATA BLOCK FOUND FOR',
                        ' INPUT CODE ',DEF.INLVLCD,' ***') (A,A,F(11,0),A);
 80    1   1         RCODE=-1;
 81    1   1         GOTO EOJ;
 82    1   1       END;
 83    1   0       READ FILE(DAF) INTO(DATA) KEY(DDKEY);
 84    1   0       DUMMY=SUBSTR(DATA,DDDISP,336);
 85    1   0   FINDYR:  ALLOCATE PNTR;
                 /*                                                          */
                 /* FIND FIRST AND LAST CHRONOLOGICAL YEARS DEFINED AND ALSO*/
                 /*  DISPLACEMENT OF FREESPACE WITHIN DATA BLOCK             */
                 /*                                                          */
 86    1   0   IF DESC.NUMBYRS=0 THEN SUBSTR(DATA,DESC.FSTDISP,8)=ZERO;
 87    1   0     PNTR=SUBSTR(DATA,DESC.FSTDISP,8);
 88    1   0      FSTYR=PNTRDATA(1);
 89    1   0     PNTR=SUBSTR(DATA,DESC.LSTDISP,8);
 90    1   0      LSTYR=PNTRDATA(1);
 91    1   0     DSP=(DESC.NUMBYRS*DESC.BLKSIZE)+DDDISP+336;
                 /*                                                          */
                 /* IF INPUT YEAR IS LESS THAN FIRST YEAR, THEN CHECK FOR    */
                 /*  SPACE IN DATA BLOCK, CHANGE POINTER IN DESCRIPTOR, AND  */
                 /*  INITIALIZE POINTERS AND DATA FOR NEW FIRST YEAR         */
                 /*                                                          */
 92    1   0       IF INPUT.INYR<FSTYR THEN DO;
 93    1   1         IF DESC.NUMBYRS=DESC.TOTBLKS THEN DO;
 94    1   2           PUT SKIP(2) FILE(SYSPRINT) EDIT('*** NOT ENOUGH YEAR BLOCKS',
                          ' ALLOCATED TO ACCOMMODATE AN EXTRA YEAR ***') (A,A);
 95    1   2           RCODE=-1;
 96    1   2           GOTO EOJ;
 97    1   2         END;
 98    1   1         PNTRDATA(1)=INPUT.INYR;
 99    1   1         PNTRDATA(2)=DDKEY;
100    1   1         PNTRDATA(3)=DESC.FSTDISP;
101    1   1         PNTRDATA(4)=0;
102    1   1         SUBSTR(DATA,DSP,8)=PNTR;
103    1   1         SUBSTR(DATA,DSP+8,DESC.BLKSIZE-8)=ZERO;
104    1   1         DESC.NUMBYRS=DESC.NUMBYRS+1;
105    1   1         DESC.FSTDISP=DSP;
106    1   1       END;
                 /*                                                          */
                 /* IF INPUT YEAR IS GREATER THAN LAST YEAR, THEN CHECK FOR */
                 /*  SPACE IN DATA BLOCK, CHANGE POINTER IN DESCRIPTOR, AND */
                 /*  INITIALIZE POINTERS AND DATA FOR NEW LAST YEAR         */
                 /*                                                          */
107    1   0       ELSE IF INPUT.INYR>LSTYR THEN DO;
108    1   1         IF DESC.NUMBYRS=DESC.TOTBLKS THEN DO;
109    1   2           PUT SKIP(2) FILE(SYSPRINT) EDIT('*** NOT ENOUGH YEAR BLOCKS',
                          ' ALLOCATED TO ACCOMMODATE AN EXTRA YEAR ***') (A,A);
110    1   2           RCODE=-1;
111    1   2           GOTO EOJ;
112    1   2         END;
113    1   1         PNTRDATA(1)=INPUT.INYR;
114    1   1         PNTRDATA(2)=-1;
115    1   1         PNTRDATA(3)=0;
116    1   1         PNTRDATA(4)=0;
117    1   1         SUBSTR(DATA,DSP,8)=PNTR;
118    1   1         SUBSTR(DATA,DSP+8,DESC.BLKSIZE-8)=ZERO;
119    1   1         PNTRDATA(1)=LSTYR;
120    1   1         PNTRDATA(2)=DDKEY;
121    1   1         PNTRDATA(3)=DSP;
122    1   1         PNTRDATA(4)=0;
123    1   1         IF DESC.NUMBYRS>0 THEN SUBSTR(DATA,DESC.LSTDISP,8)=PNTR;
124    1   1         DESC.NUMBYRS=DESC.NUMBYRS+1;
125    1   1         DESC.LSTDISP=DSP;
126    1   1       END;
                 /*                                                          */
                 /* IF INPUT YEAR EQUALS A PREVIOUSLY DEFINED YEAR, FIND    */
                 /*  ITS DISPLACEMENT WITHIN DATA BLOCK.  IF NOT, CHANGE    */
                 /*  NEXT YEAR POINTER IN APPROPRIATE YEAR BLOCK AND        */
                 /*  INITIALIZE POINTERS AND DATA FOR NEW YEAR              /*
                 /*                                                          */
127    1   0       ELSE DO;
128    1   1         AFLAG='1'B;
129    1   1         NXTDISP=DESC.FSTDISP;
130    1   1         DO I=1 TO DESC.NUMBYRS WHILE(AFLAG);
131    1   2           PNTR=SUBSTR(DATA,NXTDISP,8);
132    1   2           IF INPUT.INYR=PNTRDATA(1) THEN DO;
133    1   3             DSP=NXTDISP;
134    1   3             AFLAG='0'B;
135    1   3           END;
```

```
 136   1  2            ELSE IF INPUT.INYR<PNTRDATA(1) THEN DO;
 137   1  3               PNTRDATA(1)=FORMERYR;
 138   1  3               PNTRDATA(2)=DDKEY;
 139   1  3               PNTRDATA(3)=DSP;
 140   1  3               PNTRDATA(4)=0;
 141   1  3               SUBSTR(DATA,FORMERDSP,8)=PNTR;
 142   1  3               SUBSTR(DATA,DSP+8,DESC.BLKSIZE-8)=ZERO;
 143   1  3               PNTRDATA(1)=INPUT.INYR;
 144   1  3               PNTRDATA(2)=DDKEY;
 145   1  3               PNTRDATA(3)=NXTDISP;
 146   1  3               PNTRDATA(4)=0;
 147   1  3               SUBSTR(DATA,DSP,8)=PNTR;
 148   1  3               DESC.NUMBYRS=DESC.NUMBYRS+1;
 149   1  3               AFLAG='0'B;
 150   1  3            END;
 151   1  2            ELSE DO;
 152   1  3               FORMERYR=PNTRDATA(1);
 153   1  3               FORMERDSP=NXTDISP;
 154   1  3               NXTDISP=PNTRDATA(3);
 155   1  3            END;
 156   1  2         END;
 157   1  1      END;
 158   1  0      SUBSTR(DATA,DDDISP,336)=DUMMY;
 159   1  0      YRDATA=ZERO;
 160   1  0      YRDATA=SUBSTR(DATA,DSP,DESC.BLKSIZE);
 161   1  0    UPDATE: SPACE=1;  LENGTH=0;
 163   1  0      AFLAG='1'B;
                  /*                                                          */
                  /* FIND LOCATION AND LENGTH OF FIELD AND NUMBER OF          */
                  /*   SUBELEMENTS FOR VARIABLE CODE DEFINED ON INPUT CARD    */
                  /*                                                          */
 164   1  0      DO I=1 TO DESC.NUMBCODE WHILE(AFLAG);
 165   1  1         IF INPUT.INCD=DESC.DCODE(I).CODENUMB THEN DO;
 166   1  2            LENGTH=DESC.DCODE(I).NUMSELEM*DESC.DCODE(I).ELEMSIZE;
 167   1  2            ELEMENTS=DESC.DCODE(I).NUMSELEM;
 168   1  2            BEFORE=SPACE;
 169   1  2            AFLAG='0'B;
 170   1  2         END;
 171   1  1         SPACE=SPACE+(DESC.DCODE(I).NUMSELEM*DESC.DCODE(I).ELEMSIZE);
 172   1  1      END;
 173   1  0      IF AFLAG THEN DO;
 174   1  1         PUT SKIP(2) FILE(SYSPRINT) EDIT ('***INVALID VARIABLE CODE',
                      INPUT.INCD,' ENCOUNTERED***') (A,F(5,0),A);
 175   1  1         RCODE=-1;
 176   1  1         GOTO EOJ;
 177   1  1      END;
 178   1  0      IF INPUT.INCD<100 THEN DO;
                  /*                                                          */
                  /* UPDATE INFORMATION IF VARIABLE IS METEOROLOGICAL         */
                  /*                                                          */
 179   1  1         ALLOCATE MET;
 180   1  1         DO I=1 TO ELEMENTS;
 181   1  2            IF DEFA.ACHAR(I)='     ' THEN INPUT.INMON(I)=-9999;
 182   1  2            METDATA(I)=INPUT.INMON(I);
 183   1  2         END;
 184   1  1         SUBSTR(YRDATA,BEFORE,LENGTH)=SUBSTR(MET,1,LENGTH);
 185   1  1         FREE MET;
 186   1  1      END;
 187   1  0      IF INPUT.INCD>100 THEN DO;
                  /*                                                          */
                  /* UPDATE INFORMATION IF VARIABLE IS YIELD-TYPE             */
                  /*                                                          */
 188   1  1         ALLOCATE YLD;
 189   1  1         DO I=1 TO ELEMENTS;
 190   1  2            IF DEFX.ACHARX(I)='            ' THEN DEF.INAGDTA(I)=-9999;
 191   1  2            YLDDATA(I)=DEF.INAGDTA(I);
 192   1  2         END;
 193   1  1         SUBSTR(YRDATA,BEFORE,LENGTH)=SUBSTR(YLD,1,LENGTH);
 194   1  1         FREE YLD;
 195   1  1      END;
                  /*                                                          */
                  /* PLACE CHANGES BACK IN DATA BLOCK AND LOOP FOR NEW CARD   */
                  /*                                                          */
 196   1  0      SUBSTR(DATA,DSP,DESC.BLKSIZE)=YRDATA;
 197   1  0    GOTO GETCARD;
 198   1  0    EOJ: IF RCODE=0 THEN DO;
 199   1  1            IF DDKEY>0 THEN REWRITE FILE(DAF) FROM(DATA) KEY(DDKEY);
 200   1  1            PUT SKIP(2) FILE(SYSPRINT) EDIT ('**END OF DATA UPDATE**') (A);
 201   1  1         END;
 202   1  0    FREE INSTR;
 203   1  0    FREE INPUT;
 204   1  0    FREE DUMMY;
 205   1  0    END UPDDATA;
```

3.3.12 INITIAL DATA LOADERS

The four programs USA, USSR, CANADA, and AUSARG are provided, one each for the USA, USSR, and Canada and a common loader for Australia/Argentina, to initially load the data base.

3.3.12.1 Linkages

All loaders call GETDIR.

3.3.12.2 Interfaces

Directory entries must exist for all data entered.

3.3.12.3 Inputs

Data cards.

3.3.12.4 Outputs

Initial data load of data base.

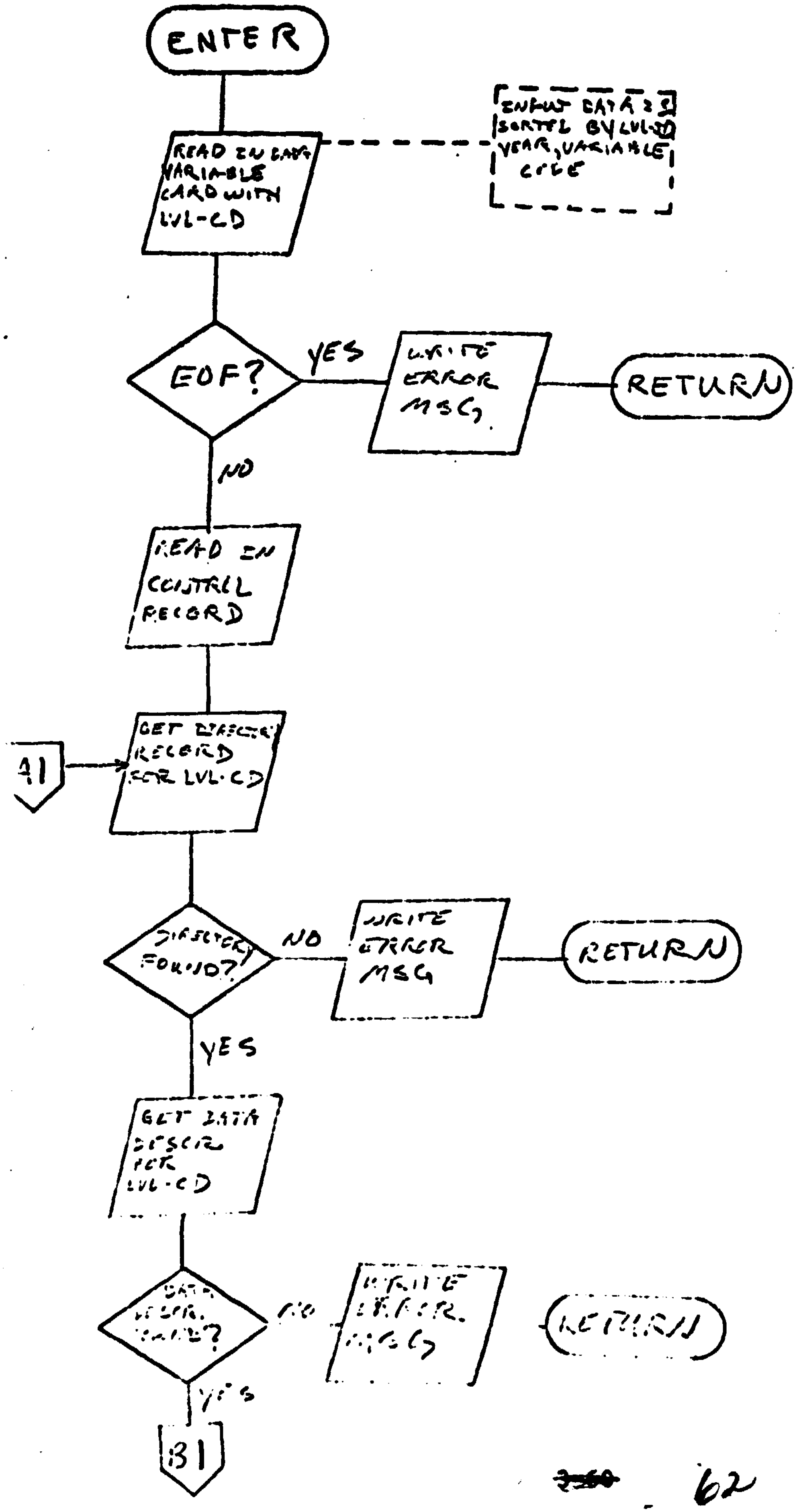
3.3.12.5 Flow Chart

A common flow chart is on the next page.

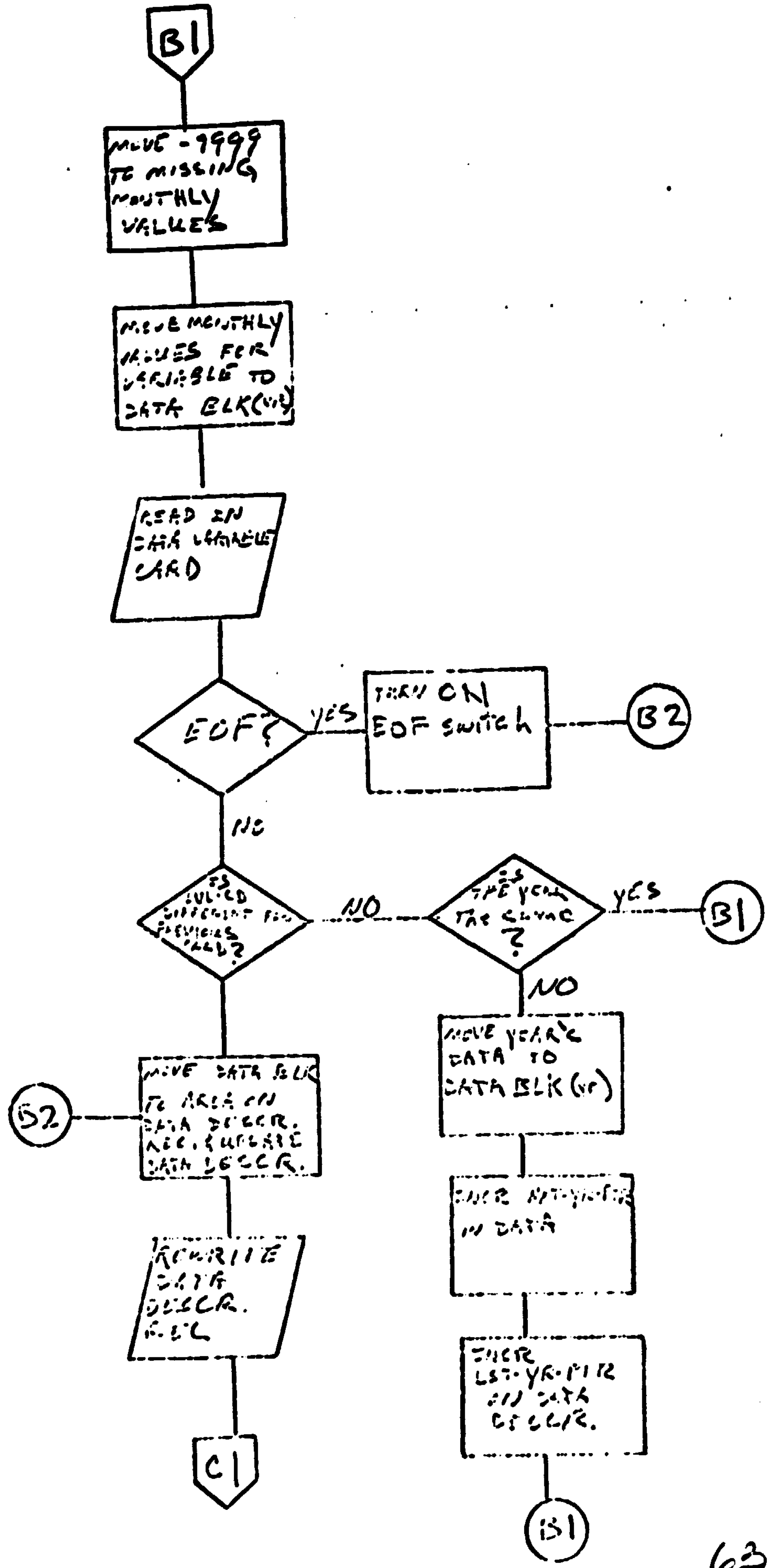
3.3.12.6 Listing

Follows flow chart.

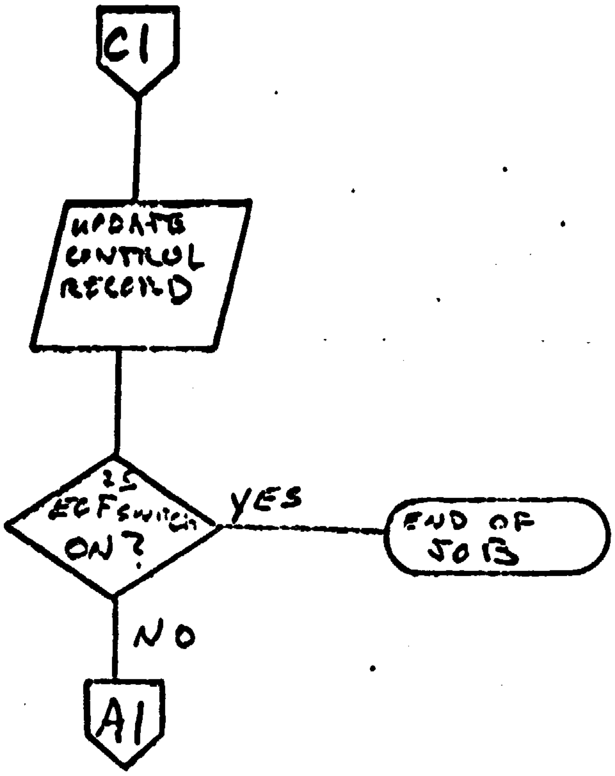
LOAD MET DATA



LOAD MET DATA, con't.


B1
MOVE -9999 TO MISSING MONTHLY VALUES
MOVE MONTHLY VALUES FOR VARIABLE TO DATA BLK(YR)
READ IN DATA VARIABLE CARD
EOF?
YES
TURN ON EOF switch
B2
NO
IS VAR-CD DIFFERENT FROM PREVIOUS VALU?
NO
IS THE YEAR THE SAME ?
YES
B1
NO
MOVE YEAR'S DATA TO DATA BLK (YR)
B2
MOVE DATA BLK TO AREA IN DATA DESCR. REC. & UPDATE DATA DESCR.
INCR NXT-YR-PTR IN DATA
REWRITE DATA DESCR. REC
INCR LST-YR-PTR IN DATA DESCR.
C1
B1

LOAD MET DATA, con't,



```
RUN NO. 19    DATE 11/12/75    TIME 0920                LISTING OF MODULE [illegible]

DESCRIPTION              DATA BASE PGM

MASTER FILE              ..EDS.CCEA.LEC..TBA
[illegible]              [illegible]
LAST DATE COPIED         NONE
LAST UPDATE              NONE

PASSWORD                 [illegible]
PROGRAMMER               LEC
LANGUAGE                 [illegible]
PROC PARAMETER           ENDJCL

  [illegible]
     DCL 1 CONTROL,
             2 [illegible]
             2 NUMBASE FIXED BIN(15,0),
             2 [illegible]
             2 NUMLEV FIXED BIN(15,0),
             2 [illegible]
             2 NUMCODE FIXED BIN(15,0),
             2 [illegible]
               3 CODENUM FIXED BIN(15,0),
               3 [illegible]
               3 BASE FIXED BIN(15,0),
               3 [illegible]
               3 [illegible] CHAR(24),
               3 [illegible]
             2 NUMONE FIXED BIN(15,0),
             2 [illegible]
               3 [illegible] FIXED BIN(15,0),
               3 [illegible]
               3 RECNUM FIXED BIN(15,0),
               3 [illegible]
               3 NAME CHAR(24),
             2 [illegible]
             2 REC(40),
               3 [illegible]
               3 FREESPACE FIXED BIN(15,0),
               3 [illegible]
     DCL 1 DISK,
             2 [illegible]
               3 LEVNUM FIXED BIN(15,0),
               3 [illegible]
               3 LAT FIXED BIN(15,0),
               3 [illegible]
               3 [illegible] CHAR(24),
               3 [illegible]
               3 [illegible] FIXED BIN(15,0),
               3 [illegible]
               3 [illegible] FIXED BIN(15,0),
               3 [illegible]
               3 [illegible] FIXED BIN(15,0),
               3 [illegible]
               3 [illegible] FIXED BIN(15,0),
               3 [illegible]
               3 [illegible](4),
                 4 [illegible]
                 4 [illegible] FIXED BIN(15,0),
                 4 [illegible]
          2 FILLER CHAR( [illegible]);
       DCL 1 [illegible],
              2  TYPE         PIC '(7)9',
              2  [illegible]  PIC [illegible],
              2  [illegible](12)  PIC 'SSSS9',
              2  [illegible],
                 3  [illegible]   PIC '99',
                 3  [illegible]   PIC [illegible],
                 3  [illegible]   PIC '99',
                 3  [illegible]   PIC '99',
                 3  [illegible]   PIC '99';
       DCL 1 [illegible],
              2  FIL1             CHAR(10),
              2  [illegible]      PIC [illegible],
              2  [illegible]      PIC '(10)9';
       DCL 1 [illegible],
              2  FIL1             CHAR(10),
              2  [illegible]      CHAR([illegible]),
              2  FIL2             CHAR(10);
       DCL 1 [illegible],
              2  FIL1             CHAR(10),
              2  [illegible]      CHAR([illegible]),
              2  FIL2             CHAR(10);
```

```
DCL 1 RUSSIA_DATA BASED(P),
        2  RUS_YR              FIXED BIN(15),
        2  RUS_NXT_YR          FIXED BIN(15),
        2  RUS_NXT_DISP        FIXED BIN(15),
        2  RESERV2              FIXED BIN(15),
        2  RUS_TEMP(12)        FIXED BIN(15),
        2  RUS_PRECIP(12)      FIXED BIN(15),
        2  RUS_HARV(4)        FIXED BIN(31),
        2  RUS_PROD(4)         FIXED BIN(31);
DCL DATA_OUT CHAR(88) BASED(P);
ALLOCATE RUSSIA_DATA;
RUSSIA_DATA.RESERV2 = 0;
ALLOCATE INPUT_DATA;
DCL 1 DATA_BLK,
         2 DATADESC,
         3 ID               FIXED BIN(31,0),
         3 WMO              FIXED BIN(31,0),
         3 LATI             FIXED BIN(15,0),
         3 LONGI            FIXED BIN(15,0),
         3 ELEV             FIXED BIN(15,0),
         3 TOTALBLKS_ALLOC  FIXED BIN(15,0),
         3 NUMBYRS_USED     FIXED BIN(15,0),
         3 BLOCKSIZE        FIXED BIN(15,0),
         3 FSTRECNO         FIXED BIN(15,0),
         3 FSTDISP          FIXED BIN(15,0),
         3 LSTRECNO         FIXED BIN(15,0),
         3 LSTDISP          FIXED BIN(15,0),
         3 RESERVED         CHAR(18),
         3 NUMSCODE         FIXED BIN(15,0),
         3 DCODE(12),
           4 CODENUME          FIXED BIN(15,0),
           4 NUMSELEM          FIXED BIN(15,0),
           4 ELEMSIZE          FIXED BIN(15,0),
           4 NUMSCODE          FIXED BIN(15,0),
           4 SUBCODE(8)        FIXED BIN(15,0),
       2  YR_BLK(32)   CHAR(88),
       2  FIL          CHAR(55) INIT(' '),
       2  DATADES2  LIKE  DATADESC,
       2  YR_BLK2(32) CHAR(88),
       2  FIL2 CHAR(68) INIT(' ');
DCL HOLD_LVLCD PIC'(10)9';
DCL HOLD_YR PIC '(7)9';
DCL DIR_CD FIXED BIN(31);
DCL FILDATA CHAR(88) BASED(P1);
DCL 1 DUMRUS BASED(P1),
        2  FILYR FIXED BIN(15),
        2   FIL_NXT_YR FIXED BIN(15),
        2   FIL_NXT_DISP FIXED BIN(15),
        2   FIL_RES2 FIXED BIN(15),
        2   FIL_TEMP(12)  FIXED BIN(15),
        2   FIL_PRECIP(12) FIXED BIN(15),
        2   FIL_HARV(4) FIXED BIN(31),
        2   FIL_PROD(4) FIXED BIN(31);
ALLOCATE FILDATA;
FIL_RESV = 0;
FIL_TEMP = -9999;
FIL_PRECIP = -9999;
FIL_HARV = 0;
FIL_PROD = 0;
DCL (DKEY,CKEY,DDKEY,DIR#,RC,YRCTR) FIXED BIN(15);
CKEY=0;
DCL STRTYR FIXED BIN(15);
STRTYR = 1956;
YRCTR=0;
RESERV2 = 0;
RUS_TEMP = -9999;
RUS_PRECIP = -9999;
RUS_HARV = 0;
RUS_PROD = 0;
DCL EOF_SW CHAR(1) INIT('0');
DCL GETDIR EXTERNAL ENTRY;
DCL CARDS FILE RECORD INPUT;
DCL DAF FILE RECORD DIRECT UPDATE KEYED ENV(REGIONAL(1));
ON CONVERSION GO TO TERM_ERR;
ON ENDFILE(CARDS) BEGIN;
   EOF_SW = '1';
   GO TO NEW_STRATA;
   END;
FRST_RD:  ;
     READ FILE(CARDS) INTO(INPUT_DATA);
      INSTRA = '00';
    INYR = TRANSLATE(INYR,'0',' ');
    INCD = TRANSLATE(INCD,'0',' ');
    INLVLCD = TRANSLATE(INLVLCD,'0',' ');
     HOLD_YR = INYR;
```

```
          HOLD_LVLCD = INLVLCD;
          DIR_CD = HOLD_LVLCD;
          READ FILE(DAFT INTO(CONTROL) KEY(CKEY);
          CALL GETDIR(DIR_CD,DIRX,CONTROL,DAF,DKEY,DIR#,RC);
          DDKEY = DREC(DIR#);
          READ FILE(DAF) INTO(DATA_BLK) KEY(DDKEY);
          GO TO CHK_CDS;
      GET_DATA:   ;
          READ FILE(CARDS) INTO(INPUT_DATA);
          INSTRA = '00';
         INYR = TRANSLATE(INYR,'0',' ');
         INCD = TRANSLATE(INCD,'0',' ');
         INLVLCD = TRANSLATE(INLVLCD,'0',' ');
          IF INLVLCD ¬= HOLD_LVLCD THEN GO TO NEW_STRATA;
          IF INYR < HOLD_YR THEN GO TO SEQ_ERR;
          IF INYR > HOLD_YR THEN GO TO PUT_DATA;
      CHK_CDS:   ;
          IF INCD =  5 THEN GO TO MV_RUS_PRECIP;
          IF INCD = 35 THEN GO TO MV_RUS_T;
          IF INCD = 101 THEN GO TO MV_RUS_HARV;
          IF INCD = 103 THEN GO TO MV_RUS_PROD;
          GO TO ERRNCD;
      MV_RUS_PRECIP:   ;
          DO N = 1 TO 12;
            IF ACHAR(N) = '        ' THEN INMON(N) = -9999;
            RUS_PRECIP(N) = INMON(N);
            END;
          GO TO GET_DATA;
      MV_RUS_T:   ;
          DO N = 1 TO 12;
            IF ACHAR(N) = '        ' THEN INMON(N) = -9999;
            RUS_TEMP(N) = INMON(N);
            END;
          GO TO GET_DATA;
      MV_RUS_HARV:   ;
          IF ACHARX(1) = '               ' THEN INAGDTA(1) = -9999;
          IF ACHARX(2) = '               ' THEN INAGDTA(2) = -9999;
          IF INREG = 01 THEN DO;
            RUS_HARV(1) = INAGDTA(1);
            END;
          IF INREG = 02 THEN DO;
            RUS_HARV(1) = INAGDTA(1);
            RUS_HARV(2) = INAGDTA(2);
            END;
          IF INREG = 03 THEN DO;
            RUS_HARV(1) = INAGDTA(1);
            END;
          GO TO GET_DATA;
      MV_RUS_PROD:   ;
          IF ACHARX(1) = '               ' THEN INAGDTA(1) = -9999;
          IF ACHARX(2) = '               ' THEN INAGDTA(2) = -9999;
          IF INREG = 01 THEN DO;
            RUS_PROD(1) = INAGDTA(1);
            END;
          IF INREG = 02 THEN DO;
            RUS_PROD(1) = INAGDTA(1);
            RUS_PROD(2) = INAGDTA(2);
            END;
          IF INREG = 03 THEN DO;
            RUS_PROD(1) = INAGDTA(1);
            END;
          GO TO GET_DATA;
      PUT_DATA:   ;
          IF YRCTR > 22 THEN GO TO NO_RM_RUS;
          YRCTR = YRCTR + 1;
          STRTYR = STRTYR + 1;
          RUS_YR = HOLD_YR;
          IF DDISP(DIR#) = 1 THEN DO;
            DATADESC.NUMBYRS_USED = YRCTR;
            RUS_NXT_YR = DATADESC.LSTRECNO;
            RUS_NXT_DISP = DATADESC.LSTDISP + 88;
            IF STRTYR = RUS_YR THEN DO;       END;
              ELSE DO;
                FILYR = STRTYR;
                FIL_NXT_YR = RUS_NXT_YR;
                FIL_NXT_DISP = RUS_NXT_DISP;
                YR_BLK(YRCTR) = FILDATA;
                DATADESC.LSTDISP = DATADESC.LSTDISP + 88;
                GO TO PUT_DATA;
                END;
            YR_BLK(YRCTR) = DATA_OUT;
            DATADESC.LSTDISP = DATADESC.LSTDISP + 88;
            END;
          ELSE DO;
            DATADES2.NUMBYRS_USED = YRCTR;
            RUS_NXT_YR = DATADES2.LSTRECNO;
            RUS_NXT_DISP = DATADES2.LSTDISP + 88;
            IF STRTYR = RUS_YR THEN DO;      END;
```

```
                ELSE DO;
                  FILYR = STRTYR;
                  FIL_NXT_YR = RUS_NXT_YR;
                  FIL_NXT_DISP = RUS_NXT_DISP;
                  YR_BLK2(YRCTR) = FILDATA;
                  DATADES2.LSTDISP = DATADES2.LSTDISP + 88;
                  GO TO PUT_DATA;
                  END;
              YR_BLK2(YRCTR) = DATA_OUT;
              DATADES2.LSTDISP = DATADES2.LSTDISP + 88;
              END;
           HOLD_YR = INYR;
           RUS_TEMP = -9999;
           RUS_PRECIP = -9999;
           RUS_HARV = 0;
           RUS_PROD = 0;
           GO TO CHK_CDS;
       TERM_ERR:  ;
           PUT SKIP LIST('**CONVERSION ERROR **',INPUT_DATA);
           GO TO EOJ;
       SEQ_ERR:  ;
           PUT SKIP LIST('** DATA NOT IN RIGHT SEQUENCE **',INPUT_DATA);
           GO TO EOJ;
       NO_RM_RUS:  ;
           PUT SKIP LIST('** NOT ENOUGH DISK SPACE ALLOCATED FOR RUSSIA **');
           PUT SKIP LIST(INPUT_DATA);
           PUT SKIP DATA(HOLD_YR,HOLD_LVLCD);
           GO TO EOJ;
       ERRNCD:  ;
           PUT SKIP LIST('** INVALID CODE **',INPUT_DATA);
           GO TO EOJ;
       NEW_STRATA:  ;
           IF YRCTR > 22 THEN GO TO NO_RM_RUS;
           YRCTR = YRCTR + 1;
           RUS_YR = HOLD_YR;
           RUS_NXT_YR = -1;
           RUS_NXT_DISP = 0;
           IF DDISP(DIR#) = 1 THEN DO;
             DATADESC.NUMBYRS_USED = YRCTR;
             YR_BLK(YRCTR) = DATA_OUT;
             END;
           ELSE DO;
             DATADES2.NUMBYRS_USED = YRCTR;
             YR_BLK2(YRCTR) = DATA_OUT;
             END;
           HOLD_YR = INYR;
           HOLD_LVLCD = INLVLCD;
           DIR_CD = HOLD_LVLCD;
           REWRITE FILE(DAF) FROM(DATA_BLK) KEY(DDKEY);
           IF EOF_SW = '1' THEN GO TO EOJ;
           CALL GETDIR(DIR_CD,DIRK,CONTROL,DAF,DKEY,DIR#,RC);
           DDKEY = DREC(DIR#);
           READ FILE(DAF) INTO(DATA_BLK) KEY(DDKEY);
           YRCTR = 0;
           STRTYR = 1956;
           RUS_TEMP = -9999;
           RUS_PRECIP = -9999;
           RUS_HARV = 0;
           RUS_PROD = 0;
           GO TO CHK_CDS;
       EOJ:  ;
           FREE RUSSIA_DATA;
           FREE INPUT_DATA;
           FREE FILDATA;
           END RUSSIA;
```

```
RUN NO. 15        DATE  11/12/76     TIME  0910               LISTING OF MODULE AUSARG

DESCRIPTION            LOADS AUS&ARG DATA TO DB

MASTER FILE            W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER        11/09/76
LAST DATE COPIED         NONE
LAST UPDATE              NONE

PASSWORD               W7GZ
PROGRAMMER             LEC
LANGUAGE               PLI
PROC PARAMETER         $NOJCL

  //DCOOK JOB ('001000HEIHEA   '.'COLUM'),COOK,REGION=200K,TIME=1
  //   EXEC SORTD,REGION=200K
  //SORTWK01 DD UNIT=SYSDA,SPACE=(6000,30,,CONTIG)
  //SORTWK02 DD UNIT=SYSDA,SPACE=(6000,80,,CONTIG)
  //SORTWK03 DD UNIT=SYSDA,SPACE=(6000,80,,CONTIG)
  //SORTWK04 DD UNIT=SYSDA,SPACE=(6000,80,,CONTIG)
  //SORTWK05 DD UNIT=SYSDA,SPACE=(6000,80,,CONTIG)
  //SORTWK06 DD UNIT=SYSDA,SPACE=(6000,80,,CONTIG)
  //SORTOUT DD DSN=&&CARDS,DISP=(,PASS),UNIT=SYSDA,SPACE=(1680,(300,30)),
  //   DCB=(RECFM=FB,LRECL=80,BLKSIZE=1680)
  //SORTIN DD DSN=W.EDS.CCEA.MET.AUSDATA,DISP=OLD
  //SYSIN DD *
     SORT FIELDS=(72,9,CH,A,1,10,CH,A),SIZE=E6000
     RECORD TYPE=F,LENGTH=(80)
     END
  /*
  //   EXEC NPLIFCLG
  //PLIL.SYSIN DD *
   ASAR: PROC OPTIONS(MAIN);
     DCL 1 CONTROL,
             2 FILEID CHAR(8),
             2 NUMPASS FIXED BIN(15,0),
             2 PASS(8) CHAR(8),
             2 NUMLEV FIXED BIN(15,0),
             2 LEVNAME(8) CHAR(24),
             2 NUMCODE FIXED BIN(15,0),
             2 CODE(32),
               3 CODENUM FIXED BIN(15,0),
               3 UNITNUM FIXED BIN(15,0),
               3 BASE FIXED BIN(15,0),
               3 SCALE FIXED BIN(15,0),
               3 CODENAME CHAR(24),
               3 UNITNAME CHAR(24),
             2 NUMONE FIXED BIN(15,0),
             2 ONE(24),
               3 CODE1NUM FIXED BIN(15,0),
               3 NUMDIPS FIXED BIN(15,0),
               3 RECNUM FIXED BIN(15,0),
               3 DISPLACE FIXED BIN(15,0),
               3 NAME CHAR(24),
             2 FILERECS FIXED BIN(15,0),
             2 REC(601),
               3 RECTYPE FIXED BIN(15,0),
               3 FRESPACE FIXED BIN(15,0),
               3 LOCATION FIXED BIN(15,0);
     DCL 1 DIRX,
             2 DIR (84),
             3 LEVNUM FIXED BIN(15,0),
             3 CODENUMB FIXED BIN(15,0),
             3 LAT FIXED BIN(15,0),
             3 LON FIXED BIN(15,0),
             3 DIRNAME CHAR(24),
             3 PREC FIXED BIN(15,0),
             3 PDISP FIXED BIN(15,0),
             3 BREC FIXED BIN(15,0),
             3 BDISP FIXED BIN(15,0),
             3 CREC FIXED BIN(15,0),
             3 CDISP FIXED BIN(15,0),
             3 DREC FIXED BIN(15,0),
             3 DDISP FIXED BIN(15,0),
             3 LEVCODE FIXED BIN(31,0),
             3 MODEL(4),
               4 CROP FIXED BIN(15,0),
               4 MREC FIXED BIN(15,0),
               4 MDISP FIXED BIN(15,0),
         2 FILLER CHAR (56);
    DCL 1 INPUT_DATA BASED(Q),
             2  INYR          PIC '(7)9',
             2  INCO          PIC '999',
             2  INMON(12)     PIC 'SSSS9',
             2  INLVLID,
               3  INCTRY      PIC '99',
               3  INREG       PIC '99',
               3  INZONE    PIC '99',
               3  INSTRA    PIC '99'.
               3  INSBSTRA PIC '99';
DCL 1 DEF_DATA BASED(Q).
        2  FIL1          CHAR(10).
        2  INAGDTA(5) PIC 'SSSSSSSSS9'.
        2  INLVLCD PIC '(10)9';
DCL 1 DEFA_DATA BASED(Q).
        2  FIL1          CHAR(10).
        2  ACHAR(12)     CHAR(5).
        2  FIL2          CHAR(10);
DCL 1 DEFX_DATA BASED(Q).
        2  FIL1          CHAR(10).
        2  ACHARX(6)     CHAR(10).
        2  FIL2          CHAR(10):
DCL 1 ASAR_DATA BASED(P),
        2  ASAR_YR               FIXED BIN(15).
        2. ASAR_NXT_YR           FIXED BIN(15).
        2  ASAR_NXT_DISP         FIXED BIN(15),
        2  RESERV2(17)           FIXED BIN(15).
        2  ASAR_TEMP(12)         FIXED BIN(15).
        2  ASAR_PRECIP(12)       FIXED BIN(15).
        2  ASAR_ZINDEX(12)       FIXED BIN(15).
        2  ASAR_HARV(2)        FIXED BIN(31).
        2  ASAR_PROD(2)          FIXED BIN(31);
DCL DATA_OUT CHAR(128) BASED(P);
ALLOCATE ASAR_DATA;
ASAR_DATA.RESERV2 = 0;
ALLOCATE INPUT_DATA;
DCL 1 DATA_BLK,
        2 DATADESC.
           3 ID                 FIXED BIN(31.0).
           3 WMO                FIXED BIN(31.0).
           3 LATI               FIXED BIN(15.0),
           3 LONGI              FIXED BIN(15.0).
           3 ELEV               FIXED BIN(15.0),
           3 TOTALBLKS_ALLOC FIXED BIN(15.0).
           3 NUMBYRS_USED       FIXED BIN(15,0),
           3 BLOCKSIZE          FIXED BIN(15.0).
           3 FSTRECNO           FIXED BIN(15.0).
           3 FSTDISP            FIXED BIN(15.0).
           3 LSTRECNO           FIXED BIN(15,0),
           3 LSTDISP            FIXED BIN(15.0).
           3 RESERVED           CHAR(18),
           3 NUMBCODE           FIXED BIN(15.0).
           3 DCODE(12),
              4 CODENUMB           FIXED BIN(15.0).
              4 NUMSELEM           FIXED BIN(15.0).
              4 ELEMSIZE           FIXED BIN(15.0).
              4 NUMSCODE           FIXED BIN(15,0).
              4 SUBCODE(8)         FIXED BIN(15,0).
        2  YR_BLK(47)   CHAR(128).
        2  FIL          CHAR(88) INIT(' ');
DCL HOLD_LVLCD PIC'(10)9';
DCL HOLD_YR PIC '(7)9';
DCL BIT_TO FIXED BIN(31):
DCL FILDATA CHAR(128) BASED(P1);
DCL 1 DUMASAR BASED(P1).
        2  FILYR FIXED BIN(15).
        2   FIL_NXT_YR FIXED BIN(15).
        2   FIL_NXT_DISP FIXED BIN(15).
        2   FIL_RESV(17) FIXED BIN(15).
        2   FIL_TEMP(12)  FIXED BIN(15).
        2   FIL_PRECIP(12) FIXED BIN(15).
        2   FIL_ZINDEX(12) FIXED BIN(15).
        2   FIL_HARV(2) FIXED BIN(31).
        2   FIL_PROD(2) FIXED BIN(31);
ALLOCATE FILDATA;
FIL_RESV = 0:
FIL_TEMP = -9999:
FIL_PRECIP = -9999;
   FIL_ZINDEX = -9999:
FIL_HARV = 0;
FIL_PROD = 0:
DCL (DKEY.CKEY.DDKEY.DIR#.RC.YRCTR) FIXED BIN(15):
CKEY=0;
DCL STRTYR FIXED BIN(15);
STRTYR = 1939:
YRCTR=0;
RESERV2 = 0;
ASAR_TEMP = -9999;
ASAR_PRECIP = -9999:
     ASAR_ZINDEX = -9999;
     ASAR_HARV = 0;
     ASAR_PROD = 0;
     DCL EOF_SW CHAR(1) INIT('0');
     DCL GETDIR EXTERNAL ENTRY;
     DCL CARDS FILE RECORD INPUT;
     DCL DAF FILE RECORD DIRECT UPDATE KEYED ENV(REGIONAL(1));
     ON CONVERSION GO TO TERM_ERR;
     ON ENDFILE(CARDS) BEGIN;
        EOF_SW = '1';
        GO TO NEW_STRATA;
        END;
FRST_RD:   ;
     READ FILE(CARDS) INTO(INPUT_DATA);
    INYR = TRANSLATE(INYR,'0',' T);
    INCD = TRANSLATE(INCD,'0',' ');
    INLVLCD = TRANSLATE(INLVLCD,'0',' ');
     IF INYR < 1940 THEN GO TO FRST_RD;
     HOLD_YR = INYR;
     HOLD_LVLCD = INLVLCD;
     DIR_CD = HOLD_LVLCD;
     READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
     CALL GETDIR(DIR_CD,DIRX,CONTROL,DAF,DKEY,DIR#,RC);
     IF RC = -1 THEN DO;
        PUT SKIP DATA(DIR_CD,DKEY,DIR#);
        END;
     DDKEY = DREC(DIR#);
     IF DDKEY = -1 THEN DO;
        PUT SKIP DATA(DIR_CD,DKEY,DIR#);
        PUT SKIP LIST(DIRX(DIR#));
        PUT SKIP  LIST(INPUT_DATA);
        GO TO EOJ;
        END;
     IF DDKEY < 0 THEN PUT SKIP DATA(DDKEY);
     IF DDKEY > 114 THEN PUT SKIP DATA(DDKEY);
     READ FILE(DAF) INTO(DATA_BLK) KEY(DDKEY);
     GO TO CHK_CDS;
GET_DATA:   ;
     READ FILE(CARDS) INTO(INPUT_DATA);
    INYR = TRANSLATE(INYR,'0',' T);
      IF INYR < 1940 THEN GO TO GET_DATA;
    INCD = TRANSLATE(INCD,'0',' ');
    INLVLCD = TRANSLATE(INLVLCD,'0',' ');
     IF INLVLCD ¬= HOLD_LVLCD THEN GO TO NEW_STRATA;
     IF INYR < HOLD_YR THEN GO TO SEQ_ERR;
     IF INYR > HOLD_YR THEN GO TO PUT_DATA;
CHK_CDS:   ;
     IF INCD =  5 THEN GO TO MV_ASAR_PRECIP;
     IF INCD = 35 THEN GO TO MV_ASAR_T;
     IF INCD = 45 THEN GO TO MV_ASAR_ZINDEX;
     IF INCD = 101 THEN GO TO MV_ASAR_HARV;
     /*  IF INCD = 102 THEN GO TO MV_ASAR_PLNT;   */
     IF INCD = 103 THEN GO TO MV_ASAR_PROD;
     GO TO ERRNCD;
MV_ASAR_PRECIP:   ;
     DO N = 1 TO 12;
        IF ACHAR(N) = '        ' THEN INMON(N) = -9999;
        ASAR_PRECIP(N) = INMON(N);
        END;
     GO TO GET_DATA;
MV_ASAR_T:   ;
     DO N = 1 TO 12;
        IF ACHAR(N) = '        ' THEN INMON(N) = -9999;
        ASAR_TEMP(N) = INMON(N);
        END;
     GO TO GET_DATA;
MV_ASAR_ZINDEX:   ;
     DO N = 1 TO 12;
     IF ACHAR(N) = '        ' THEN INMON(N) = -9999;
     ASAR_ZINDEX(N) = INMON(N);
     END;
     GO TO GET_DATA;
MV_ASAR_HARV:   ;
     IF ACHARX(1) = '                ' THEN INAGDTA(1) = -9999;
     IF ACHARX(2) = '                ' THEN INAGDTA(2) = -9999;
     IF DCODE(7).SUBCODE(1) = 201 THEN ASAR_HARV(1) = INAGDTA(1);
     IF DCODE(7).SUBCODE(1) = 202 THEN ASAR_HARV(1) = INAGDTA(1);
     IF DCODE(7).SUBCODE(2) = 202 THEN ASAR_HARV(2) = INAGDTA(2);
     GO TO GET_DATA;
/*   THIS PARAGRAPH IS DUMMIED OUT AS A COMMENT
MV_ASAR_PLNT:   ;
     IF ACHARX(1) = '                ' THEN INAGDTA(1) = -99999;
     IF ACHARX(2) = '                ' THEN INAGDTA(2) = -99999;
     IF DCODE(8).SUBCODE(1) = 201 THEN ASAR_PLNT(1) = INAGDTA(1);
     IF DCODE(8).SUBCODE(1) = 202 THEN ASAR_PLNT(1) = INAGDTA(1);
     IF DCODE(8).SUBCODE(2) = 202 THEN ASAR_PLNT(2) = INAGDTA(2);
     GO TO GET_DATA;
            THE END OF COMMENT      */
```

```
MV_ASAR_PROD:  ;
     IF ACHARX(1) = '                 ' THEN INAGDTA(1) = -9999;
     IF ACHARX(2) = '                 ' THEN INAGDTA(2) = -9999;
     IF DCODE(8).SUBCODE(1) = 201 THEN ASAR_PROD(1) = INAGDTA(1);
     IF DCODE(8).SUBCODE(1) = 202 THEN ASAR_PROD(1) = INAGDTA(1);
     IF DCODE(8).SUBCODE(2) = 202 THEN ASAR_PROD(2) = INAGDTA(2);
     GO TO GET_DATA;
PUT_DATA:  ;
     IF YRCTR > 47 THEN GO TO NO_RM_ASAR;
     YRCTR = YRCTR + 1;
     STRTYR = STRTYR + 1;
     ASAR_YR = HOLD_YR;
       DATADESC.NUMBYRS_USED = YRCTR;
       ASAR_NXT_YR = DATADESC.LSTRECNO;
       ASAR_NXT_DISP = DATADESC.LSTDISP + 128;
       IF STRTYR = ASAR_YR THEN DO;           END;
          ELSE DO;
             FILYR = STRTYR;
             FIL_NXT_YR = ASAR_NXT_YR;
             FIL_NXT_DISP = ASAR_NXT_DISP;
             YR_BLK(YRCTR) = FILDATA;
             DATADESC.LSTDISP = DATADESC.LSTDISP + 128;
             GO TO PUT_DATA;
             END;
       YR_BLK(YRCTR) = DATA_OUT;
       DATADESC.LSTDISP = DATADESC.LSTDISP + 128;
     HOLD_YR = INYR;
     ASAR_TEMP = -9999;
     ASAR_PRECIP = -9999;
     ASAR_ZINDEX = -9999;
     ASAR_HARV = 0;
     ASAR_PROD = 0;
     GO TO CHK_CDS;
TERM_ERR:  ;
     PUT SKIP LIST('**CONVERSION ERROR **',INPUT_DATA);
     GO TO EOJ;
SEQ_ERR:  ;
     PUT SKIP LIST('** DATA NOT IN RIGHT SEQUENCE **',INPUT_DATA);
     GO TO EOJ;
NO_RM_ASAR:  ;
     PUT SKIP LIST('** NOT ENOUGH DISK SPACE ALLOCATED FOR ASAR **');
     PUT SKIP LIST(INPUT_DATA);
     PUT SKIP DATA(HOLD_YR,HOLD_LVLCD);
     GO TO EOJ;
ERRNCD:  ;
     PUT SKIP LIST('** INVALID CODE **',INPUT_DATA);
     GO TO EOJ;
NEW_STRATA:  ;
     IF YRCTR > 47 THEN GO TO NO_RM_ASAR;
     YRCTR = YRCTR + 1;
     ASAR_YR = HOLD_YR;
     ASAR_NXT_YR = -1;
     ASAR_NXT_DISP = 0;
       DATADESC.NUMBYRS_USED = YRCTR;
       YR_BLK(YRCTR) = DATA_OUT;
     HOLD_YR = INYR;
     HOLD_LVLCD = INLVLCD;
     DIR_CD = HOLD_LVLCD;
     REWRITE FILE(DAF) FROM(DATA_BLK) KEY(DDKEY);
     IF EOF_SW = '1' THEN GO TO EOJ;
     CALL GETDIR(DIR_CD,DIRK,CONTROL,DAF,DKEY,DIR#,RC);
     DDKEY = DREC(DIR#);
     IF DDKEY < 0 THEN DO;
       PUT SKIP DATA(DIR_CD,DIR#,DDKEY);
       PUT SKIP LIST(DIR(DIR#));
       END;
     IF DDKEY > 114 THEN PUT SKIP DATA(DIR_CD,DIR#,DDKEY);
     READ FILE(DAF) INTO(DATA_BLK) KEY(DDKEY);
     YRCTR = 0;
     STRTYR = 1939;
     ASAR_TEMP = -9999;
     ASAR_ZINDEX = -9999;
     ASAR_HARV = 0;
     ASAR_PROD = 0;
     GO TO CHK_CDS;
EOJ:  ;
     FREE ASAR_DATA;
     FREE INPUT_DATA;
     FREE FILDATA;
     END ASAR;
//LKED.SYSLIB DD
//             DD DSN=*.EDS.CCEA.LEC.LOAD,DISP=SHR
//GO.SYSPRINT DD SYSOUT=A
//CARDS DD DSN=&&CARDS,DISP=(OLD,DELETE),UNIT=SYSDA,
//   DCB=(RECFM=FB,LRECL=80,BLKSIZE=1680)
//DAF DD DSN=*.EDS.CCEA.MET.AUSARG,DISP=OLD
/*
```

```
RUN NO. 15      DATE  11/12/76     TIME  0910                LISTING OF MODULE CANADA

DESCRIPTION              DATA BASE PGM

MASTER FILE              W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER          10/13/76
LAST DATE COPIED           NONE
LAST UPDATE                NONE

PASSWORD                 GDBG
PROGRAMMER               LEC
LANGUAGE                 PLI
PROC PARAMETER           $NOJCL

   CANADA: PROC OPTIONS(MAIN):
       DCL 1 CONTROL,
               2 FILEID CHAR(6),
               2 NUMPASS FIXED BIN(15,0),
               2 PASS(8) CHAR(5),
               2 NUMLEV FIXED BIN(15,0),
               2 LEVNAME(8) CHAR(24),
               2 NUMCODE FIXED BIN(15,0),
               2 CODE(32),
                 3 CODENUM FIXED BIN(15,0),
                 3 UNITNUM FIXED BIN(15,0),
                 3 BASE FIXED BIN(15,0),
                 3 SCALE FIXED BIN(15,0),
                 3 CODENAME CHAR(24),
                 3 UNITNAME CHAR(24),
               2 NUMONE FIXED BIN(15,0),
               2 ONE(24),
                 3 CODEINUM FIXED BIN(15,0),
                 3 NUMDIRS FIXED BIN(15,0),
                 3 RECNUM FIXED BIN(15,0),
                 3 DISPLACE FIXED BIN(15,0),
                 3 NAME CHAR(24),
               2 FILERECS FIXED BIN(15,0),
               2 REC(601),
                 3 RECTYPE FIXED BIN(15,0),
                 3 FRESPACE FIXED BIN(15,0),
                 3 LOCATION FIXED BIN(15,0);
       DCL 1 DIRX,
               2 DIR (84),
               3 LEVNUM FIXED BIN(15,0),
               3 CODENUMB FIXED BIN(15,0),
               3 LAT FIXED BIN(15,0),
               3 LON FIXED BIN(15,0),
               3 DIRNAME CHAR(24),
               3 PREC FIXED BIN(15,0),
               3 PDISP FIXED BIN(15,0),
               3 BREC FIXED BIN(15,0),
               3 BDISP FIXED BIN(15,0),
               3 CREC FIXED BIN(15,0),
               3 CDISP FIXED BIN(15,0),
               3 DREC FIXED BIN(15,0),
               3 DDISP FIXED BIN(15,0),
               3 LEVCODE FIXED BIN(31,0),
               3 MODEL(4),
                 4 CROP FIXED BIN(15,0),
                 4 MREC FIXED BIN(15,0),
                 4 MDISP FIXED BIN(15,0),
          2 FILLER CHAR (56);
        DCL 1 INPUT_DATA BASED(Q),
                2  INYR            PIC '(7)9',
                2  INCO            PIC '999',
                2  INMON(12)       PIC 'SSSS9',
                2  INLVLID,
                   3  INCTRY       PIC '99',
                   3  INREG        PIC '99',
                   3  INZONE       PIC '99',
                   3  INSTRA       PIC '99',
                   3  INSBSTRA     PIC '99';
        DCL 1 DEF_DATA BASED(Q),
                2  FIL1            CHAR(10),
                2  IMAGDTA(6) PIC '(10)9',
                2  INLVLCO PIC '(10)9';
        DCL 1 CANADA_DATA BASED(P),
                2  CAN_YR               FIXED BIN(15),
                2  CAN_NXT_YR           FIXED BIN(15),
                2  CAN_NXT_DISP         FIXED BIN(15),
                2  RESERV2                FIXED BIN(15),
                2  CAN_MAX_TEMP(12)     FIXED BIN(15),
                2  CAN_MIN_TEMP(12)     FIXED BIN(15),
                2  CAN_TEMP(12)         FIXED BIN(15),
                2  CAN_PRECIP(12)       FIXED BIN(15),
                2  CAN_PLANT(3)         FIXED BIN(31),
                2  CAN_PROD(3)          FIXED BIN(31);
```

```
      DCL DATA_OUT CHAR(128) BASED(P);
      ALLOCATE CANADA_DATA;
      ALLOCATE INPUT_DATA;
      DCL 1 DATA_BLK,
            2 DATADESC,
             3 ID                FIXED BIN(31,0),
             3 WMO               FIXED BIN(31,0),
             3 LATI              FIXED BIN(15,0),
             3 LONGI             FIXED BIN(15,0),
             3 ELEV              FIXED BIN(15,0),
             3 TOTALBLKS_ALLOC   FIXED BIN(15,0),
             3 NUMBYRS_USED      FIXED BIN(15,0),
             3 BLOCKSIZE         FIXED BIN(15,0),
             3 FSTRECNO          FIXED BIN(15,0),
             3 FSTDISP           FIXED BIN(15,0),
             3 LSTRECNO          FIXED BIN(15,0),
             3 LSTDISP           FIXED BIN(15,0),
             3 RESERVED          CHAR(18),
             3 NUMBCODE          FIXED BIN(15,0),
             3 DCODE(12),
               4 CODENUMB           FIXED BIN(15,0),
               4 NUMSELEM           FIXED BIN(15,0),
               4 ELEMSIZE           FIXED BIN(15,0),
               4 NUMSCODE           FIXED BIN(15,0),
               4 SUBCODE(8)         FIXED BIN(15,0),
           2  YR_BLK(47)   CHAR(128),
           2  FIL          CHAR(88) INIT('  ');
      DCL HOLD_LVLCD PIC'(10)9';
      DCL HOLD_YR PIC '(7)9';
      DCL DIR_CD FIXED BIN(31);
      DCL (DKEY,CKEY,DDKEY,DIR#,RC,YRCTR) FIXED BIN(15);
      CKEY=0;
      YRCTR=0;
      CAN_MAX_TEMP = -9999;
      CAN_MIN_TEMP = -9999;
      CAN_TEMP     = -9999;
      CAN_PRECIP   = -9999;
      CAN_PLANT    = 0;
      CAN_PROD     = 0;
      DCL EOF_SW CHAR(1) INIT('0');
      DCL GETDIR EXTERNAL ENTRY;
      DCL CARDS FILE RECORD INPUT;
      DCL DAF FILE RECORD DIRECT UPDATE KEYED ENV(REGIONAL(1));
      ON CONVERSION GO TO TERM_ERR;
      ON ENDFILE(CARDS) BEGIN;
         EOF_SW = '1';
         GO TO NEW_STRATA;
         END;
   FRST_RD:   ;
       READ FILE(CARDS) INTO(INPUT_DATA);
      INYR = TRANSLATE(INYR,'0',' T);
      INCD = TRANSLATE(INCD,'0',' ');
      INLVLCD = TRANSLATE(INLVLCD,'0',' ');
       HOLD_YR = INYR;
       HOLD_LVLCD = INLVLCD;
       DIR_CD = HOLD_LVLCD;
       READ FILE(DAFT INTO(CONTROL) KEY(CKEY);
       CALL GETDIR(DIR_CD,DIRK,CONTROL,DAF,DKEY,DIR#,RC);
       DDKEY = DREC(DIR#);
       READ FILE(DAF) INTO(DATA_BLK) KEY(DDKEY);
       GO TO CHK_CDS;
   GET_DATA:   ;
       READ FILE(CARDS) INTO(INPUT_DATA);
      INYR = TRANSLATE(INYR,'0',' T);
      INCD = TRANSLATE(INCD,'0',' ');
      INLVLCD = TRANSLATE(INLVLCD,'0',' ');
       IF INLVLCD ¬= HOLD_LVLCD THEN GO TO NEW_STRATA;
       IF INYR < HOLD_YR THEN GO TO SEQ_ERR;
       IF INYR > HOLD_YR THEN GO TO PUT_DATA;
   CHK_CDS:   ;
       IF INCD =  5 THEN GO TO MV_CAN_PRECIP;
       IF INCD = 15 THEN GO TO MV_CAN_TX;
       IF INCD = 25 THEN GO TO MV_CAN_IN;
       IF INCD = 35 THEN GO TO MV_CAN_T;
       IF INCD = 102 THEN GO TO MV_CAN_PLT;
       IF INCD = 103 THEN GO TO MV_CAN_PROD;
       GO TO ERRNCD;
   MV_CAN_PRECIP:   ;
       DO N = 1 TO 12;
          CAN_PRECIP(N) = INMON(N);
          END;
       GO TO GET_DATA;
```

```
MV_CAN_TX:  ;
     DO N = 1 TO 12;
        CAN_MAX_TEMP(N) = INMON(N);
        END;
     GO TO GET_DATA;
MV_CAN_TN:  ;
     DO N = 1 TO 12;
        CAN_MIN_TEMP(N) = INMON(N);
        END;
     GO TO GET_DATA;
MV_CAN_T:  ;
     DO N = 1 TO 12;
        CAN_TEMP(N) = INMON(N);
        END;
     GO TO GET_DATA;
MV_CAN_PLT:  ;
     CAN_PLANT(1) = INAGDTA(1);
     GO TO GET_DATA;
MV_CAN_PROD:  ;
     CAN_PROD(1) = INAGDTA(1);
     GO TO GET_DATA;
PUT_DATA:  ;
     IF YRCTR > 47 THEN GO TO NO_RM_CAN;
     YRCTR = YRCTR + 1;
     CAN_YR = HOLD_YR;
     LSTRECNO = LSTRECNO;
     LSTDISP = LSTDISP;
     NUMBYRS_USED = YRCTR;
     CAN_NXT_YR = LSTRECNO;
     CAN_NXT_DISP = LSTDISP + 128;
     YR_BLK(YRCTR) = DATA_OUT;
      LSTDISP = LSTDISP + 128;
     HOLD_YR = INYR;
     CAN_MAX_TEMP = -9999;
     CAN_MIN_TEMP = -9999;
     CAN_TEMP     = -9999;
     CAN_PRECIP   = -9999;
     CAN_PLANT    = 0;
     CAN_PROD     = 0;
     GO TO CHK_CDS;
TERM_ERR:  ;
     PUT SKIP LIST('**CONVERSION ERROR **',INPUT_DATA);
     GO TO EOJ;
SEQ_ERR:  ;
     PUT SKIP LIST('** DATA NOT IN RIGHT SEQUENCE **',INPUT_DATA)
     GO TO EOJ;
NO_RM_CAN:  ;
     PUT SKIP LIST('** NOT ENOUGH DISK SPACE ALLOCATED FOR CANADA **');
     GO TO EOJ;
ERRNCD:  ;
     PUT SKIP LIST('** INVALID CODE **',INPUT_DATA);
     GO TO EOJ;
NEW_STRATA:  ;
     IF YRCTR > 47 THEN GO TO NO_RM_CAN;
     YRCTR = YRCTR + 1;
     CAN_YR = HOLD_YR;
     LSTRECNO = LSTRECNO;
     LSTDISP = LSTDISP;
     NUMBYRS_USED = YRCTR;
     CAN_NXT_YR = -1;
     CAN_NXT_DISP = 0;
     YR_BLK(YRCTR) = DATA_OUT;
     HOLD_YR = INYR;
     HOLD_LVLCD = INLVLCD;
     DIR_CD = HOLD_LVLCD;
     REWRITE FILE(DAF) FROM(DATA_BLK) KEY(DDKEY);
     IF EOF_SW = '1' THEN GO TO EOJ;
     CALL GETDIR(DIR_CD,DIRX,CONTROL,DAF,DKEY,DIR#,RC);
     DDKEY = DREC(DIR#);
     READ FILE(DAF) INTO(DATA_BLK) KEY(DDKEY);
     YRCTR = 0;
     CAN_MAX_TEMP = -9999;
     CAN_MIN_TEMP = -9999;
     CAN_TEMP     = -9999;
     CAN_PRECIP   = -9999;
     CAN_PLANT    = 0;
     CAN_PROD     = 0;
     GO TO CHK_CDS;
EOJ:  ;
     FREE CANADA_DATA;
     FREE INPUT_DATA;
     END CANADA;
```

```
RUN NO. 15       DATE  11/12/75      TIME  0910                LISTING OF MODULE USA

DESCRIPTION             DATA BASE PGM

MASTER FILE             W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER         10/13/75
LAST DATE COPIED           NONE
LAST UPDATE                NONE

PASSWORD                WCEZ
PROGRAMMER              LEC
LANGUAGE                PLI
PROC PARAMETER          $NOJCL

    USA: PROC OPTIONS(MAIN);
       DCL 1 CONTROL,
              2 FILEID CHAR(8),
              2 NUMPASS FIXED BIN(15,0),
              2 PASS(8) CHAR(8),
              2 NUMLEV FIXED BIN(15,0),
              2 LEVNAME(8) CHAR(24),
              2 NUMCODE FIXED BIN(15,0),
              2 CODE(32),
                3 CODENUM FIXED BIN(15,0),
                3 UNITNUM FIXED BIN(15,0),
                3 BASE FIXED BIN(15,0),
                3 SCALE FIXED BIN(15,0),
                3 CODENAME CHAR(24),
                3 UNITNAME CHAR(24),
              2 NUMONE FIXED BIN(15,0),
              2 ONE(24),
                3 CODE1NUM FIXED BIN(15,0),
                3 NUMDIRS FIXED BIN(15,0),
                3 RECNUM FIXED BIN(15,0),
                3 DISPLACE FIXED BIN(15,0),
                3 NAME CHAR(24),
              2 FILERECS FIXED BIN(15,0),
              2 REC(601),
                3 RECTYPE FIXED BIN(15,0),
                3 FRESPACE FIXED BIN(15,0),
                3 LOCATION FIXED BIN(15,0);
       DCL 1 DIRX,
              2 DIR (84),
                3 LEVNUM FIXED BIN(15,0),
                3 CODENUMR FIXED BIN(15,0),
                3 LAT FIXED BIN(15,0),
                3 LON FIXED BIN(15,0),
                3 DIRNAME CHAR(24),
                3 PREC FIXED BIN(15,0),
                3 PDISP FIXED BIN(15,0),
                3 BREC FIXED BIN(15,0),
                3 BDISP FIXED BIN(15,0),
                3 CREC FIXED BIN(15,0),
                3 CDISP FIXED BIN(15,0),
                3 DREC FIXED BIN(15,0),
                3 DDISP FIXED BIN(15,0),
                3 LEVCODE FIXED BIN(31,0),
                3 MODEL(4),
                  4 CROP FIXED BIN(15,0),
                  4 MREC FIXED BIN(15,0),
                  4 MDISP FIXED BIN(15,0),
          2 FILLER CHAR (56);
         DCL 1 INPUT_DATA BASED(Q),
                2  INYR          PIC '(7)9',
                2  INCD          PIC '999',
                2  INMON(12)     PIC 'SSSS9',
                2  INLVLID,
                   3  INCTRY     PIC '99',
                   3  INREG      PIC '99',
                   3  INZONE     PIC '99',
                   3  INSTRA     PIC '99',
                   3  INSBSTRA   PIC '99';
         DCL 1 DEF_DATA BASED(Q),
                2  FIL1          CHAR(10),
                2  IMAGDTA(6) PIC 'SSSSSSSSS9',
                2  INLVLCD PIC '(10)9';
         DCL 1 DEFA_DATA BASED(Q),
                2  FIL1          CHAR(10),
                2  ACHAR(12)     CHAR(5),
                2  FIL2          CHAR(10);
         DCL 1 DEFX_DATA BASED(Q),
                2  FIL1          CHAR(10),
                2  ACHARX(6)     CHAR(10),
                2  FIL2          CHAR(10);
```

```
   DCL 1 USA_DATA BASED(P).
        2  USA_YR               FIXED BIN(15).
        2  USA_NXT_YR           FIXED BIN(15).
        2  USA_NXT_DISP         FIXED BIN(15).
        2  RESERV2               FIXED BIN(15),
        2  USA_TEMP(12)  .      FIXED BIN(15).
        2  USA_PRECIP(12)       FIXED BIN(15).
        2  USA_DEGDAY(12)       FIXED BIN(15).
        2  USA_HARV(4)         FIXED BIN(31).
        2  USA_PLNT(4)          FIXED BIN(31).
        2  USA_PROD(4)          FIXED BIN(31);
   DCL DATA_OUT CHAR(128) BASED(P);
   ALLOCATE USA_DATA;
   USA_DATA.RESERV2 = 0;
   ALLOCATE INPUT_DATA;
   DCL 1 DATA_BLK,
          2 DATADESC.
         3 ID                 FIXED BIN(31.0).
         3 XMO                FIXED BIN(31.0).
         3 LATI               FIXED BIN(15.0).
         3 LONGI              FIXED BIN(15.0).
         3 ELEV               FIXED BIN(15.0).
         3 TOTALBLKS_ALLOC    FIXED BIN(15.0).
         3 NUMBYRS_USED       FIXED BIN(15.0).
         3 BLOCKSIZE          FIXED BIN(15.0).
         3 FSTRECNO           FIXED BIN(15.0).
         3 FSTDISP            FIXED BIN(15.0).
         3 LSTRECNO           FIXED BIN(15.0).
         3 LSTDISP            FIXED BIN(15.0).
         3 RESERVED           CHAR(18).
         3 NUMHCODE           FIXED BIN(15.0).
         3 DCODE(12).
           4 CODENUMB            FIXED BIN(15.0).
           4 NUMSELEM            FIXED BIN(15.0).
           4 ELEMSIZE            FIXED BIN(15.0).
           4 NUMSCODE            FIXED BIN(15.0).
           4 SUBCODE(8)          FIXED BIN(15.0).
         2  YR_BLK(47)    CHAR(128).
         2  FIL           CHAR(64) INIT('  ');
    DCL HOLD_L/LCD PIC'(10)9';
    DCL HOLD_YR PIC '(7)9';
    DCL DIR_CD FIXED BIN(31);
    DCL FILDATA CHAR(128) BASED(P1);
    DCL 1 DUMUSA BASED(P1).
           2  FILYR FIXED BIN(15).
           2  FIL_NXT_YR FIXED BIN(15).
           2  FIL_NXT_DISP FIXED BIN(15).
           2  FIL_RESV FIXED BIN(15).
           2  FIL_TEMP(12)  FIXED BIN(15).
           2  FIL_PRECIP(12) FIXED BIN(15).
          2  FIL_DEGDAY(12)  FIXED BIN(15).
           2  FIL_HARV(4) FIXED BIN(31).
          2  FIL_PLNT(4)     FIXED BIN(31).
           2  FIL_PROD(4) FIXED BIN(31);
    ALLOCATE FILDATA;
    FIL_RESV = 0;
    FIL_TEMP = -9999;
    FIL_PRECIP = -9999;
       FIL_DEGDAY = -9999;
    FIL_HARV = 0;
     FIL_PLNT = 0;
    FIL_PROD = 0;
    DCL (DKEY.CKEY.DJKEY.DIRX.RC.YRCTR) FIXED BIN(15);
    CKEY=0;
    DCL STRTYR FIXED BIN(15);
    STRTYR = 1930;
    YRCTR=0;
    RESERV2 = 0;
    USA_TEMP = -9999;
    USA_PRECIP = -9999;
    USA_DEGDAY = -9999;
    USA_HARV = 0;
    USA_PLNT = 0;
    USA_PROD = 0;
    DCL EOF_S  CHAR(1) INIT('0');
    DCL GETDIR EXTERNAL ENTRY;
    DCL CARDS FILE RECORD INPUT;
    DCL DAF FILE RECORD DIRECT UPDATE KEYED ENV(REGIONAL(1));
    ON CONVERSION GO TO TERM_ERR;
    ON ENDFILE(CARDS) BEGIN;
      EOF_SW = '1';
      GO TO NEW_STRATA;
      END;
FRST_RD:  ;  .
    READ FILE(CARDS) INTO(INPUT_DATA);
```

#77

```
         INYR = TRANSLATE(INYR.'0'.' '):
         INCD = TRANSLATE(INCD.'0'.' '):
         INLVLCD = TRANSLATE(INLVLCD.'0'.' '):
          IF INYR < 1931 THEN GO TO FRST_RD:
          HOLD_YR = INYR:
          HOLD_LVLCD = INLVLCD:
          DIR_CD = HOLD_LVLCD:
          READ FILE(DAFT INTO(CONTROL) KEY(CKEY);
          CALL GETDIR(DIR_CD.DIRX.CONTROL.DAF.DKEY.DIR#.RC):
          IF RC = -1 THEN DO:
            PUT SKIP DATA(DIR_CD.DKEY.DIR#):
            END:
          DDKEY = DREC(DIR#):
          IF DDKEY = -1 THEN DO:
            PUT SKIP DATA(DIR_CD.DKEY.DIR#):
            PUT SKIP LIST(DIR(DIR#)):
            PUT SKIP  LIST(INPUT_DATA):
            GO TO EOJ:
            END:
          IF DDKEY < 0 THEN PUT SKIP DATA(DDKEY):
          IF DDKEY > 114 THEN PUT SKIP DATA(DDKEY):
          READ FILE(DAF) INTO(DATA_BLK) KEY(DDKEY):
          GO TO CHK_CDS:
      GET_DATA:  :
          READ FILE(CARDS) INTO(INPUT_DATA):
         INYR = TRANSLATE(INYR.'0'.' '):
           IF INYR < 1931 THEN GO TO GET_DATA:
         INCD = TRANSLATE(INCD.'0'.' '):
         INLVLCD = TRANSLATE(INLVLCD.'0'.' '):
          IF INLVLCD = 0301460000 THEN GO TO GET_DATA:
          IF INLVLCD ¬= HOLD_LVLCD THEN GO TO NEW_STRATA:
          IF INYR < HOLD_YR THEN GO TO SEQ_ERR:
          IF INYR > HOLD_YR THEN GO TO PUT_DATA:
      CHK_CDS:  :
          IF INCD =  5 THEN GO TO MV_USA_PRECIP:
          IF INCD = 35 THEN GO TO MV_USA_T:
          IF INCD = 40 THEN GO TO MV_USA_DEGDAY:
          IF INCD = 101 THEN GO TO MV_USA_HARV:
          IF INCD = 102 THEN GO TO MV_USA_PLNT:
          IF INCD = 103 THEN GO TO MV_USA_PROD:
          GO TO ERRNCD:
      MV_USA_PRECIP:  :
          DO N = 1 TO 12:
            IF ACHAR(N) = '      ' THEN INMON(N) = -9999:
            USA_PRECIP(N) = INMON(N):
            END:
          GO TO GET_DATA:
      MV_USA_T:  :
          DO N = 1 TO 12:
            IF ACHAR(N) = '      ' THEN INMON(N) = -9999:
            USA_TEMP(N) = INMON(N):
            END:
          GO TO GET_DATA:
      MV_USA_DEGDAY:  :
          DO N = 1 TO 12:
          IF ACHAR(N) = '      ' THEN INMON(N) = -9999:
           USA_DEGDAY(N) = INMON(N):
            END:
          GO TO GET_DATA:
      MV_USA_HARV:  :
          IF ACHARX(1) = '               ' THEN INAGDTA(1) = -9999:
          IF ACHARX(2) = '               ' THEN INAGDTA(2) = -9999:
          IF DCODE(7).SUBCODE(1) = 201 THEN USA_HARV(1) = INAGDTA(1):
          IF DCODE(7).SUBCODE(1) = 202 THEN USA_HARV(1) = INAGDTA(1):
          IF DCODE(7).SUBCODE(2) = 202 THEN USA_HARV(2) = INAGDTA(2):
          GO TO GET_DATA:
      MV_USA_PLNT:  :
          IF ACHARX(1) = '               ' THEN INAGDTA(1) = -99999:
          IF ACHARY(2) = '               ' THEN INAGDTA(2) = -99999:
          IF DCODE(8).SUBCODE(1) = 201 THEN USA_PLNT(1) = INAGDTA(1):
          IF DCODE(8).SUBCODE(1) = 202 THEN USA_PLNT(1) = INAGDTA(1):
          IF DCODE(8).SUBCODE(2) = 202 THEN USA_PLNT(2) = INAGDTA(2):
          GO TO GET_DATA:
      MV_USA_PROD:  :
          IF ACHARX(1) = '               ' THEN INAGDTA(1) = -9999:
          IF ACHARX(2) = '               ' THEN INAGDTA(2) = -9999:
          IF DCODE(9).SUBCODE(1) = 201 THEN USA_PROD(1) = INAGDTA(1):
          IF DCODE(9).SUBCODE(1) = 202 THEN USA_PROD(1) = INAGDTA(1):
          IF DCODE(9).SUBCODE(2) = 202 THEN USA_PROD(2) = INAGDTA(2):
          GO TO GET_DATA:
      PUT_DATA:  :
          IF YRCTR > 47 THEN GO TO 40_RM_USA:
          YRCTR = YRCTR + 1:
```

```
        GO TO EOJ;
    NO_RM_USA:  :
        PUT SKIP LIST('** NOT ENOUGH DISK SPACE ALLOCATED FOR USA **');
        PUT SKIP LIST(INPUT_DATA);
        PUT SKIP DATA(HOLD_YR.HOLD_LVLCD);
        GO TO EOJ;
    ERRNCD:  :
        PUT SKIP LIST('** INVALID CODE **'.INPUT_DATA);
        GO TO EOJ;
    NEW_STRATA:  :
        IF YRCTR > 47 THEN GO TO NO_RM_USA;
        YRCTR = YRCTR + 1;
        USA_YR = HOLD_YR;
        USA_NXT_YR = -1;
        USA_NXT_DISP = 0;
          DATADESC.NUMBYRS_USED = YRCTR;
          YR_BLK(YRCTR) = DATA_OUT;
        HOLD_YR = INYR;
        HOLD_LVLCD = INLVLCD;
        DIR_CD = HOLD_LVLCD;
        REWRITE FILE(DAF) FROM(DATA_BLK) KEY(DDKEY);
        IF EOF_SW = '1' THEN GO TO EOJ;
        CALL GETDIR(DIR_CD.DIRX.CONTROL.DAF.DKEY.DIR#.RC);
        DDKEY = DREC(DIR#);
        IF DDKEY < 0 THEN DO;
          PUT SKIP DATA(DIR_CD.DIR#.DDKEY);
          PUT SKIP LIST(DIR(DIR#));
          END;
        IF DDKEY > 114 THEN PUT SKIP DATA(DIR_CD.DIR#.DDKEY);
        READ FILE(DAF) INTO(DATA_BLK) KEY(DDKEY);
        YRCTR = 0;
        STRTYR = 1930;
        USA_TEMP = -9999;
        USA_DEGDAY = -9999;
        USA_HARV = 0;
        USA_PLNT = 0;
        USA_PROD = 0;
        GO TO CHK_CDS;
    EOJ:  :
        FREE USA_DATA;
        FREE INPUT_DATA;
        FREE FILDATA;
        END USA;
        STRTYR = STRTYR + 1;
        USA_YR = HOLD_YR;
          DATADESC.NUMBYRS_USED = YRCTR;
          USA_NXT_YR = DATADESC.LSTRECNO;
          USA_NXT_DISP = DATADESC.LSTDISP + 128;
          IF STRTYR = USA_YR THEN DO;          END;
            ELSE DO;
              FILYR = STRTYR;
              FIL_NXT_YR = USA_NXT_YR;
              FIL_NXT_DISP = USA_NXT_DISP;
              YR_BLK(YRCTR) = FILDATA;
              DATADESC.LSTDISP = DATADESC.LSTDISP + 128;
              GO TO PUT_DATA;
              END;
          YR_BLK(YRCTR) = DATA_OUT;
          DATADESC.LSTDISP = DATADESC.LSTDISP + 128;
        HOLD_YR = INYR;
        USA_TEMP = -9999;
        USA_PRECIP = -9999;
        USA_DEGDAY = -9999;
        USA_HARV = 0;
        USA_PLNT = 0;
        USA_PROD = 0;
        GO TO CHK_CDS;
    TERM_ERR:  :
        PUT SKIP LIST('**CONVERSION ERROR **'.INPUT_DATA);
        GO TO EOJ;
    SEQ_ERR:  :
        PUT SKIP LIST('** DATA NOT IN RIGHT SEQUENCE **'.INPUT_DATA);
```

3.3.13 CONTROL BLOCK LISTER (YESLS02)

YESLS02 is provided to list the contents of the control block.

3.3.13.1 Linkages

None.

3.3.13.2 Interfaces

INITIAL must be run before YESLS02.

3.3.13.3 Inputs

Card containing ++END OF COMMAND.

3.3.13.4 Outputs

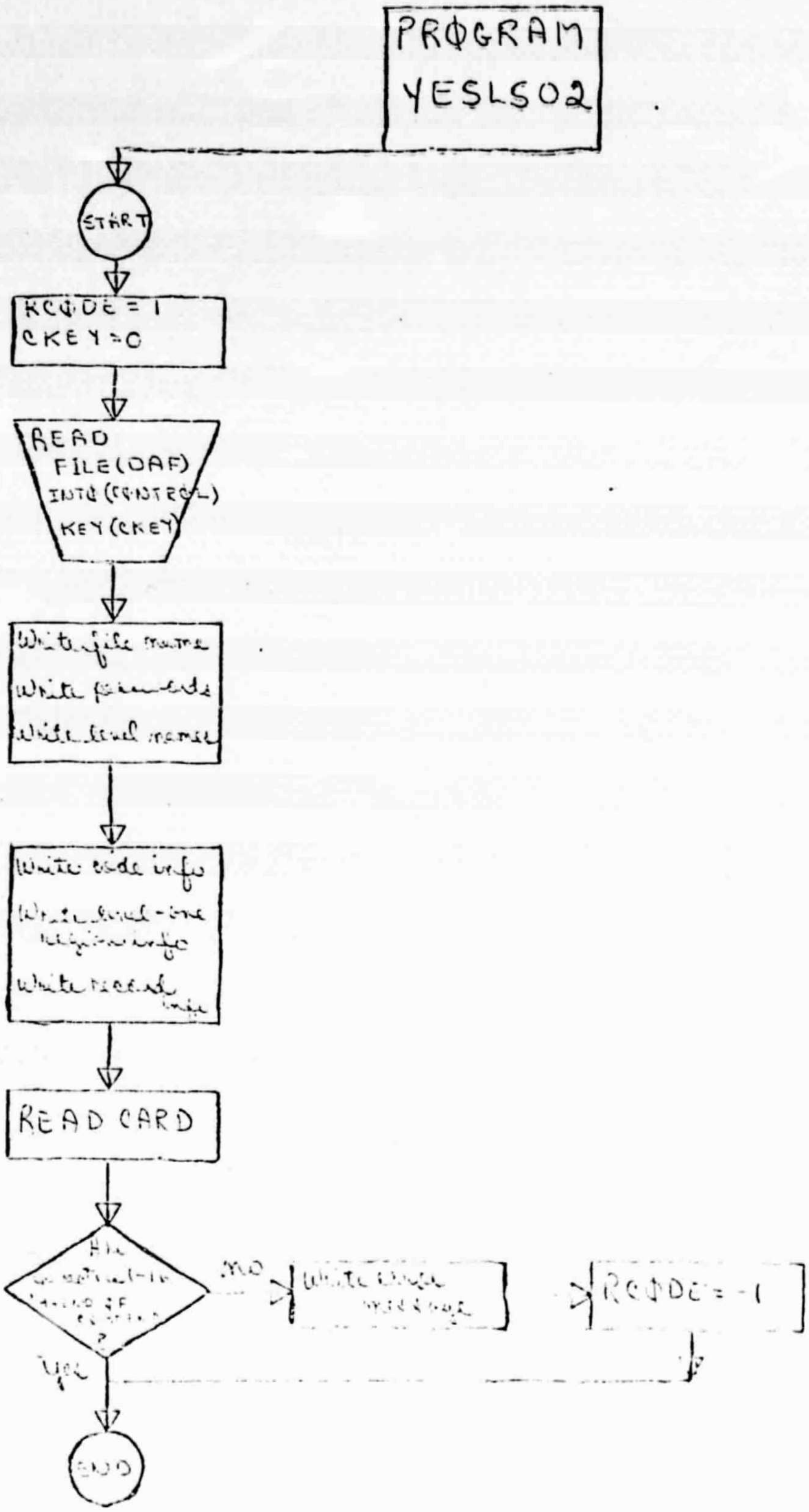
Control block listing.

3.3.13.5 Flow Chart

Next page.

3.3.13.6 Listing

Follows flow chart.


PRØGRAM
YES$LS02
START
KCØDE = 1
CKEY = 0
READ
FILE(DAF)
INTO (CONTROL)
KEY (CKEY)
Write file name
Write pointers
Write level names
Write code info
Write level-one region info
Write record info
READ CARD
All
?
no
Write error message
RCØDE = -1
yes
END

```
RUN NO. 15        DATE  11/12/76     TIME  0910                  LISTING OF MODULE LISTCON

DESCRIPTION           LIST DATA BASE PGM

MASTER FILE           W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER       10/13/76
LAST DATE COPIED        NONE
LAST UPDATE             NONE

PASSWORD              VKRH
PROGRAMMER            LEC
LANGUAGE              PLI
PROC PARAMETER        $NOJCL

   YESLS02:PROC OPTIONS(MAIN);
   /*                                                                          */
   /* ~~T-IS [illegible] TO LIST [illegible]~~                                 */
   /*                                                                          */
    DCL 1 CONTROL,
            2 FILEID CHAR(8),
            2 NUMPASS FIXED BIN(15,0),
            2 PASS(8) CHAR (8),
            2 NUMLEV FIXED BIN(15,0),
            2 LEVNAME(8) CHAR(24),
            2 NUMCODE FIXED BIN(15,0),
            2 CODE(32),
              3 CODENUM FIXED BIN(15,0),
              3 UNITNUM FIXED BIN(15,0),
              3 BASE FIXED BIN(15,0),
              3 SCALE FIXED BIN(15,0),
              3 CODENAME CHAR(24),
              3 UNITNAME CHAR(24),
            2 NUMONE FIXED BIN(15,0),
            2 ONE(24),
              3 CODEINUM FIXED BIN(15,0),
              3 NUMDIRS FIXED BIN(15,0),
              3 RECNUM FIXED BIN(15,0),
              3 DISPLACE FIXED BIN(15,0),
              3 NAME CHAR(24),
            2 FILERECS FIXED BIN(15,0),
            2 REC(601),
              3 RECTYPE FIXED BIN(15,0),
              3 FRESPACE FIXED BIN(15,0),
              3 LOCATION FIXED BIN(15,0);
    DCL CKEY FIXED BIN(10,0);
    DCL P1 POINTER;
    DCL D CHAR(8) BASED(P1);
    DCL B BIT(64) BASED(P1);
    DCL (I,J,NJOB,RCODE) FIXED BIN(15,0);
    DCL INSTR CHAR(80);
    DCL SYSIN FILE STREAM INPUT;
    DCL SYSPRINT FILE STREAM OUTPUT;
    DCL DAF FILE RECORD DIRECT KEYED ENV(REGIONAL(1));
    DCL SUBSTR BUILTIN;
   DCL PW(8) CHAR(8);
   DCL CONCARD CHAR(16);
    OPEN FILE(SYSPRINT) STREAM OUTPUT LINESIZE(130);
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
   RCODE=1;
   CKEY.I=0;
   READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
   PUT SKIP FILE(SYSPRINT) EDIT('+++LIST CONTROL BLOCK PROGRAM+++',
     'THE FILE IDENTIFICATION NAME IS ',FILEID) (A,SKIP(4),A,A(8));
   ALLOCATE D SET(P1);
   DO I=1 TO NUMPASS;
     D=PASS(I);
           B=¬B;
     PW(I)=D;
   END;
   PUT SKIP(3) FILE(SYSPRINT) EDIT ('THE PASSWORD(S) ARE ',(PW(I) DO
     I=1 TO NUMPASS)) (A,8 A);
   PUT SKIP(3) FILE(SYSPRINT) EDIT ('THE LEVEL NAMES ARE ',(LEVNAME(I)
     DO I=1 TO NUMLEV)) (A,4 A, SKIP, X(20),4 A);
   PUT SKIP(3) FILE(SYSPRINT) EDIT
     ('THE FOLLOWING CODE NUMBERS ARE USED FOR DATA IN THE FILE',
     'CODE#  UNIT#  CODE NAME','UNIT NAME','BASE  SCALE',
     ( CODE(I).CODENUM,CODE(I).UNITNUM,CODE(I).CODENAME,CODE(I).UNITNAME,
     CODE(I).BASE,CODE(I).SCALE  DO I=1 TO NUMCODE)) (A,SKIP,X(5),A,
     2(X(15),A),SKIP,32(2 F(7,0),X(3),2 A(24),F(4,0),F(6,0),SKIP));
   PUT SKIP(3) FILE(SYSPRINT) EDIT
     ('THE FOLLOWING COUNTRIES ARE INCLUDED IN THE FILE','CODE#  COUNTRY',
     'NUMBER OF DIRECTORIES  LOCATION OF FIRST DIRECTORY',
     ( ONE(I).CODEINUM,ONE(I).NAME,ONE(I).NUMDIRS,ONE(I).RECNUM,
     ONE(I).DISPLACE  DO I=1 TO NUMONE)) (A,SKIP,X(3),A,X(18),A,SKIP,
     24(F(7,0),X(3),A,X(9),F(4,0),X(18),2 F(5,0),SKIP));
   PUT SKIP(3) FILE(SYSPRINT) EDIT('THE FILE HAS BEEN DEFINED TO CONTAIN',
     FILERECS,' RECORDS, EACH 6440 BYTES LONG',
     'RECORD NUMBER ZERO CONTAINS THE CONTROL BLOCK FOR THE FILE',
     'KEY TO RECORD TYPE CODES: CODE#  RECORD TYPE',' 0     BLANK,UNUSED',
     '-1     DIRECTORY BLOCK','+1     DATA DESCRIPTOR AND DATA BLOCK',
     '+2     MODEL DEFINITION BLOCK') (A,F(4,0),A,SKIP(3),A,SKIP(3),
     A,SKIP,4(X(27),A,SKIP));
   PUT PAGE FILE(SYSPRINT) EDIT
     ('RECORD#  TYPE  FREESPACE(IN BYTES)  LOCATION OF FREESPACE',
      'RECORD#  TYPE  FREESPACE(IN BYTES)  LOCATION OF FREESPACE',
     ( I,REC(I).RECTYPE,REC(I).FRESPACE,REC(I).LOCATION  DO I=1 TO
     FILERECS)) (X(2),A,X(11),A,SKIP,300(2 F(7,0),X(9),F(4,0),X(18),
     F(4,0),X(19),2 F(7,0),X(9),F(4,0),X(14),F(4,0),SKIP));
   GET FILE(SYSIN) EDIT (CONCARD) (COL(1),A(15));
   IF CONCARD='++END OF COMMAND' THEN GOTO EOJ;
   PUT SKIP FILE(SYSPRINT) EDIT ('***END OF COMMAND CARD MISSING***') (A);
   RCODE=-1;
   FREE D;
   EOJ:RETURN;
   END YESLS02;
```

3.3.14 DIRECTORY BLOCK LISTER (YESLS04)

YESLS04 is provided to list directory information.

3.3.14.1 Linkages

None.

3.3.14.2 Interfaces

The directories requested must have been defined.

3.3.14.3 Inputs

Cards requesting directories.

3.3.14.4 Outputs

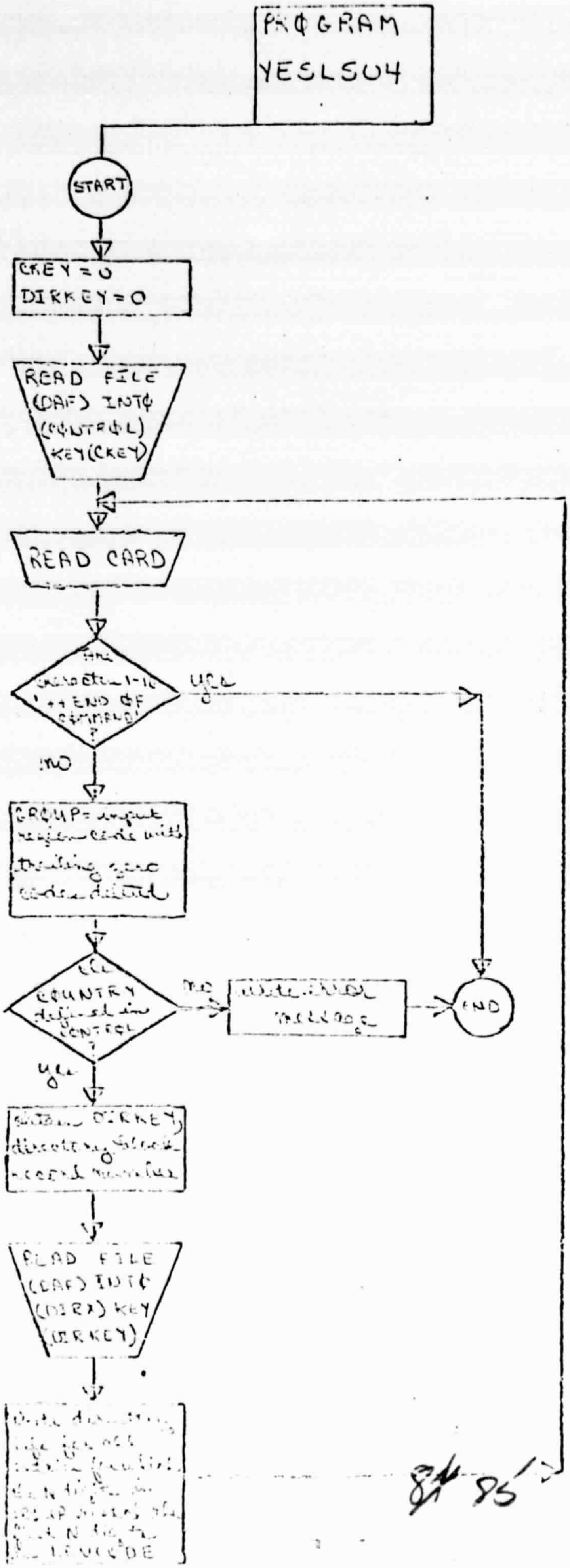
Directory listings.

3.3.14.5 Flow Chart

Next page.

3.3.14.6 Listing

Follows flow chart.


PRØGRAM
YESLSO4
START
CKEY=0
DIRKEY=0
READ FILE (DAF) INTØ (CØNTRØL) KEY(CKEY)
READ CARD
Are characters 1-10 'END OF COMMANDS' ?
yes
no
GRØUP = input region code with trailing zero codes deleted
Is COUNTRY defined in CONTROL ?
no
Write error message
END
yes
Obtain DIRKEY, directory block record number
READ FILE (DAF) INTØ (DIRX) KEY (DIRKEY)
Write directory info for all [illegible] GROUP [illegible] LEVCODE

```
RUN NO. 18        DATE  11/12/76     TIME  0920              LISTING OF MODULE LISTDIR

DESCRIPTION            LIST DATA BASE PGM

MASTER FILE            W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER        10/13/76
LAST DATE COPIED          NONE
LAST UPDATE               NONE

PASSWORD               XGHT
PROGRAMMER             LEC
LANGUAGE               PLI
PROC PARAMETER         $NOJCL

   YESLS04: PROC OPTIONS(MAIN):
   /*                                                                    */
   ~~/*THIS PROGRAM IS CALLED BY [illegible] TO LIST DIRECTORY ENTRIES*/~~
   /*                                                                    */
       DCL 1 CONTROL.
               2 FILEID CHAR(8).
               2 NUMPASS FIXED BIN(15.0).
               2 PASS(8) CHAR(8).
               2 NUMLEV FIXED BIN(15.0).
               2 LEVNAME(8) CHAR(24),
               2 NUMCODE FIXED BIN(15.0).
               2 CODE(32),
                  3 CODENUM FIXED BIN(15.0).
                  3 UNITNUM FIXED BIN(15.0).
                  3 BASE FIXED BIN(15.0).
                  3 SCALE FIXED BIN(15.0).
                  3 CODENAME CHAR(24).
                  3 UNITNAME CHAR(24).
               2 NUMONE FIXED BIN(15.0).
               2 ONE(24).
                  3 CODEINUM FIXED BIN(15.0).
                  3 NUMDIRS FIXED BIN(15.0).
                  3 RECNUM FIXED BIN(15.0).
                  3 DISPLACE FIXED BIN(15.0).
                  3 NAME CHAR(24).
               2 FILERECS FIXED BIN(15.0).
               2 REC(601).
                  3 RECTYPE FIXED BIN(15.0),
                  3 FRESPACE FIXED BIN(15.0).
                  3 LOCATION FIXED BIN(15.0):
       DCL 1 DIRX.
               2 DIR (84).
               3 LEVNUM FIXED BIN(15.0).
               3 CODENUMB FIXED BIN(15.0).
               3 LAT FIXED BIN(15.0).
               3 LON FIXED BIN(15.0),
               3 DIRNAME CHAR(24).
               3 PREC FIXED BIN(15.0).
               3 PDISP FIXED BIN(15.0).
               3 BREC FIXED BIN(15.0).
               3 BDISP FIXED BIN(15.0).
               3 CREC FIXED BIN(15.0).
               3 CDISP FIXED BIN(15.0),
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
                 3 DREC FIXED BIN(15.0).
                 3 DDISP FIXED BIN(15.0).
                 3 LEVCODE FIXED BIN(31.0).
                 3 MODEL(4).
                   4 CROP FIXED BIN(15.0).
                   4 MREC FIXED BIN(15.0).
                  4 MDISP FIXED BIN(15.0).
           2 FILLER CHAR (56);
    DCL (CKEY.DIRKEY.DIRNUM.I.K.M.COUNTRY) FIXED BIN(15.0);
    DCL SYSIN FILE STREAM INPUT;
    DCL SYSPRINT FILE STREAM OUTPUT;
    DCL DAF FILE RECORD DIRECT KEYED ENV(REGIONAL(1));
    DCL INSTR CHAR(80);
    DCL SUBSTR BUILTIN;
    DCL (CODES,GROUP,DIV) FIXED BIN(31.0);
     OPEN FILE(SYSPRINT) STREAM OUTPUT LINESIZE(130);
    ON ENDFILE(SYSIN) BEGIN;
       PUT SKIP FILE(SYSPRINT) EDIT
         ('***ERROR - END OF FILE SYSIN ENCOUNTERED***') (A);
       RCODE=-1;
       GOTO EOJ;
    END;
    PUT SKIP FILE(SYSPRINT) EDIT('+++LIST DIRECTORY PROGRAM+++') (A);
    RCODE=1;
    CKEY=0;
    READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
    GETCARD: GET FILE(SYSIN) EDIT(INSTR) (COL(1).A(80));
    IF SUBSTR(INSTR.1.16)='++END OF COMMAND' THEN GOTO EOJ;
      DIV=1;
       IF SUBSTR(INSTR.10.2)='00' THEN DIV=100;
       IF SUBSTR(INSTR.8.4)='0000' THEN DIV=10000;
       IF SUBSTR(INSTR.6.6)='000000' THEN DIV=1000000;
       IF SUBSTR(INSTR.4.8)='00000000'THEN DIV=100000000;
    GET STRING(INSTR) EDIT (CODES) (X(1).F(10.0));
    GET STRING(INSTR) EDIT (COUNTRY) (X(1).F(2,0));
    DIRKEY.DIRNUM.I.K.M=0;
    GROUP=CODES/DIV;
    DO I = 1 TO CONTROL.NUMONE;
      IF COUNTRY=CONTROL.ONE(I).CODEINUM THEN DO;
         DIRKEY=CONTROL.ONE(I).RECNUM;
         DIRNUM=CONTROL.ONE(I).NUMDIRS;
         PUT PAGE FILE(SYSPRINT) EDIT (' THE COUNTRY '.CONTROL.ONE(I).NAME.
           'HAS '.CONTROL.ONE(I).NUMDIRS.' DIRECTORY ENTRIES ON RECORD '.
            DIRKEY) (A.A(24).A.F(4.0).A.F(4.0));
      END;
    END;
    IF DIRKEY>0 THEN DO;
           PUT SKIP(4) FILE(SYSPRINT) EDIT ('LEVEL'.'CODE'.'NAME'.'LATITUDE'
             .'LONGITUDE'.'PARENT'.'BROTHER'.'CHILD'.'DATA'.'MODELS')
              (A.X(1).A.X(2).A.X(19).A.X(1).A.X(2).A.X(4).A.X(2).A.X(5).A.
              X(5).A);
      READ FILE(DAF) INTO(DIRX) KEY(DIRKEY);
      DO K = 1 TO DIRNUM;
        M=M+1;
        IF M=85 THEN DO;
           M=1;
           DIRKEY=DIRKEY+1;
           READ FILE(DAF) INTO(DIRX) KEY(DIRKEY);
        END;
       IF GROUP=FLOOR(DIR(M).LEVCODE/DIV) THEN PUT SKIP FILE(SYSPRINT) EDIT
            (DIR(M).LEVNUM.DIR(M).CODENUMB.DIR(M).DIRNAME.DIR(M).LAT.
            DIR(M).LON.DIR(M).PREC.DIR(M).PDISP.DIR(M).BREC.DIR(M).BDISP.
            DIR(M).CREC.DIR(M).CDISP.DIR(M).DREC.DIR(M).DDISP.
            ( DIR(M).MODEL(J).MREC.DIR(M).MODEL(J).MDISP  DO J=1 TO 4))
            (F(3.0).F(5.0).X(4).A(24).P'---9.9'.X(3).P'---9.9'.
            8 F(5.0).X(1).4(F(4.0).F(5.0)));
      END;
    END;
    ELSE PUT SKIP FILE(SYSPRINT)
      EDIT('COUNTRY HAS NO DIRECTORY ENTRY') (A);
    GOTO GETCARD;
    EOJ: END YESLS04;
```

3.3.15 LISTING DATA IN THE DATA BASE (LISTJOB)

LISTJOB is provided to list data in the data base.

3.3.15.1 Linkages

None.

3.3.15.2 Interfaces

Data and control block entries must exist for the countries requested.

3.3.15.3 Inputs

Request for data cards on a country basis.

3.3.15.4 Outputs

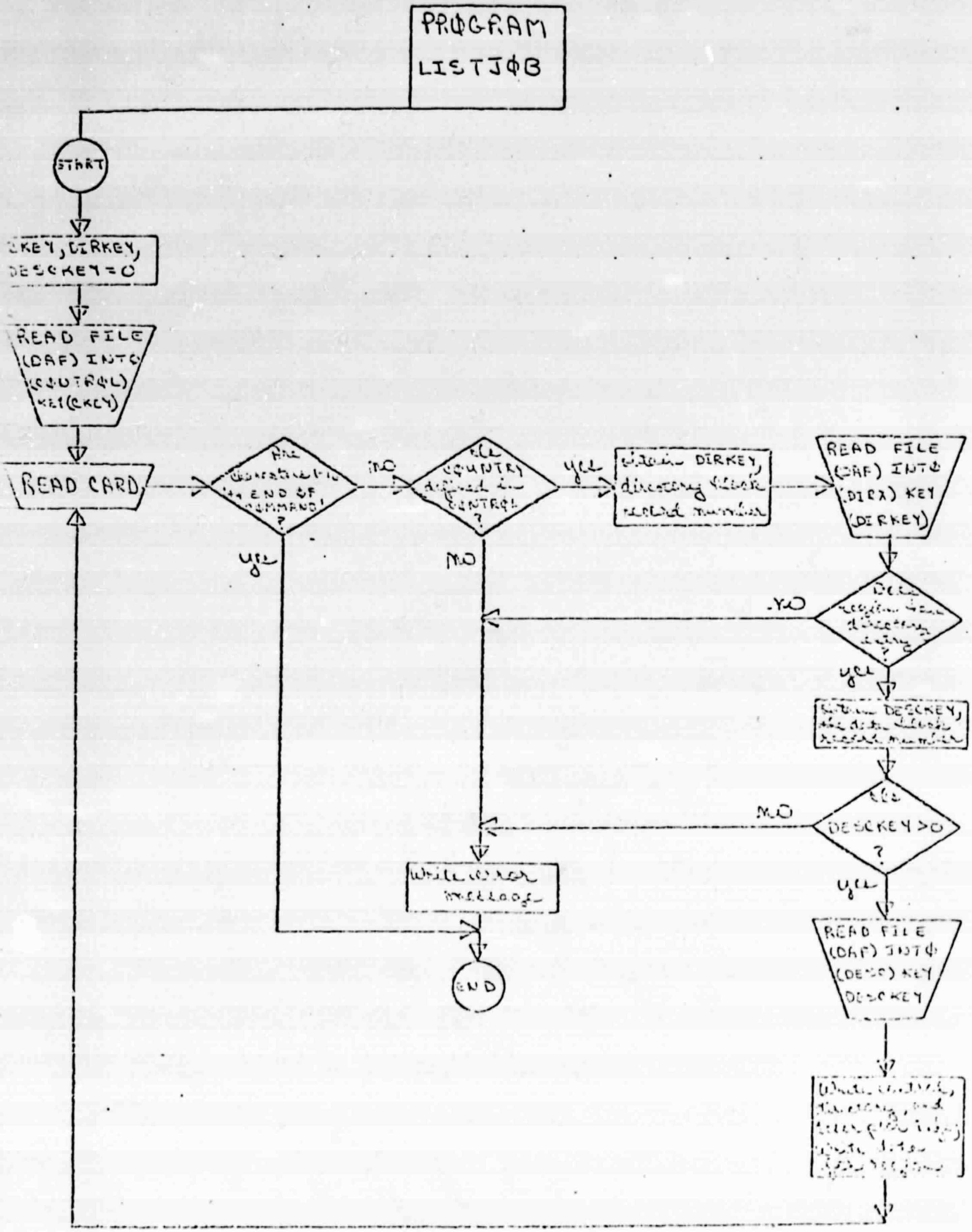
Listings of data by country.

3.3.15.5 Flow Chart

Next page.

3.3.15.6 Listing

Follows flow chart.


PROGRAM LISTJOB
START
CKEY, DIRKEY, DESCKEY = 0
READ FILE (DAF) INTO (CONTROL) KEY (CKEY)
READ CARD
Are characters the END OF COMMAND ?
NO
yes
Is COUNTRY defined in CONTROL ?
yes
NO
Obtain DIRKEY, directory block record number
READ FILE (DAF) INTO (DIRX) KEY (DIRKEY)
Does region have directory entry ?
NO
yes
Obtain DESCKEY, the data block record number
Is DESCKEY >0 ?
NO
yes
READ FILE (DAF) INTO (DESC) KEY DESCKEY
Write control, directory, and [illegible] [illegible] [illegible]
Write error message
END

```
RUN NO. 15          DATE  11/12/76       TIME  0910                 LISTING OF MODULE LISTDATA

DESCRIPTION                 LIST DATA BASE PGM

MASTER FILE                 W.EDS.CCEA.LEC.LIBR
ADDED TO MASTER             10/13/76
LAST DATE COPIED              NONE
LAST UPDATE                   NONE

PASSWORD                    DZXZ
PROGRAMMER                  LEC
LANGUAGE                    PLI
PROC PARAMETER              $NOJCL

 LISTJOB: PROC OPTIONS(MAIN);
 /*                                                                      */
 /*~~THIS PROGRAM IS CALLED BY [illegible]~~*/
 /*                                                                      */
     DCL 1 CONTROL.
             2 FILEID CHAR(8).
             2 NUMPASS FIXED BIN(15,0).
             2 PASS(8) CHAR(8).
             2 NUMLEV FIXED BIN(15.0).
             2 LEVNAME(8) CHAR(24).
             2 NUMCODE FIXED BIN(15.0).
             2 CODE(32).
               3 CODENUM FIXED BIN(15.0).
               3 UNITNUM FIXED BIN(15.0).
               3 BASE FIXED BIN(15.0).
               3 SCALE FIXED BIN(15.0).
               3 CODENAME CHAR(24).
               3 UNITNAME CHAR(24).
             2 NUMONE FIXED BIN(15.0).
             2 ONE(24).
               3 CODEINUM FIXED BIN(15.0).
               3 NUMDIRS FIXED BIN(15.0).
               3 RECNUM FIXED BIN(15.0).
               3 DISPLACE FIXED BIN(15,0).
               3 NAME CHAR(24).
             2 FILERECS FIXED BIN(15.C).
             2 REC(601).
               3 RECTYPE FIXED BIN(15.0).
               3 FRESPACE FIXED BIN(15.0).
               3 LOCATION FIXED BIN(15.0);
     DCL 1 DIRX.
             2 DIR (84).
             3 LEVNUM FIXED BIN(15.0).
             3 CODENUMB FIXED BIN(15.0).
             3 LAT FIXED BIN(15.0).
             3 LON FIXED BIN(15.0).
             3 DIRNAME CHAR(24).
             3 PREC FIXED BIN(15.0).
             3 PDISP FIXED BIN(15.0).
             3 BREC FIXED BIN(15.0).
             3 BDISP FIXED BIN(15.0).
             3 CREC FIXED BIN(15.0).
             3 CDISP FIXED BIN(15.0)).
             3 DREC FIXED BIN(15.0).
             3 DDISP FIXED BIN(15.0).
             3 LEVCODE FIXED BIN(31.0).
             3 MODEL(4).
               4 CROP FIXED BIN(15.0).
               4 YREC FIXED BIN(15.0).
              4 MDISP FIXED BIN(15.0).
          2 FILLER CHAR (56);
   DCL 1 DESCDATA1.
          2 DESC1.
            3 ID FIXED BIN(31.0).
            3 YMO FIXED BIN(31.0),
            3 FILLER1 CHAR(4).
            3 ELEV FIXED BIN(15.)).
            3 TOTBLKS FIXED BIN(15.0).
            3 BLKSUSED FIXED BIN(15,0).
            3 BLKSIZE FIXED BIN(15.0).
            3 FSTREC FIXED BIN(15.0).
            3 FSTDISP FIXED BIN(15.0).
            3 LSTREC FIXED BIN(15.0).
            3 LSTDISP FIXED BIN(15.0).
            3 FILLER2 CHAR(18).
            3 NUMCODE FIXED BIN(15.0).
```

ORIGINAL PAGE IS

```
        3 CODE(12),
          4 CODENUM FIXED BIN(15,0),
          4 NUMELEM FIXED BIN(15,0),
          4 ELEMSIZE FIXED BIN(15,0),
          4 NUMSCODE FIXED BIN(15,0),
          4 SUBCODE(8) FIXED BIN(15,0),
      2 DATA CHAR(6104);
 DCL 1 DESCDATA2,
      2 DATA1 CHAR(3220),
      2 DESC2,
        3 ID FIXED BIN(31,0),
        3 WMO FIXED BIN(31,0),
        3 FILLER1 CHAR(4),
        3 ELEV FIXED BIN(15,0),
        3 TOTBLKS FIXED BIN(15,0),
        3 BLKSUSED FIXED BIN(15,0),
        3 BLKSIZE FIXED BIN(15,0),
        3 FSTREC FIXED BIN(15,0),
        3 FSTDISP FIXED BIN(15,0),
        3 LSTREC FIXED BIN(15,0),
        3 LSTDISP FIXED BIN(15,0),
        3 FILLER2 CHAR(18),
        3 NUMCODE FIXED BIN(15,0),
        3 CODE(12),
          4 CODENUM FIXED BIN(15,0),
          4 NUMELEM FIXED BIN(15,0),
          4 ELEMSIZE FIXED BIN(15,0),
          4 NUMSCODE FIXED BIN(15,0),
          4 SUBCODE(8) FIXED BIN(15,0),
      2 DATA2 CHAR(2884);
 DCL 1 US,
        2 DESC CHAR(336),
        2 DATA(47),
          3 YR FIXED BIN(15,0),
          3 NXTYR FIXED BIN(15,0),
          3 NXTDISP FIXED BIN(15,0),
          3 FILLER1 FIXED BIN(15,0),
          3 TEMP(12) FIXED BIN(15,0),
          3 PRCP(12) FIXED BIN(15,0),
          3 DEGDY(12) FIXED BIN(15,0),
          3 HARV(4) FIXED BIN(31,0),
          3 PLANT(4) FIXED BIN(31,0),
          3 PROD(4) FIXED BIN(31,0),
        2 FILLER2 CHAR(88);
  DCL 1 RUSSIA,
        2 USR(2),
          3 DESC CHAR(336),
          3 DATA(32),
            4 YR FIXED BIN(15,0),
            4 NXTYR FIXED BIN(15,0),
            4 NXTDISP FIXED BIN(15,0),
            4 FILLER FIXED BIN(15,0),
            4 TEMP(12) FIXED BIN(15,0),
            4 PRCP(12) FIXED BIN(15,0),
            4 HARV(4) FIXED BIN(31,0),
            4 PROD(4) FIXED BIN(31,0),
          3 FILL CHAR(68);
  DCL 1 CANADA,
        2 DESC CHAR(336),
        2 DATA(47),
          3 YR FIXED BIN(15,0),
          3 NXTYR FIXED BIN(15,0),
          3 NXTDISP FIXED BIN(15,0),
          3 FILLER1 FIXED BIN(15,0),
          3 MXTP(12) FIXED BIN(15,0),
          3 MNTP(12) FIXED BIN(15,0),
          3 TEMP(12) FIXED BIN(15,0),
          3 PRCP(12) FIXED BIN(15,0),
          3 PLANT(3) FIXED BIN(31,0),
          3 PROD(3) FIXED BIN(31,0),
        2 FILLER2 CHAR(88);
 DCL (CKEY,DIRKEY,DESCKEY,DESCDISP,COUNTRY,DIRNUM) FIXED BIN(15,0);
 DCL (I,J,K,L,M,KK) FIXED BIN(15,0);
 DCL CODES FIXED BIN(31,0);
 DCL SYSIN FILE STREAM INPUT;
 DCL SYSPRINT FILE STREAM OUTPUT;
 DCL DAF FILE RECORD DIRECT KEYED ENV(REGIONAL(1));
 DCL INSTR CHAR(80);
 DCL SUBSTR BUILTIN;
 ON ENDFILE(SYSIN) BEGIN;
   PUT SKIP FILE(SYSPRINT) EDIT
     ('***ERROR - END OF FILE SYSIN ENCOUNTERED***') (A);
   RCODE=-1;
   GOTO EOJ;
 END;
 RCODE=1;
 CKEY=0;
 READ FILE(DAF) INTO(CONTROL) KEY(CKEY);
 GETCARD: GET FILE(SYSIN) EDIT(INSTR) (COL(1),A(80));
```

```
IF SUBSTR(INSTR.1.16)='++END OF COMMAND' THEN GOTO EOJ:
GET STRING(INSTR) EDIT (CODES) (X(1).F(10.0)):
COUNTRY = FLOOR(CODES/100000000):
CKEY.DIRKEY.DESCKEY.DIRNUM=0:
I.J.K.L.M.DESCDISP.KK=0:
DO I = 1 TO CONTROL.NUMONE:
  IF COUNTRY=CONTROL.ONE(I).CODE1NUM THEN DO:
.    DIRKEY=CONTROL.ONE(I).RECNUM:
     DIRNUM=CONTROL.ONE(I).NUMDIRS:
     PUT PAGE FILE(SYSPRINT) EDIT (' THE COUNTRY '.CONTROL.ONE(I).NAME.
       'HAS '.CONTROL.ONE(I).NUMDIRS.' DIRECTORY ENTRIES ON RECORD '.
       DIRKEY) (A.A(24).A.F(4.0).A.F(4.0)):
     PUT SKIP FILE(SYSPRINT) EDIT ('CODES    NAME'.'UNIT'.'BASE SCALE'.
       ( CONTROL.CODE(J).CODENUM.CONTROL.CODE(J).CODENAME.
       CONTROL.CODE(J).UNITNAME.CONTROL.CODE(J).BASE.
       CONTROL.CODE(J).SCALE  DO J=1 TO CONTROL.NUMCODE)) (SKIP.X(5).A.
       X(21).A.X(19).A.SKIP.32(F(9.0).X(2).2 A(24).
       2 F(5.0).SKIP)):
  END:
END:
IF DIRKEY>0 THEN DO:
  READ FILE(DAF) INTO(DIRX) KEY(DIRKEY):
  DO K = 1 TO DIRNUM:
    M=M+1:
     IF M=85 THEN DO:
       M=1:
       DIRKEY=DIRKEY+1:
       READ FILE(DAF) INTO(DIRX) KEY(DIRKEY):
     END:
     IF CODES=DIRX.DIR(M).LEVCODE THEN DO:
       DESCKEY=DIRX.DIR(M).DREC:
       DESCDISP=DIRX.DIR(M).DDISP:
       PUT SKIP(4) FILE(SYSPRINT) EDIT ('LEVEL'.'CODE'.'NAME'.'LATITUDE'
         .'LONGITUDE'.'PARENT'.'BROTHER'.'CHILD'.'DATA'.'MODELS'.
         DIR(M).LEVNUM.DIR(M).CODENUMB.DIR(M).DIRNAME.DIR(M).LAT.
         DIR(M).LON.DIR(M).PREC.DIR(M).PDISP.DIR(M).BREC.DIR(M).BDISP.
         DIR(M).CREC.DIR(M).CDISP.DIR(M).DREC.DIR(M).DDISP.
         ( DIR(M).MODEL(J).MREC.DIR(M).MODEL(J).MDISP  DO J=1 TO 4))
         (A.X(1).A.X(2).A.X(19).A.X(1).A.X(2).A.X(4).A.X(2).A.X(6).A.
         X(5).A.SKIP.F(3.0).F(5.0).X(4).A(24).P'---9.9'.X(3).P'---9.9'.
         8 F(5.0).X(1).4(F(4.0).F(5.0))):
     END:
   END:
   IF DESCKEY>0 THEN DO:
    PUT SKIP(4) FILE(SYSPRINT) EDIT ('DATA DESCRIPTOR IS ON RECORD'.
       DESCKEY.'ID'.'WMO'.'ELEV'.'YR.ALLOC'.'YR.USED'.'BLKSIZE'.
       'FSTREC'.'LSTREC'.'#CODE')
       (A.F(4.0).SKIP(2).X(3).A.X(6).4(A.X(1)).A.X(2).A.X(4).A.X(3).A):
     IF DESCDISP=1 THEN READ FILE(DAF) INTO(DESCDATA1) KEY(DESCKEY):
     IF DESCDISP=3221 THEN DO:
       READ FILE(DAF) INTO(DESCDATA2) KEY(DESCKEY):
       DESCDATA1.DESC1=DESCDATA2.DESC2. BY NAME:
     END:
     PUT SKIP FILE(SYSPRINT) EDIT (DESC1.ID.DESC1.WMO.DESC1.ELEV.
       DESC1.TOTBLKS.DESC1.BLKSUSED.DESC1.BLKSIZE.DESC1.FSTREC.
       DESC1.FSTDISP.DESC1.LSTREC.DESC1.LSTDISP.DESC1.NUMCODE.
       '  DATA CODES   SUBCODE NUMBERS'.(DESC1.CODE(J).CODENUM.
       (DESC1.CODE(J).SUBCODE(L)  DO L=1 TO 8) DO J=1 TO 12))
       (F(11.0).2 F(5.0).2 F(7.0).F(9.0).X(1).4 F(5.0).F(6.0). SKIP(2).
       X(5).A. SKIP. 12(F(13.0).X(6).8 F(4.0).SKIP)):
      PUT PAGE:
      IF COUNTRY=5 THEN DO:
        READ FILE(DAF) INTO(CANADA) KEY(DESCKEY):
        DO L = 1 TO DESC1.BLKSUSED:
        KK=KK+1:
          PUT SKIP(2) FILE(SYSPRINT) EDIT ('YEAR'.'NAT.YR.REC'.
          'NAT.YR.DISP'.CANADA.DATA(L).YR.CANADA.DATA(L).NXTYR.
          CANADA.DATA(L).NXTDISP.
          'CROP CODE             '.(DESC1.CODE(8).SUBCODE(J) DO J=1 TO 3).
          'PLANTED(HECTARES)     '.(CANADA.DATA(L).PLANT(J) DO J=1 TO 3).
          'PRODUCTION(QUINTALS)'.(CANADA.DATA(L).PROD(J) DO J=1 TO 3).
          'JAN     FEB     MARCH    APRIL    MAY     JUNE      JULY      AUG     SEPT '.
          '     OCT      NOV       DEC'.
          'MAX TEMP(CENTIGRADE)  '.(CANADA.DATA(L).MXTP(J) DO J=1 TO 12).
          'MIN. TEMP(CENTIGRADE) '.(CANADA.DATA(L).MNTP(J) DO J=1 TO 12).
          'MEAN TEMP(CENTIGRADE)'.(CANADA.DATA(L).TEMP(J) DO J=1 TO 12).
          'PRECIP(MILLIMETERS)   '.(CANADA.DATA(L).PRCP(J) DO J=1 TO 12))
          (2(A.X(2)).A. SKIP. F(4.0).X(5).F(3.0).X(4).F(4.0). SKIP(2).
          3(X(10).A.X(7).3 F(10.0). SKIP ).SKIP.X(34).A.A.SKIP.
          3(X(10).A.12 P'---9.9'.SKIP).X(10).A.12 F(7.0)):
        IF KK=4 THEN DO:
          PUT PAGE FILE(SYSPRINT):
          KK=0:
        END:
            END:
          END:
        ELSE IF COUNTRY=8 THEN DO:
            READ FILE(DAF) INTO(RUSSIA) KEY(DESCKEY):
            IF DESCDISP=1 THEN M=1:
            IF DESCDISP=1221 THEN M=2:
            DO L = 1 TO DESC1.BLKSUSED:
            KK=KK+1:
              PUT SKIP(2) FILE(SYSPRINT) EDIT ('YEAR'.'NXT.YR.REC'.
              'NXT.YR.DISP'.USR(M).DATA(L).YR.USR(M).DATA(L).NXTYR.
              USR(M).DATA(L).NXTDISP.
              'CROP CODE              '.(DESC1.CODE(6).SUBCODE(J) DO J=1 TO 4).
              'HARVESTED(HECTARES)  '.(USR(M).DATA(L).HARV(J) DO J=1 TO 4).
              'PRODUCTION(QUINTALS)'.(USR(M).DATA(L).PROD(J) DO J=1 TO 4).
              'JAN     FEB    MARCH  APRIL     MAY     JUNE     JULY      AUG     SEPT '.
              '     OCT      NOV      DEC'.
              'MEAN TEMP(CENTIGRADE)'.(USR(M).DATA(L).TEMP(J) DO J=1 TO 12).
              'PRECIP(MILLIMETERS)   '.(USR(M).DATA(L).PRCP(J) DO J=1 TO 12))
              (2(A.X(2)).A. SKIP. F(4.0).X(5).F(3.0).X(9).F(4.0). SKIP(2).
              3(X(10).A.X(7).4 F(10.0). SKIP). SKIP. X(34).A.A. SKIP.
              X(10).A.12 P'----9.9'.SKIP. X(10).A.12 F(7.0)):
            IF KK=4 THEN DO:
              PUT PAGE FILE(SYSPRINT):
              KK=0:
            END:
          END:
        END:
        ELSE IF COUNTRY=3 THEN DO:
            READ FILE(DAF) INTO(US) KEY(DESCKEY):
          DO L=1 TO DESC1.BLKSUSED:
            KK=KK+1:
              PUT SKIP(2) FILE(SYSPRINT) EDIT ('YEAR'.'NXT.YR.REC'.
              'NXT.YR.DISP'.US.DATA(L).YR.US.DATA(L).NXTYR.
              US.DATA(L).NXTDISP.
              'CROP CODE              '.(DESC1.CODE(7).SUBCODE(J) DO J=1 TO 4).
              'HARVESTED(HECTARES)  '.(US.DATA(L).HARV(J) DO J=1 TO 4).
              'PLANTED(HECTARES)     '.(US.DATA(L).PLANT(J) DO J=1 TO 4).
              'PRODUCTION(QUINTALS)'.(US.DATA(L).PROD(J) DO J=1 TO 4).
              'JAN     FEB    MARCH  APRIL     MAY     JUNE     JULY      AUG     SEPT '.
              '     OCT      NOV      DEC'.
              'MEAN TEMP(CENTIGRADE)'.(US.DATA(L).TEMP(J) DO J=1 TO 12).
              'PRECIP(MILLIMETERS)   '.(US.DATA(L).PRCP(J) DO J=1 TO 12).
              'DEGREE DAYS ABOVE     '.(US.DATA(L).DEGDY(J) DO J=1 TO 12))
              (2(A.X(2)).A. SKIP. F(4.0).X(5).F(3.0).X(9).F(4.0). SKIP(2).
              4(X(10).A.X(7).4 F(10.0). SKIP). SKIP. X(34).A.A. SKIP.
              X(10).A.12 P'----9.9'.SKIP.X(10).A.12 F(7.0).SKIP,
              X(10).A.12 P'----9.9'):
            IF KK=4 THEN DO:
              PUT PAGE FILE(SYSPRINT):
              KK=0:
            END:
          END:
        END:
        ELSE PUT SKIP(2) FILE(SYSPRINT) EDIT('UNDEFINED COUNTRY') (A):
      END:
      ELSE PUT SKIP(2) FILE(SYSPRINT) EDIT ('ENTRY HAS NO DATA') (A):
    END:
    ELSE PUT SKIP(2) FILE(SYSPRINT)
      EDIT('COUNTRY HAS NO DIRECTORY ENTRY') (A):
    GOTO GETCARD:
    EOJ: END LISTJOB:
```

4. OPERATION

This section describes the operation of each of the monthly yield data base support programs.

4.1 OPERATING INSTRUCTIONS

There are four types of programs run in maintaining and using the YES Yield Monthly Data Base: data base initialization, data base definition, data base load and update and listing.

4.1.1 DATA BASE INITIALIZATION

The file initialization program is the first program to be run in setting up the data base. This program defines the first record of the file to be the control block and all other records as blank. It also sets the variables in the control block to some dummy values which will be changed in subsequent programs to accommodate the actual situation. One variable, the number of records contained in the file excluding the control block, is dependent on the user's facilities and must be filled into the program before it is run.

4.1.2 DATA BASE DEFINITION

Definition of the data base involves establishing the control block, defining the directories, and entering the data definitions. An example run setup is given in appendix C.

4.1.2.1 Definition of the Control Block

Definition of the control block is the second step in the creation of the data base.

1. Not all sections and subsections must be defined by the user. The file initialization program sets all variables to standard values and some of these values should be changed only when they are automatically modified during execution of other programs.

These include the number of level-one entries, the information about level-one entries, the number of records on the file, and the information about each of the records on the file; these are sections 8, 9, 10, and 11, respectively. All other sections should be defined.

2. Information is read in on cards with only one section or only one subsection of a section on a card.

3. Names are punched left-justified, or starting in the leftmost column of the field, and numbers are punched right-justified.

4. The section number must be punched in columns 6 and 7, and the subsection number in columns 9 and 10. Zero is used if the section has no subsection.

5. Since each section contains different types of information, the formats in which they are entered must also change.

   a. For sections 2, 4, 6, 8, and 10, the appropriate number is punched in the field of columns 12 to 15.

   b. For section 1, the file identification name is punched in the field of columns 12 to 19.

   c. For all subsections of section 3, the password is punched in the field of columns 12 to 19. Note that once sections 2 and 3 are defined, subsequent programs accessing the file will require a password card.

   d. For all subsections of section 5, the level name is punched in the field of columns 12 to 35.

   e. For all subsections of section 7, the code number, unit number, base, scale, code name, and unit name should be punched in the field of columns 12 to 15, 17 to 20, 22 to 25, 27 to 30, 32 to 55, and 57 to 80, respectively.

   f. For all subsections of section 9, the code number, number of directories, record number, displacement, and name should be punched in the field of columns 12 to 15, 17 to 20, 22 to 25, 27 to 30, and 32 to 55, respectively.

g. For all subsections of section 11, the record type and amount and location of free space should be punched in the field of columns 12 to 15, 17 to 20, and 22 to 25, respectively.

Updating the control block is done in two ways, manually by the user or automatically with the other programs.

1. The manual update of the control block is done with the same program that was used for defining it. Consequently, the same formats for each of the sections and/or subsections are followed. Any section and/or subsection can be changed using the program but only sections 1 through 7 or section 10 should ever need to be changed. For the subsections of sections 7, 9, and 11, all the variables must be punched on the card even if some values remain the same; if a variable's field is empty, it will be coded as blank on the file.

2. The automatic update of the control block is done by the other programs which add information to the file. The sections 8 and 9 are changed when the directory block for a new level-one region is defined. Whenever a block on the file is read into for the first time, section 11 is changed to show which type of information was read in; also whenever information is added in a block, section 11 is changed to show the amount of free space remaining.

4.1.2.2 Defining the Directories

Definition of the directory entries in the directory block, or blocks, is the third step in the creation of the data base.

1. Information for each directory entry is read in on one card.

2. Names are punched left-justified and numbers are punched right-justified in their appropriate field of columns.

3. Sections 1 through 5 and section 14 are punched in columns 3 to 4, 5 to 8, 10 to 14, 15 to 19, 21 to 44, and 71 to 80, respectively. These are the level number, code number, latitude, longitude, entry name and the unique ten-digit code.

4. Sections 12 and 13 are defined during execution of the program which defines the data descriptor entries, and section 15 is defined during execution of the program which defines the model definition blocks; no user definition is required.

5. Sections 6 through 11 are defined with the define directory program, but some user input is necessary. In the field of columns 45 to 48, the position in the input card deck of the entry's parent is coded. For example, if Colorado, a level-three region, is the third directory entry card, then the entries for the level-four regions in Colorado would have a 3 coded in column 48. Level-one regions would have a negative one coded since they have no parent. In the fields of columns 49 to 52 and 53 to 56 are coded the positions in the card deck of the directory entries corresponding to the entry's brother and child. Negative ones are coded if there is no brother or child.

6. Only directory entries for one level-one region and the higher levels within it can be defined during one execution of the define directory program. The program will be terminated if a second level-one card is encountered.

7. The first input card must be the level-one region's directory entry. If the level of the first card is not one, then the program will be terminated. The remaining cards can be in any order; however, calculations of the parent, brother, and child positions would be facilitated if the entries were kept in sequence.

8. If the define directory program is run twice with the same input cards, then there will be two directory blocks for the same country, and the country will be listed twice in the control block information.

Directory entries are automatically updated by the other programs which add information to the file. Sections 12 and 13 are changed when the data descriptor entries are added to the file. The subsections of section 15 are changed when the model definition blocks are added to the file.

4.1.2.3 Defining the Data Descriptors

Definition of the data descriptor entries must be done before the data can be placed on the file and after the directory entries have been defined.

1. Information for each data descriptor entry is read in on a set of cards, the number of cards dependent on the number of variable codes required for the data.
2. Numbers are punched right-justified in their appropriate field of columns.
3. Sections 1, 2, 5, 6, 8, and 14 are punched in columns 2 to 11, 13 to 17, 19 to 22, 24 to 25, 27 to 30, and 32 to 33, respectively. These are the identification number, WMO number, elevation, total number of years for which data could be defined, length in bytes for storage of one year's data, and the number of codes.
4. The information for each code in section 15 is punched on a separate card. The number of code cards must be equal to the number of codes specified on the first card. The code number, number of elements, element size, and number of subcodes are punched in the field of columns 2 to 4, 6 to 8, 10 to 11, and 13, respectively. The one to eight subcode numbers are punched

in the fields of columns 15 to 17, 19 to 21, 23 to 25, 27 to 29, 31 to 33, 35 to 37, 39 to 41, and 43 to 45, as needed.

5. Sections 3 and 4 are defined during execution of the define descriptor program by copying the information from the region's directory entry. Sections 7 and 9 to 12 are defined during execution of the program which defines the data onto the file. No user definition is required.

6. The entire set of cards is repeated for each data descriptor entry being defined.

Updating the data descriptor entries is done two ways, manually by the user and automatically with the define data programs.

1. The manual update of the descriptor entries is done with the same program that was used for defining the entries. In order to update a particular descriptor entry which is already defined, the entire set of cards used to define that entry is input again with appropriate corrections made. The pointers to the data, sections 9 to 12, are not changed when the define descriptor program is used for update. To add more data descriptor entries to the file, the same format is used to construct the set of cards for each entry and the define descriptor program used again.

2. The automatic update of the data descriptor entries is done during execution of programs which define or update data on the file. The sections involved are 7 and 9 to 12.

4.1.3 ENTERING AND UPDATING DATA

Initial load of data may be done either by the updating program UPDDATA, or the individual country loaders AUSARG, USSR, CANADA, and USA.

4.1.3.1 Entering Data With the Individual Country Loaders

Before the data can be placed on the file the control block, directory entries and data descriptor entries must all be defined.

1. Data for each variable within a certain year and region are entered on separate cards. The cards are grouped by year and sorted chronologically within each region before execution of a define data program.

2. Numbers are punched right-justified in their appropriate field of columns.

3. There are two different formats for entering data, one for meteorological data and one for yield data. Both formats require the year, variable code, and identification number to be punched in the field of columns 4 to 7, 8 to 10, and 71 to 80, respectively.

   a. For a meteorological variable, the data for each of the 12 months are punched in the field of columns 11 to 15, 16 to 20, 21 to 25, 26 to 30, 31 to 35, 36 to 40, 41 to 45, 46 to 50, 51 to 55, 56 to 60, 61 to 65, and 66 to 70. If any of the 12 fields is blank, the value of the variable for that month will be coded as -9999 on the file to indicate a missing value.

   b. For a yield variable, the data for each crop are punched in the field of columns 11 to 20, 21 to 30, 31 to 40, and 41 to 50, as needed. If more than one crop is reported, then it is assumed that the data are organized in ascending order according to crop code. For example, spring wheat with code 201 is punched in the field 11 to 20 and winter wheat with code 202 in the field 21 to 30. If there is only one crop, the value of the yield variable is punched in the field 11 to 20. Extra fields should be left blank.

4.1.3.2 Updating Data

Data are updated by use of the update data program UPDDATA. Update includes changing data for years which already exist on the file and adding data for new years. It does not include adding data for regions which have no data descriptor entry; the data descriptor entry must be defined first.

1. The same formats used for defining data of the meteorological and yield variables are used for updating those data.
2. Cards can be entered in any order, although sorting the cards by year for each region identification number makes the program more efficient.
3. In the case of meteorological data, values of the variable for any month which are left blank will be assigned values of -9999. Therefore, if one month of a year's precipitation data needs to be changed, the values for all months should be coded. In the case of yield data, the same procedure will hold for the values of the variables for the different crops which are missing.
4. When a new year is added to the file, it is not necessary that all variables be defined; one variable card for a year not previously defined is sufficient to initialize space and change all appropriate pointers for the new year. However, values of the undefined variables will be zero rather than the value -9999 which usually denotes a missing value. The -9999 can be assigned for all values of a variable by entering a card for that variable with blanks for all months or crop information.
5. When an old year is updated, only the variable or variables which need changes need input cards. The other variables remain unchanged.

6. The program assumes that enough free space exists in the data block for extra years if they need to be defined. It does not start a new record as a second data block for the region.

7. The program assumes that the variable being updated or defined is one which is already defined in the region's data descriptor entry. It cannot define new variables until their code number, position, and length are put in the data descriptor.

4.1.4 LISTING PROGRAMS

Three listing programs are provided: YESLS02 to list the control block, YESLS04 to list directories, and LISTJOB to list data.

4.1.4.1 Listing the Control Block

To list information in the control block, the program YESLS02 is used. The only input required is a card with '++END OF COMMAND' punched in columns 1 to 16.

4.1.4.2 Listing the Directory Blocks

To list directory information, the program YESLS04 is used. The ten-digit identification code for the appropriate region should be punched in columns 2 to 11. The program will then list the directory entry for that region and all smaller regions within that region. For example, if all the Canadian directory entries are needed, the code 0500000000 is used; if only the Alberta regions are needed, then the code 0502030000 is used. Any number of input code cards can be used, and then all followed by an '++END OF COMMAND' card.

4.1.4.3 Listing the Data Descriptor and Data Blocks

To list descriptor information and data, the program LISTJOB is used. The ten-digit identification code for the appropriate region should be punched in columns 2 to 11. Some control and directory information will be printed as well as data for all available years for that region. Any number of input code cards can be used, and then all followed by an '++END OF COMMAND' card.

APPENDIX A

STRUCTURES

Appendix A: Structures

Control Block
This is the first record on the file and is 6440 bytes long.

```
DCL 1 CONTROL
      2 FILEID          CHAR(8),
      2 NUMPASS         FIXED BIN(15,0),
      2 PASS(8)         CHAR(8),
      2 NUMLEV          FIXED BIN(15,0),
      2 LEVNAME(8)      CHAR(24),
      2 NUMCODE         FIXED BIN(15,0),
      2 CODE(32),
        3 CODENUM       FIXED BIN(15,0),
        3 UNITNUM       FIXED BIN(15,0),
        3 BASE          FIXED BIN(15,0),
        3 SCALE         FIXED BIN(15,0),
        3 CODENAME      CHAR(24),
        3 UNITNAME      CHAR(24),
      2 NUMONE          FIXED BIN(15,0),
      2 ONE(24),
        3 CODE1NUM      FIXED BIN(15,0),
        3 NUMDIRS       FIXED BIN(15,0),
        3 RECNUM        FIXED BIN(15,0),
        3 DISPLACE      FIXED BIN(15,0),
        3 NAME          CHAR(24),
      2 FILERECS        FIXED BIN(15,0),
      2 REC(601),
        3 RECTYPE       FIXED BIN(15,0),
        3 FRESPACE      FIXED BIN(15,0),
        3 LOCATION      FIXED BIN(15,0);
```

Directory Entry

A maximum of 84 directory entries can be placed in a directory block; each entry is 76 bytes long.

```
DCL 1 DIR,
      2 LEVNUM        FIXED BIN(15,0),
      2 CODENUMB      FIXED BIN(15,0),
      2 LAT           FIXED BIN(15,0),
      2 LON           FIXED BIN(15,0),
      2 DIRNAME       FIXED BIN(15,0),
      2 PREC          FIXED BIN(15,0),
      2 PDISP         FIXED BIN(15,0),
      2 BREC          FIXED BIN(15,0),
      2 BDISP         FIXED BIN(15,0),
      2 CREC          FIXED BIN(15,0),
      2 CDISP         FIXED BIN(15,0),
      2 DREC          FIXED BIN(15 0),
      2 DDISP         FIXED BIN(15,0),
      2 LEVCODE       FIXED BIN(31,0),
      2 MODEL(4),
        3 CROP        FIXED BIN(15,0),
        3 MREC        FIXED BIN(15,0),
        3 MDISP       FIXED BIN(15,0);
```

Data Descriptor Entry
This precedes the data for each region in the data blocks; it is 336 bytes long.

```
DCL 1 DESC,
        2 ID             FIXED BIN(31,0),
        2 WMO            FIXED BIN(31,0),
        2 LATI           FIXED BIN(15,0),
        2 LONG           FIXED BIN(15,0),
        2 ELEV           FIXED BIN(15,0),
        2 TOTBLKS        FIXED BIN(15,0),
        2 NUMBYRS        FIXED BIN(15.0),
        2 BLKSIZE        FIXED BIN(15,0),
        2 FSTRECNO       FIXED BIN(15,0),
        2 FSTDISP        FIXED BIN(15,0),
        2 LSTRECNO       FIXED BIN(15,0),
        2 LSTDISP        FIXED BIN(15,0),
        2 RESERVED       CHAR(18),
        2 NUMBCODE       FIXED BIN(15,0),
        2 DCODE(12),
          3 CODENUMB     FIXED BIN(15,0),
          3 NUMSELEM     FIXED BIN(15,0),
          3 ELEMSIZE     FIXED BIN(15,0),
          3 NUMSCODE     FIXED BIN(15,0),
          3 SUBCODE(8)   FIXED BIN(15,0);
```

Australia Data Year Entry
There is a maximum of 47 years following the data descriptor entry in a data block for each Australian region; each year entry is 128 bytes long.

```
DCL 1 AUSTRALIA,
      2 YEAR             FIXED BIN(15,0),
      2 NXTYRREC         FIXED BIN(15 0),
      2 NXTYRDISP        FIXED BIN(15,0),
      2 FILLER(17)       FIXED BIN(15,0),
      2 MEANTEMP(12)     FIXED BIN(15,0),
      2 PRECIP(12)       FIXED BIN(15,0),
      2 Z(12)            FIXED BIN(15,0),
      2 PRODUCTION(2)    FIXED BIN(31,0),
      2 HARVESTED(2)     FIXED BIN(31,0);
```

Canada Data Year Entry
There is a maximum of 47 years following the data descriptor entry in a data block for each Canadian region; each year entry is 128 bytes long.

```
DCL 1 CANADA,
      2 YEAR             FIXED BIN(15,0),
      2 NXTYRREC         FIXED BIN(15,0),
      2 NXTRYDISP        FIXED BIN(15,0),
      2 FILLER           FIXED BIN(15,0),
      2 MAXTEMP(12)      FIXED BIN(15,0),
      2 MINTEMP(12)      FIXED BIN(15,0),
      2 MEANTEMP(12)     FIXED BIN(15,0),
      2 PRECIP(12)       FIXED BIN(15,0),
      2 PLANTED(3)       FIXED BIN(31,0),
      2 PRODUCTION(3)    FIXED BIN(31,0);
```

U.S.S.R. Data Year Entry
There is a maximum of 22 years following the data descriptor entry for each Russian region in a data block; the data for two regions can be placed in each data block. Each year entry is 88 bytes long.

```
DCL 1 USSR,
      2 YEAR             FIXED BIN(15,0),
      2 NXTYRREC         FIXED BIN(15,0),
      2 NXTYRDISP        FIXED BIN(15,0),
      2 FILLER           FIXED BIN(15,0),
      2 MEANTEMP(12)     FIXED BIN(15,0),
      2 PRECIP(12)       FIXED BIN(15,0),
      2 HARVESTED(4)     FIXED BIN(31,0),
      2 PRODUCTION(4)    FIXED BIN(31,0);
```

United States Data Year Entry
There is a maximum of 47 years following the data descriptor entry in a data block for each United States region; each year entry is 128 bytes long.

```
DCL 1 US,
      2 YEAR             FIXED BIN(15.0),
      2 NXTYRREC         FIXED BIN(15,0),
      2 NXTYRDISP        FIXED BIN(15,0),
      2 FILLER           FIXED BIN(15,0),
      2 MEANTEMP(12)     FIXED BIN(15,0),
      2 PRECIP(12)       FIXED BIN(15,0),
      2 DEGREEDAY(12)    FIXED BIN(15,0),
      2 HARVESTED(4)     FIXED BIN(31,0),
      2 PLANTED(4)       FIXED BIN(31,0),
      2 PRODUCTION(4)    FIXED BIN(31,0);
```

APPENDIX B

VARIABLE CODES

Appendix B: Variables Codes

Meteorological Variables	
Precipitation	5
Maximum Temperature	15
Minimum Temperature	25
Mean Temperature	35
Degree Days Above	40
Degree Days Below	50
Palmer Drought Z-Index	45
Yield Variables	
Harvested	101
Planted	102
Production	103
Harvested Yield	104
Planted Yield	105
Crops	
Spring Wheat	201
Winter Wheat	202
Rice	206
Corn	211
Soybeans	216
Sorghum	221
Flax	226
Unit of Measurement	
Inches	102
Bushels	128
Acres	136
Degrees Fahrenheit	141
Bushels/Acre	151
Millimeters	201
Quintals	228
Hectares	236
Degrees Centigrade	241
Quintals/Hectare	251
Monthly	5
Others	
Hourly	1
3-hourly	3
6-hourly	6
Daily	11
Weekly	16
Monthly	26
Year	61
Pointer	90
Record Pointer	91
Displacement Pointer	92
Filler or Reserved Space	99

ORIGINAL PAGE IS
OF POOR QUALITY

APPENDIX C

SAMPLE INPUT TO YESM001

```
RUN NO. 7          DATE  10/28/76     TIME  1106            LISTING OF MODULE DHDATA

DESCRIPTION            DEFINE CNTRL&DIR&DESC/CANARUS

MASTER FILE            W.EOS.CCEA.LEC.LIBR
ADDED TO MASTER        10/28/76
LAST DATE COPIED         NONE
LAST UPDATE              NONE

PASSWORD               VDCC
PROGRAMMER             LFC
LANGUAGE               DAT
PROC PARAMETER         $NOJCL

   //OCOOK JOB (*DD1000BETHEA  *.*COLUM*).COOK.REGION=160K.TIME=1                                     000010
   //   EXEC PGM=YESM001                                                                                000020
   //STEPLIB DD DSN=W.EDS.CCEA.LEC.LOAD.DISP=SHR                                                       000030
   //SYSPRINT DD SYSOUT=A                                                                              000040
   //DAF DD DSN=W.EOS.CCEA.MET.CLIMAT.DISP=SHR.DCB=BUFNO=1                                            000050
   //SYSIN DD *                                                                                        000060
   ++COMMAND=DEFINE     ++OPERAND=CONTROL                                                              000070
         2   0    2                                                                                    000080
         3   1  COOK                                                                                   000090
         3   2  SLEMONS                                                                                000100
         6   0    16                                                                                   000110
         7   1     5   201     2     0 PRECIPITATION             MILLIMETERS                           000120
         7   2    15   241     2    -1 MAXIMUM TEMPERATURE       DEGREES CENTIGRADE                    000130
         7   3    25   241     2    -1 MINIMUM TEMPERATURE       DEGREES CENTIGRADE                    000140
         7   4    35   241     2    -1 MEAN TEMPERATURE          DEGREES CENTIGRADE                    000150
         7   5   101   235     4     0 HARVESTED                 HECTARES                              000160
         7   6   102   236     4     0 PLANTED                   HECTARES                              000170
         7   7   103   228     4     0 PRODUCTION                QUINTALS                              000180
         7   8    50     5     2     0 DEGREE DAYS ABOVE         MONTHLY                               000190
         7   9   201     0     3    12 SPRING WHEAT                                                    000200
         7  10   202     0     3    12 WINTER WHEAT                                                    000210
         7  11    26     0     3     7 MONTHLY                                                         000220
         7  12    61     0     2     0 YEAR                                                            000230
         7  13    90     0     2     0 POINTER                                                         000240
         7  14    91     0     2     0 RECORD POINTER                                                  000250
         7  15    92     0     2     0 DISPLACEMENT POINTER                                            000260
         7  16    99     0     2     0 FILLER                                                          000270
   ++END OF COMMAND                                                                                    000280
   ++COMMAND=DEFINE     ++OPERAND=DIRECTORY                                                            000290
       1     5      550 1050 CANADA                       -1   -1    2            500000000   000300
       2     2      550 1050 PRAIRIE PROVINCES             1   -1    3            502000000   000310
       3     3      556 1146 ALBERTA                       2    7    4            502030000   000320
       4     1      515 1128 SOUTHERN                      3    5   -1            502030100   000330
       4     2      555 1113 CENTRAL                       3    6   -1            502030200   000340
       4     3      570 1171 NORTHERN                      3   -1   -1            502030300   000350
       3     4      545 1060 SASKATCHEWAN                  2   17    8            502040000   000360
       4     1      496 1025 SOUTHEASTERN                  7    9   -1            502040100   000370
       4     2      497 1045 REGINA-WEYBURN                7   10   -1            502040200   000380
       4     3      498 1075 SOUTH CENTRAL                 7   11   -1            502040300   000390
       4     4      498 1095 SOUTHWESTERN                  7   12   -1            502040400   000400
       4     5      513 1030 EAST CENTRAL                  7   13   -1            502040500   000410
       4     6      513 1030 CENTRAL                       7   14   -1            502040600   000420
       4     7      515 1089 WEST CENTRAL                  7   15   -1            502040700   000430
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
RUN NO. 7         DATE  10/28/76      TIME  1106            LISTING OF MODULE DBDATA

     4    8    527 1035 NORTHEASTERN                        7   16   -1                        502040800   000440
     4    9    565 1060 NORTHWESTERN                        7   -1   -1                        502040900   000450
     3    5    550  970 MANITOBA                            2   -1   18                        502050000   000460
     4    1    500  960 SOUTHEASTERN                       17   19   -1                        502050100   000470
     4    2    495  976 SOUTH CENTRAL                      17   20   -1                        502050200   000480
     4    3    496  992 SOUTHWESTERN                       17   21   -1                        502050300   000490
     4    4    510  992 NORTHERN                           17   -1   -1                        502050400   000500
  ++END OF COMMAND                                                                                         000510
  ++COMMAND=DEFINE       ++OPERAND=DESCRIPTOR                                                   00000020   000520
   502030100        0      0 47  128   9                                                        00000030   000530
    61    1    2  0                                                                                        000540
    90    2    2  2   91   92                                                                              000550
    99    1    2  0                                                                                        000560
    15   12    2  1   26                                                                            0000   000570
    25   12    2  1   26                                                                            0000   000580
    35   12    2  1   26                                                                            0000   000590
     5   12    2  1   26                                                                            0000   000600
   102    3    4  1  201                                                                        00000080   000610
   103    3    4  1  201                                                                        00000090   000620
   502030200        0      0 47  128   9                                                        00000100   000630
    61    1    2  0                                                                                        000640
    90    2    2  2   91   92                                                                              000650
    99    1    2  0                                                                                        000660
    15   12    2  1   26                                                                            0000   000670
    25   12    2  1   26                                                                            0000   000680
    35   12    2  1   26                                                                            0000   000690
     5   12    2  1   26                                                                            0000   000700
   102    3    4  1  201                                                                        00000150   000710
   103    3    4  1  201                                                                        00000160   000720
   502030300        0      0 47  128   9                                                        00000170   000730
    61    1    2  0                                                                                        000740
    90    2    2  2   91   92                                                                              000750
    99    1    2  0                                                                                        000760
    15   12    2  1   26                                                                            0000   000770
    25   12    2  1   26                                                                            0000   000780
    35   12    2  1   26                                                                            0000   000790
     5   12    2  1   26                                                                            0000   000800
   102    3    4  1  201                                                                        00000220   000810
   103    3    4  1  201                                                                        00000230   000820
   502040100        0      0 47  128   9                                                        00000240   000830
    61    1    2  0                                                                                        000840
    90    2    2  2   91   92                                                                              000850
    99    1    2  0                                                                                        000860
    15   12    2  1   26                                                                            0000   000870
    25   12    2  1   26                                                                            0000   000880
    35   12    2  1   26                                                                            0000   000890
     5   12    2  1   26                                                                            0000   000900
   102    3    4  1  201                                                                        00000290   000910
   103    3    4  1  201                                                                        00000300   000920
   502040200        0      0 47  128   9                                                        00000310   000930
    61    1    2  0                                                                                        000940
    90    2    2  2   91   92                                                                              000950
    99    1    2  0                                                                                        000960
    15   12    2  1   26                                                                            0000   000970
    25   12    2  1   26                                                                            0000   000980
    35   12    2  1   26                                                                            0000   000990
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
RUN NO. 1          DATE  10/28/76     TIME  1106            LISTING OF MODULE DMDATA
   45   12   2  1   26                                                          0000   001000
  102    3   4  1  201                                                      00000360   001010
  103    3   4  1  201                                                      00000370   001020
   502040300         0      0 47   128   9                                  00000380   001030
   61    1   2  0                                                                      001040
   90    2   2  2   91   92                                                            001050
   99    1   2  0                                                                      001060
   15   12   2  1   26                                                          0000   001070
   25   12   2  1   26                                                          0000   001080
   35   12   2  1   26                                                          0000   001090
   45   12   2  1   26                                                          0000   001100
  102    3   4  1  201                                                      00000430   001110
  103    3   4  1  201                                                      00000440   001120
   502040400         0      0 47   128   9                                  00000450   001130
   61    1   2  0                                                                      001140
   90    2   2  2   91   92                                                            001150
   99    1   2  0                                                                      001160
   15   12   2  1   26                                                          0000   001170
   25   12   2  1   26                                                          0000   001180
   35   12   2  1   26                                                          0000   001190
   45   12   2  1   26                                                          0000   001200
  102    3   4  1  201                                                      00000500   001210
  103    3   4  1  201                                                      00000510   001220
   502040500         0      0 47   128   9                                  00000520   001230
   61    1   2  0                                                                      001240
   99    2   2  2   91   92                                                            001250
   99    1   2  0                                                                      001260
   15   12   2  1   26                                                          0000   001270
   25   12   2  1   26                                                          0000   001280
   35   12   2  1   26                                                          0000   001290
   45   12   2  1   26                                                          0000   001300
  102    3   4  1  201                                                      00000570   001310
  103    3   4  1  201                                                      00000580   001320
   502040600         0      0 47   128   9                                  00000590   001330
   61    1   2  0                                                                      001340
   90    2   2  2   91   92                                                            001350
   99    1   2  0                                                                      001360
   15   12   2  1   26                                                          0000   001370
   25   12   2  1   26                                                          0000   001380
   35   12   2  1   26                                                          0000   001390
   45   12   2  1   26                                                          0000   001400
  102    3   4  1  201                                                      00000640   001410
  103    3   4  1  201                                                      00000650   001420
   502040700         0      0 47   128   9                                  00000660   001430
   61    1   2  0                                                                      001440
   90    2   2  2   91   92                                                            001450
   99    1   2  0                                                                      001460
   15   12   2  1   26                                                          0000   001470
   25   12   2  1   26                                                          0000   001480
   35   12   2  1   26                                                          0000   001490
   45   12   2  1   26                                                          0000   001500
  102    3   4  1  201                                                      00000710   001510
  103    3   4  1  201                                                      00000720   001520
   502040800         0      0 47   128   9                                  00000730   001530
   61    1   2  0                                                                      001540
   90    2   2  2   91   92                                                            001550
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
RUN NO. 7          DATE  10/28/76     TIME  1106               LISTING OF MODULE DBDATA
  99    1   2 0                                             001560
  15   12   2 1   26                                  0000  001570
  25   12   2 1   26                                  0000  001580
  35   12   2 1   26                                  0000  001590
   5   12   2 1   26                                  0000  001600
 102    3   4 1  201                              00000780  001610
 103    3   4 1  201                              00000790  001620
  502040900       0     0 47  128  9              00000800  001630
  61    1   2 0                                             001640
  90    2   2 2   91   92                                   001650
  99    1   2 0                                             001660
  15   12   2 1   26                                  0000  001670
  25   12   2 1   26                                  0000  001680
  35   12   2 1   26                                  0000  001690
   5   12   2 1   26                                  0000  001700
 102    3   4 1  201                              00000850  001710
 103    3   4 1  201                              00000860  001720
  502050100       0     0 47  128  9              00000870  001730
  61    1   2 0                                             001740
  90    2   2 2   91   92                                   001750
  99    1   2 0                                             001760
  15   12   2 1   26                                  0000  001770
  25   12   2 1   26                                  0000  001780
  35   12   2 1   26                                  0000  001790
   5   12   2 1   26                                  0000  001800
 102    3   4 1  201                              00000920  001810
 103    3   4 1  201                              00000930  001820
  502050200       0     0 47  128  9              00000940  001830
  61    1   2 0                                             001840
  90    2   2 2   91   92                                   001850
  99    1   2 0                                             001860
  15   12   2 1   26                                  0000  001870
  25   12   2 1   26                                  0000  001880
  35   12   2 1   26                                  0000  001890
   5   12   2 1   26                                  0000  001900
 102    3   4 1  201                              00000990  001910
 103    3   4 1  201                              00001000  001920
  502050300       0     0 47  128  9              00001010  001930
  61    1   2 0                                             001940
  90    2   2 2   91   92                                   001950
  99    1   2 0                                             001960
  15   12   2 1   26                                  0000  001970
  25   12   2 1   26                                  0000  001980
  35   12   2 1   26                                  0000  001990
   5   12   2 1   26                                  0000  002000
 102    3   4 1  201                              00001060  002010
 103    3   4 1  201                              00001070  002020
  502050400       0     0 47  128  9              00001080  002030
  61    1   2 0                                             002040
  90    2   2 2   91   92                                   002050
  99    1   2 0                                             002060
  15   12   2 1   26                                  0000  002070
  25   12   2 1   26                                  0000  002080
  35   12   2 1   26                                  0000  002090
   5   12   2 1   26                                  0000  002100
 102    3   4 1  201                              00001130  002110
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
  103    3   4 1 201                                                                                00001140   002120
++END OF COMMAND                                                                                    00001150   002130
++COMMAND=DEFINE            ++OPERAND=DIRECTORY                                                     00000000   002140
    1    8     550 -550 USSR                                      -1    -1     2                  800000000   002150
    2    1     501 -359 WESTERN                                    1    12     3                  801000000   002160
    3    1     554 -250 BALTICS                                    2     4    -1                  801010000   002170
    3    2     557 -257 BELORUSSIA                                 2     5    -1                  801020000   002180
    3    3     500 -252 WEST UKRAINE                               2     6    -1                  801030000   002190
    3    4     499 -302 NORTH CENTRAL UKRAINE                      2     7    -1                  801040000   002200
    3    5     500 -350 NORTHEAST UKRAINE                          2     8    -1                  801050000   002210
    3    6     455 -353 EASTERN UKRAINE                            2     9    -1                  801060000   002220
    3    7     453 -303 SOUTHERN UKRAINE                           2    10    -1                  801070000   002230
    3    8     450 -258 MOLDAVIA                                   2    11    -1                  801080000   002240
    3    9     450 -400 KRASNODAR                                  2    -1    -1                  801090000   002250
    2    2     506 -450 CENTRAL                                    1    22    13                  802000000   002260
    3   10     452 -408 NORTHEAST CAUCACUS                        12    14    -1                  802100000   002270
    3   11     504 -353 WEST BLACK SOIL ZONE                      12    15    -1                  802110000   002280
    3   12     504 -402 EAST BLACK SOIL ZONE                      12    16    -1                  802120000   002290
    3   13     551 -355 CENTRAL REGION                            12    17    -1                  802130000   002300
    3   14     552 -454 VOLGA-VYATSK                              12    18    -1                  802140000   002310
    3   15     503 -500 UPPER VOLGA                               12    19    -1                  802150000   002320
    3   16     502 -455 MIDDLE VOLGA                              12    20    -1                  802160000   002330
    3   17     458 -450 LOWER VOLGA                               12    21    -1                  802170000   002340
    3   18     555 -551 NORTHWEST URALS                           12    -1    -1                  802180000   002350
    2    3     551 -701 EAST CENTRAL                               1    -1    23                  803000000   002360
    3   19     506 -552 SOUTHERN URALS                            22    24    -1                  803190000   002370
    3   20     562 -700 NORTHEASTERN URALS                        22    25    -1                  803200000   002380
    3   21     455 -550 WESTERN KAZAKHSTAN                        22    26    -1                  803210000   002390
    3   22     552 -608 KUSTANAY                                  22    27    -1                  803220000   002400
    3   23     502 -558 TSELINOGRAD                               22    28    -1                  803230000   002410
    3   24     506 -659 NORTHERN KAZAKHSTAN                       22    29    -1                  803240000   002420
    3   25     502 -752 PAVLODAR                                  22    30    -1                  803250000   002430
    3   26     552 -800 WESTERN SIBERIA                           22    31    -1                  803260000   002440
    3   27     504 -807 ALTAI KRAY                                22    -1    -1                  803270000   002450
++END OF COMMAND                                                                                    00000000   002460
++COMMAND=DEFINE            ++OPERAND=DESCRIPTOR                                                    00000020   002470
  801010000          0      0 22    88   7                                                          00000030   002480
   81     1     2 6                                                                                 00000040   002490
   90     2     2 2    91   92                                                                      00000050   002500
   99     1     2 6                                                                                 00000060   002510
   35    12     2 1    20                                                                           00000070   002520
    5    12     2 1    20                                                                           00000080   002530
  101     4     4 1   202                                                                           00000090   002540
  103     4     4 1   202                                                                           00000100   002550
  801020000          0      0 22    88   7                                                          00000110   002560
   81     1     2 0                                                                                 00000120   002570
   90     2     2 2    91   92                                                                      00000130   002580
   99     1     2 0                                                                                 00000140   002590
   35    12     2 1    20                                                                           00000150   002600
    5    12     2 1    20                                                                           00000160   002610
  101     4     4 1   202                                                                           00000170   002620
  103     4     4 1   202                                                                           00000180   002630
  801030000          0      0 22    88   7                                                          00000190   002640
   81     1     2 6                                                                                 00000200   002650
   90     2     2 2     9   92                                                                      00000210   002660
   99     1     2 6                                                                                 00000220   002670
```

```
RUN NO. 7          DATE  10/28/76     TIME  1106               LISTING OF MODULE OBDATA

     35   12   2  1   26                                                               00000230   002680
      5   12   2  1   26                                                               00000240   002690
    101    4   4  1  202                                                               00000250   002700
    103    4   4  1  202                                                               00000260   002710
     801040000         0     0 22    88   7                                            00000270   002720
     61    1   2  0                                                                    00000280   002730
     90    2   2  2   91   92                                                          00000290   002740
     99    1   2  0                                                                    00000300   002750
     35   12   2  1   26                                                               00000310   002760
      5   12   2  1   26                                                               00000320   002770
    101    4   4  1  202                                                               00000330   002780
    103    4   4  1  202                                                               00000340   002790
     801050000         0     0 22    88   7                                            00000350   002800
     61    1   2  0                                                                    00000360   002810
     90    2   2  2   91   92                                                          00000370   002820
     99    1   2  0                                                                    00000380   002830
     35   12   2  1   26                                                               00000390   002840
      5   12   2  1   26                                                               00000400   002850
    101    4   4  1  202                                                               00000410   002860
    103    4   4  1  202                                                               00000420   002870
     801060000         0     0 22    88   7                                            00000430   002880
     61    1   2  0                                                                    00000440   002890
     90    2   2  2   91   92                                                          00000450   002900
     99    1   2  0                                                                    00000460   002910
     35   12   2  1   26                                                               00000470   002920
      5   12   2  1   26                                                               00000480   002930
    101    4   4  1  202                                                               00000490   002940
    103    4   4  1  202                                                               00000500   002950
     801070000         0     0 22    88   7                                            00000510   002960
     61    1   2  0                                                                    00000520   002970
     90    2   2  2   91   92                                                          00000530   002980
     99    1   2  0                                                                    00000540   002990
     35   12   2  1   26                                                               00000550   003000
      5   12   2  1   26                                                               00000560   003010
    101    4   4  1  202                                                               00000570   003020
    103    4   4  1  202                                                               00000580   003030
     801080000         0     0 22    88   7                                            00000590   003040
     61    1   2  0                                                                    00000600   003050
     90    2   2  2   91   92                                                          00000610   003060
     99    1   2  0                                                                    00000620   003070
     35   12   2  1   26                                                               00000630   003080
      5   12   2  1   26                                                               00000640   003090
    101    4   4  1  202                                                               00000650   003100
    103    4   4  1  202                                                               00000660   003110
     801090000         0     0 22    88   7                                            00000670   003120
     61    1   2  0                                                                    00000680   003130
     90    2   2  2   91   92                                                          00000690   003140
     99    1   2  0                                                                    00000700   003150
     35   12   2  1   26                                                               00000710   003160
      5   12   2  1   26                                                               00000720   003170
    101    4   4  1  202                                                               00000730   003180
    103    4   4  1  202                                                               00000740   003190
     802100000         0     0 22    88   7                                            00000750   003200
     61    1   2  0                                                                    00000760   003210
     90    2   2  2   91   92                                                          00000770   003220
     99    1   2  0                                                                    00000780   003230
```

```
RUN NO. 7        DATE  10/28/76     TIME  1106          LISTING OF MODULE DBDATA
    35   12   2  1   26                                                     00000790   003240
     5   12   2  1   26                                                     00000800   003250
   101    4   4  2  201  202                                                00000810   003260
   103    4   4  2  201  202                                                00000820   003270
     802110000        0      0 22     88   7                                00000830   003280
    61    1   2  0                                                          00000840   003290
    98    2   2  2   91   92                                                00000850   003300
    99    1   2  0                                                          00000860   003310
    35   12   2  1   26                                                     00000870   003320
     5   12   2  1   26                                                     00000880   003330
   101    4   4  2  201  202                                                00000890   003340
   103    4   4  2  201  202                                                00000900   003350
     802120000        0      0 22     88   7                                00000910   003360
    61    1   2  0                                                          00000920   003370
    98    2   2  2   91   92                                                00000930   003380
    99    1   2  0                                                          00000940   003390
    35   12   2  1   26                                                     00000950   003400
     5   12   2  1   26                                                     00000960   003410
   101    4   4  2  201  202                                                00000970   003420
   103    4   4  2  201  202                                                00000980   003430
     802130000        0      0 22     88   7                                00000990   003440
    61    1   2  0                                                          00001000   003450
    98    2   2  2   91   92                                                00001010   003460
    99    1   2  0                                                          00001020   003470
    35   12   2  1   26                                                     00001030   003480
     5   12   2  1   26                                                     00001040   003490
   101    4   4  2  201  202                                                00001050   003500
   103    4   4  2  201  202                                                00001060   003510
     802140000        0      0 22     88   7                                00001070   003520
    61    1   2  0                                                          00001080   003530
    98    2   2  2   91   92                                                00001090   003540
    99    1   2  0                                                          00001100   003550
    35   12   2  1   26                                                     00001110   003560
     5   12   2  1   26                                                     00001120   003570
   101    4   4  2  201  202                                                00001130   003580
   103    4   4  2  201  202                                                00001140   003590
     802150000        0      0 22     88   7                                00001150   003600
    61    1   2  0                                                          00001160   003610
    98    2   2  2   91   92                                                00001170   003620
    99    1   2  0                                                          00001180   003630
    35   12   2  1   26                                                     00001190   003640
     5   12   2  1   26                                                     00001200   003650
   101    4   4  2  201  202                                                00001210   003660
   103    4   4  2  201  202                                                00001220   003670
     802160000        0      0 22     88   7                                00001230   003680
    61    1   2  0                                                          00001240   003690
    98    2   2  2   91   92                                                00001250   003700
    99    1   2  0                                                          00001260   003710
    35   12   2  1   26                                                     00001270   003720
     5   12   2  1   26                                                     00001280   003730
   101    4   4  2  201  202                                                00001290   003740
   103    4   4  2  201  202                                                00001300   003750
     802170000        0      0 22     88   7                                00001310   003760
    61    1   2  0                                                          00001320   003770
    98    2   2  2   91   92                                                00001330   003780
    99    1   2  0                                                          00001340   003790
```

~~07~~ 119
C-1

```
RUN NO. 7          DATE  10/28/76     TIME  1105              LISTING OF MODULE DHDATA
    35    12   2  1   26                                                                           00001350   003800
     5    12   2  1   26                                                                           00001360   003810
   101     4   4  2  201  202                                                                      00001370   003820
   101     4   4  2  201  202                                                                      00001380   003830
     00318000         0      0 22    88   1                                                        00001390   003840
    61     1   2  0                                                                                00001400   003850
    91     2   2  2   91   92                                                                      00001410   003860
    94     1   2  0                                                                                00001420   003870
    35    12   2  1   26                                                                           00001430   003880
     5    12   2  1   26                                                                           00001440   003890
   101     4   4  2  201  202                                                                      00001450   003900
   101     4   4  2  201  202                                                                      00001460   003910
     00319000         0      0 22    88   1                                                        00001470   003920
    61     1   2  0                                                                                00001480   003930
    91     2   2  2   91   92                                                                      00001490   003940
    94     1   2  0                                                                                00001500   003950
    35    12   2  1   26                                                                           00001510   003960
     5    12   2  1   26                                                                           00001520   003970
   101     4   4  1  201                                                                           00001530   003980
   101     4   4  1  201                                                                           00001540   003990
     00320000         0      0 22    88   1                                                        00001550   004000
    61     1   2  0                                                                                00001560   004010
    91     2   2  2   91   92                                                                      00001570   004020
    94     1   2  0                                                                                00001580   004030
    35    12   2  1   26                                                                           00001590   004040
     5    12   2  1   26                                                                           00001600   004050
   101     4   4  1  201                                                                           00001610   004060
   101     4   4  1  201                                                                           00001620   004070
     00321000         0      0 22    88   1                                                        00001630   004080
    61     1   2  0                                                                                00001640   004090
    91     2   2  2   91   92                                                                      00001650   004100
    94     1   2  0                                                                                00001660   004110
    35    12   2  1   26                                                                           00001670   004120
     5    12   2  1   26                                                                           00001680   004130
   101     4   4  1  201                                                                           00001690   004140
   101     4   4  1  201                                                                           00001700   004150
     00322000         0      0 22    88   1                                                        00001710   004160
    61     1   2  0                                                                                00001720   004170
    91     2   2  2   91   92                                                                      00001730   004180
    94     1   2  0                                                                                00001740   004190
    35    12   2  1   26                                                                           00001750   004200
     5    12   2  1   26                                                                           00001760   004210
   101     4   4  1  201                                                                           00001770   004220
   101     4   4  1  201                                                                           00001780   004230
     00323000         0      0 22    88   1                                                        00001790   004240
    61     1   2  0                                                                                00001800   004250
    91     2   2  2   91   92                                                                      00001810   004260
    94     1   2  0                                                                                00001820   004270
    35    12   2  1   26                                                                           00001830   004280
     5    12   2  1   26                                                                           00001840   004290
   101     4   4  1  201                                                                           00001850   004300
   101     4   4  1  201                                                                           00001860   004310
     00324000         0      0 22    88   1                                                        00001870   004320
    61     1   2  0                                                                                00001880   004330
    91     2   2  2   91   92                                                                      00001890   004340
    94     1   2  0                                                                                00001900   004350
```

```
RUN NO. 7          DATE  10/28/75     TIME  1106              LISTING OF MODULE OBDATA
   35   12   2  1   26                                                   00001910  004360
    5   12   2  1   26                                                   00001920  004370
  101    4   4  1  201                                                   00001930  004380
  103    4   4  1  201                                                   00001940  004390
   403250000        0     0 22   88  1                                   00001950  004400
   91    1   2  0                                                        00001960  004410
   93    2   2  2   91  92                                               00001970  004420
   99    1   2  0                                                        00001980  004430
   35   12   2  1   26                                                   00001990  004440
    5   12   2  1   26                                                   00002000  004450
  101    4   4  1  201                                                   00002010  004460
  103    4   4  1  201                                                   00002020  004470
   403260000        0     0 22   88  1                                   00002030  004480
   91    1   2  0                                                        00002040  004490
   93    2   2  2   91  92                                               00002050  004500
   99    1   2  0                                                        00002060  004510
   35   12   2  1   26                                                   00002070  004520
    5   12   2  1   26                                                   00002080  004530
  101    4   4  1  201                                                   00002090  004540
  103    4   4  1  201                                                   00002100  004550
   403270000        0     0 22   88  1                                   00002110  004560
   91    1   2  0                                                        00002120  004570
   93    2   2  2   91  92                                               00002130  004580
   99    1   2  0                                                        00002140  004590
   35   12   2  1   26                                                   00002150  004600
    5   12   2  1   26                                                   00002160  004610
  101    4   4  1  201                                                   00002170  004620
  103    4   4  1  201                                                   00002180  004630
 ++END OF COMMAND                                                        00002190  004640
 ++END OF JOB                                                            00002200  004650
```

~~C-5~~

C-7

121

ORIGINAL PAGE IS
OF POOR QUALITY

ORIGINAL PAGE IS
OF POOR QUALITY