

FINAL REPORT
A COMPUTER BASED INFORMATION SYSTEM
FOR COUNTY EQUIPMENT COST RECORDS
HR-173

Submitted to the
Iowa Department of Transportation
Highway Division
and the
Iowa Highway Research Board

by

John D. Poyzer

and

John M. Liittschwager

Systems Division, College of Engineering

The University of Iowa

Iowa City, Iowa 52242

July 31, 1975

APPENDIX D - DATA PROCESSING USER'S GUIDE

This appendix is divided into three sections. The first section contains abstracts of each of the eight computer programs in the system, instructions for keypunching the three input documents, and computer operating instructions pertaining to each program. The second section contains system flowcharts for the entire system as well as program flowcharts for each program. The last section contains PL/1 program listings of each program.

PROGRAM ABSTRACTS, KEYPUNCH INSTRUCTIONS,
AND COMPUTER OPERATING INSTRUCTIONS

Program abstracts and computer operating instructions are given in this section for the eight computer programs which make up the information system for county equipment cost records. They are presented in the order in which the programs are to be executed: DIRECT, INDCOST, STCHNGE, UPDATM, MAIN, CTYSUMRY, MFGAGE, and UPDATE. Key punch instructions are also included for input data associated with programs DIRECT, INDCOST and STCHNGE, which are the only programs processing county supplied data.

Abstract for Program DIRECT

1. Purpose: To sort and edit the Direct Cost Summary Cards.
2. Procedure: Program DIRECT is a two-step job. The first step is a utility sort routine which sorts the input cards by county number and equipment number. The second step performs reasonableness checks of the input cards on the county field, the month field, the year field, and the remaining data fields taken collectively. Records without errors are written to a disk output file. Input records containing errors are grouped by county and recorded on printed output.
3. Configuration: IBM 370/145
One Card Reader
One Printer
Six to Eight Cylinders of IBM 3330
Disk Space
4. Source Language: PL/1
5. Limitations: None
6. Running Time: Approximately five minutes CPU time
7. Additional Remarks: After running DIRECT, a check should be made of the error report to eliminate errors associated with keypunching and local data processing. If such errors are found, corrections should be made and DIRECT should be rerun prior to running MAIN. Error listings associated with county input are return-

ed to the counties with the other output from the system.

8. Subroutines: None

Keypunch Instructions for Program DIRECT

Input Document: Direct Cost Summary Form

Card Columns	Field Name	Numeric	Alpha-numeric
1-3	County Number	X	
4-11	Equipment Number		X
12-13	Month	X	
14-15	Year	X	
16-21	Fuel Cost	X	
22-26	Lubricant Cost	X	
27-30	Antifreeze Cost	X	
31-36	Tires and Tubes Cost	X	
37-42	Expendable Parts Cost	X	
43-48	Repair Parts Cost	X	
49-54	Labor Cost	X	
55-59	Mileage or Hours	X	
60-63	Down Time in Hours	X	
64-65	Number of Times Repaired	X	

Special Instructions: Use left zero capability on all numeric fields when setting up drum card. A sample Direct Cost Summary Form appears on the next page.

DIRECT COST SUMMARY FORM

COUNTY NAME _____

DATE _____

COUNTY NUMBER	EQUIPMENT NUMBER	DATE MM YY	FUEL COST	LUBR. COST	ANTI-FREEZE	TIRES, TUBES	EXPENDABLE PARTS	REPAIR PARTS	LABOR COST	MILES/HOURS	DOWN TIME	TIMES REPAIRED
1	4	12 14	16	22	27	31	37	43	49	55	60	64

This section consists of 12 horizontal lines, each containing a grid of vertical tick marks. These lines are intended for data entry, corresponding to the columns defined in the header above.

Computer Operating Instructions for Program DIRECT

Frequency of Run: Annually

Region: 100K

Source Language: PL/1

Run Time: Approximately five minutes CPU time

Job Description: Sorts and edits Direct Cost Summary Cards,
prints list of invalid input cards, writes
correct input data to disk file.

Printer Used: Yes Punch Used: No

Printer Forms: Wide stock forms with standard alignment and
standard carriage control tape.

Sort Sequence: Card Positions 1-11, CH, A

Data Sets:

	<u>DDNAME</u>	<u>DSNAME</u>	<u>T/D/C/P</u>	<u>I/O</u>	<u>DISP</u>	<u>LRECL</u>
<u>Step 1</u>	SORTIN	*	C	I		
	SORTOUT	&&DCOST	D	O	NEW,PASS	80
<u>Step 2</u>	DCOST	DCOST1	D	O	NEW,KEEP	65
	SDCOST	*.STEP1.SORT.SORTOUT	D	I	OLD,DELETE	80
	SYSIN					
	SYSPRINT					

Abstract for Program INDCOST

1. Purpose: To sort and edit the Indirect Cost Cards.
2. Procedure: Program INDCOST is a two-step job. The first step is a utility sort routine which sorts the input cards by county number. The second step performs reasonableness checks of the input cards on the county field and the remaining data fields taken collectively. Records without errors are written to a disk output file. Input records containing errors are listed on printed output.
3. Configuration: IBM 370/145
One Card Reader
One Printer
One to Two Tracks of IBM 3330 Disk Space
4. Source Language: Pl/1
5. Limitations: None
6. Running Time: Approximately one minute CPU time
7. Additional Remarks: After running INDCOST, a check should be made of the error report to eliminate errors associated with keypunching and local data processing. If such errors are found, corrections should be made and INDCOST should be rerun prior to running MAIN. Error listings associated with county input are returned to the counties with the other output from the system.
8. Subroutines: None

Keypunch Instructions for Program INDCOST

Input Document: Indirect Cost Form

Card Columns	Field Name	Numeric	Alpha- numeric
1-3	County Number	X	
4-10	Salaries-Supervisory Employees	X	
11-17	Salaries-Clerical Employees	X	
18-24	Utilities	X	
25-31	Building Depreciation, Maintenance, Rental	X	
32-37	Shop Equipment Depreciation	X	
38-43	Replacement of Expendable Shop Tools	X	
44-49	Offices Supplies	X	
50-56	Cost of Moving Equipment	X	
57-63	Equipment Insurance	X	

Special Instructions: Use left zero capability on all numeric fields when setting up drum card. A sample Indirect Cost Form appears on the next page.

INDIRECT COST FORM

D-9

COUNTY NAME _____

DATE _____

1 [][][][]

COUNTY NUMBER

4 [][][][][][][][][]

SALARIES AND EXPENSES OF SUPERVISORY EMPLOYEES NOT DIRECTLY EMPLOYED IN SERVICING OR REPAIRING EQUIPMENT

11 [][][][][][][][][]

SALARIES OF CLERICAL EMPLOYEES ENGAGED IN ACCOUNTING AND PREPARING REPORTS FOR EQUIPMENT.

SHOP STORAGE AND MISCELLANEOUS OVERHEAD COSTS RELATING TO EQUIPMENT CARE AND HANDLING:

18 [][][][][][][][][]

a) UTILITIES

25 [][][][][][][][][]

b) SHOP AND OFFICE BUILDING DEPRECIATION, MAINTENANCE AND/OR RENTAL

32 [][][][][][][][]

c) DEPRECIATION OF SHOP EQUIPMENT

38 [][][][][][][][]

d) REPLACEMENT COST OF EXPENDABLE SHOP TOOLS

44 [][][][][][][][]

e) OFFICE SUPPLIES

50 [][][][][][][][][]

f) COST OF MOVING EQUIPMENT

57 [][][][][][][][][]

g) EQUIPMENT INSURANCE

Computer Operating Instructions for Program INDCOST

Frequency of Run: Annually

Region: 100K

Source Language: PL/1

Run Time: Approximately one minute CPU time

Job Description: Sorts and edits Indirect Cost Cards,
prints list of invalid input cards, writes
correct data to disk file.

Printer Used: Yes Punch Used: No

Printer Forms: Wide stock forms with standard alignment and
standard carriage control tape.

Sort Sequence: Card Positions 1-3, CH, A

Data Sets:

	<u>DDNAME</u>	<u>DSNAME</u>	<u>T/D/C/P</u>	<u>I/O</u>	<u>DISP</u>	<u>LRECL</u>
<u>Step 1</u>	SORTIN	*	C	I		
	SORTOUT	&&ICOST	D	O	NEW,PASS	80
<u>Step 2</u>	ICOST	ICOST1	D	O	NEW,KEEP	63
	SICOST	*.STEP1.SORT.SORTOUT	D	I	OLD,DELETE	80
	SYSIN					
	SYSPRINT					

Abstract for Program STCHNGE

1. Purpose: To sort and edit the Equipment Status Change Cards.
2. Procedure: Program STCHNGE is a two-step job. The first step is a utility sort routine which sorts the input cards by county number, equipment number, and card type. The second step first reads in from cards the 55 equipment class codes and descriptions. It then performs reasonableness checks on all three cards of each equipment status change record. Input records containing errors are grouped by county and listed on printed output. Input records without errors have the appropriate district number and equipment class description inserted in the record with the record then written to a disk output file.
3. Configuration: IBM 370/145
One Card Reader
One Printer
Three to Four Cylinders of IBM 3330
Disk Space
4. Source Language: PL/1
5. Limitations: None
6. Running Time: Approximately two minutes CPU time
7. Additional Remarks: After running STCHNGE, a check should be made of the error report to eliminate errors

associated with keypunching and local data processing.

If such errors are found, corrections should be made and STCHNGE should be rerun prior to running UPDATEM. Error listings associated with county input are returned to the counties with other output from the system.

8. Subroutines: None

Keypunch Instructions for Program STCHNGECard 1

Input Document: Equipment Status Change Form

Card Columns	Field Name	Numeric	Alpha- numeric
1-3	County Number	X	
4-11	Equipment Number		X
12	Card Type	X	
13-14	District	X	
15-16	Class Code	X	
17-36	Class Description		X
37-39	Manufacturer Code Number	X	
40-41	Year Equipment Manufactured	X	
42-55	Make and Model Description		X
56-69	Manufacturer's Serial Number		X
70-72	Wheelbase	X	
73-78	Date Purchased	X	

Key punch Instructions for Program STCHNGECard 2

Input Document: Equipment Status Change Form

Card Columns	Field Name	Numeric	Alpha- numeric
1-3	County Number	X	
4-11	Equipment Number		X
12	Card Type	X	
13	Type of Engine		X
14-27	Engine Make & Model Description		X
28-29	Engine Manufacturer Code	X	
30-32	Rated Horsepower	X	
33-34	Number of Cylinders in Engine	X	
35	Transmission Type		X
36-47	Dealer Purchased From		X
48-55	Original Purchase Cost	X	
56-62	Salvage Value	X	
63-70	Book Value	X	
71	Sold, Junked or Traded Code		X
72-77	Date Sold, Junked or Traded	X	

Keypunch Instructions for Program STCHNGECard 3

Input Document: Equipment Status Change Form

Card Columns	Field Name	Numeric	Alpha-numeric
1-3	County Number	X	
4-11	Equipment Number		X
12	Card Type	X	
13-18	Miles or Hours-Life	X	
19-25	Fuel Cost-Life	X	
26-31	Lubricant Cost-Life	X	
32-37	Tires and Tubes Cost-Life	X	
38-43	Expendable Parts Cost-Life	X	
44-48	Antifreeze Cost-Life	X	
49-55	Parts Cost-Life	X	
56-62	Labor Cost-Life	X	
63-69	Indirect Cost-Life	X	

Special Instructions: All three cards must be keypunched. Use left zero capability on all numeric fields when setting up drum card. Blank numeric fields may be skipped, rather than filled with all zeros. A sample Equipment Status Change Form appears on the next page.

EQUIPMENT STATUS CHANGE FORM

D-16

COUNTY NUMBER _____ COUNTY NAME _____
 EQUIPMENT NUMBER _____ DATE _____

12 CARD TYPE 1
 13 DISTRICT
 15 CLASS CODE
 17 CLASS DESCRIPTION
 37 MANUFACTURER CODE NUMBER
 40 YEAR EQUIPMENT MANUFACTURED
 42 MAKE AND MODEL DESCRIPTION
 56 MANUFACTURER'S SERIAL NUMBER
 70 WHEELBASE IN INCHES
 73 DATE PURCHASED-MMDDYY

12 CARD TYPE 2
 13 TYPE OF ENGINE - GAS (G) OR DIESEL (D)
 14 ENGINE MAKE AND MODEL DESCRIPTION
 28 ENGINE MANUFACTURER CODE
 30 RATED HORSEPOWER
 33 NUMBER OF CYLINDERS IN ENGINE
 35 TRANSMISSION TYPE - AUTOMATIC (A) OR STANDARD (S)
 36 DEALER PURCHASED FROM
 48 ORIGINAL PURCHASE COST
 56 SALVAGE VALUE
 63 BOOK VALUE
 71 SOLD, JUNKED, OR TRADED CODE - (S)(J)(T)
 72 DATE SOLD, JUNKED OR TRADED - MMDDYY

12 CARD TYPE 3
 13 MILES OR HOURS - LIFE
 19 FUEL COST
 26 LUBRICANT COST - LIFE
 32 TIRES AND TUBES COST - LIFE
 38 EXPENDABLE PARTS COST - LIFE
 44 ANTIFREEZE COST - LIFE
 49 REPAIR PARTS COST - LIFE
 56 LABOR COST - LIFE
 63 INDIRECT COST - LIFE

Computer Operating Instructions for Program STCHNGE

Frequency of Run: Annually

Region: 100K

Source Language: PL/1

Run Time: Approximately two minutes CPU time

Job Description: Sorts and edits Equipment Status Change Cards, prints list of invalid input cards, combines the three input cards per piece of equipment into one logical record, and writes correct input data to disk file. Also reads in list of class codes and descriptions from cards and places proper description in each equipment status change record.

Printer Used: Yes Punch Used: No

Printer Forms: Wide stock forms with standard alignment and standard carriage control tape.

Sort Sequence: Card Positions 1-12, CH, A

Data Sets:

	DDNAME	DSNAME	T/D/C/P	I/O	DISP	LRECL
<u>Step 1</u>	SORTIN	*	C	I		
	SORTOUT	&&CHANGE	D	O	NEW,PASS	80
<u>Step 2</u>	TSC	*.STEP1.SORT.SORTOUT	D	I	OLD,DELETE	80
	SCHANGE	SCHANGE1	D	O	NEW,KEEP	199
	SYSIN					
	SYSPRINT					

Abstract for Program UPDATEM

1. Purpose: To add new equipment records to the equipment master file, to correct master file records which have been found to contain incorrect data, to flag records of equipment disposed of, to calculate the total book value of all equipment for each county, and to print out an inventory listing of all equipment in each county.
2. Procedure: Records are read from the equipment master file (a tape file saved from the previous year's processing) and from the equipment status change record file created in program STCHNGE. Equipment status change records which do not match any master file record are added to the master file while matching records are corrected or flagged for later deletion.

While the program is adding records to the master file and updating others, it also calculates the total book value of all equipment for each county. A disk output file is constructed containing records for each county. These records contain the county number and the total book value of all county equipment recorded in the master file.

Printed output is also provided giving an inventory listing by county of each piece of equipment held in the master file.
3. Configuration: IBM 370/145

One Printer

One Tape Drive

Approximately Eight Cylinders of IBM
3330 Disk Space

4. Source Language: PL/I
5. Limitations: Cannot be run until after STCHNGE has been run. The first time the system is run, a dummy equipment master file with only an EOF marker must have been created.
6. Running Time: Approximately five minutes CPU time.
7. Additional Remarks: Two copies of the printed output are to be produced for county use.
8. Subroutines: No external procedures.

Computer Operating Instructions for Program UPDATM

Frequency of Run: Annually

Region: 100K

Source Language: PL/1

Run Time: Approximately five minutes CPU time

Job Description: Reads master file and equipment status change record file. Attempts to match records from these files by county number and equipment number. Matching records cause master file to be updated. Non-matching equipment status change records are added to master file. Total equipment book value is calculated by county and written to disk output file. A printed inventory listing of all equipment by county is produced.

Printer Used: Yes Punch Used: No

Printer Forms: Wide stock forms with standard alignment and standard carriage control tape. Make two copies of printed output.

Data Sets:

<u>DDNAME</u>	<u>DSNAME</u>	<u>T/D/C/P</u>	<u>I/O</u>	<u>DISP</u>	<u>LRECL</u>
EQMAST	EQMAST1	T	I	OLD,KEEP	300
NEQMAST	EQMAST2	D	O	NEW,KEEP	300
STCHNG	SCHANGE1	D	I	OLD,DELETE	199
BKVALU	BKVALU1	D	O	NEW,KEEP	13
SYSPRINT					

Abstract for Program MAIN

1. Purpose: To update all master file records with the past year's direct and indirect cost data, to calculate cost data, to calculate the past year's and lifetime costs per mile/hour for each piece of equipment, and to produce a detailed printed output for all equipment by county and equipment classes showing direct costs, indirect costs, and cost per mile/hour figures for the past year and for the equipment's lifetime.
2. Procedure: MAIN is a three step job. The first step updates the master file, the second step is a utility sort to sort the master file into the appropriate sequence for the printed output, which is then produced in the third step.

The first step reads the records from the indirect cost record file and the total book value file created by INDCOST and UPDATEM, respectively, and places the contents of the records read into arrays. Next, records are read from the master file created in UPDATEM and the direct cost summary records file created in DIRECT. The program logic attempts to match records from these two files by county number and equipment number. All master file records which are matched by a corresponding direct cost summary record are updated. The past year's direct costs from the direct cost summary record are entered in-

to the master file record and the lifetime direct costs are updated. A portion of the county's total indirect equipment costs are allocated to each piece of equipment based upon the ratio of each piece's book value to the county's total equipment book value. The current book value of each piece of equipment is then updated through an appropriate depreciation calculation.

If a match does not occur between the master file record and the direct cost summary record, a printed listing of the nonmatching record is produced. Most of the listings in this output will stem from using an incorrect equipment number or county number on the direct cost summary form, or not submitting a direct cost summary form for a piece of equipment currently on the master file.

The second step of MAIN is a utility sort routine. The existing master file is sorted by county number and equipment number within the county number. The output desired from MAIN is classified by county, equipment class, and the equipment number. Therefore, the sort is necessary to get all pieces of equipment for each county into their proper equipment class so class totals can be calculated and printed along with the individual listings for each piece of equipment in each equipment class.

The third step of MAIN calculates total direct cost for each record for the past year and for the equipment's lifetime. Cost per mile/hour figures are then calculated.

Equipment class and county totals are accumulated, both for the past year's costs as well as for lifetime costs. Output is produced by county and equipment class showing detailed cost breakdowns for each piece of equipment for the past year's costs as well as for lifetime costs.

Equipment class and county totals are also shown.

3. Configuration: IBM 370/145
One Printer
Six to Eight Cylinders of IBM 3330 Disk Space
4. Source Language: PL/1
5. Limitations: Cannot be run until DIRECT, INDCOST, and UPDATEM have been run.
6. Running Time: Approximately 30 minutes CPU time.
7. Additional Remarks: Two copies of the printed output are to be produced for county use.
8. Subroutines: No external procedures.

Computer Operating Instructions for Program MAIN

Frequency of Run: Annually

Region: 100K

Source Language: PL/1

Run Time: Approximately 30 minutes CPU time

Job Description: Reads equipment master file, direct cost summary file, indirect cost record file, total book value file, and updates master file records from matching direct cost summary records producing updated equipment master file. Sorts updated equipment master file, produces detailed printed cost figures on all equipment records in master file by county and equipment class.

Printer Used: Yes

Punch Used: No

Printed Forms: Wide stock forms with standard alignment and standard carriage control tape. Make two copies of printed output.

Sort Sequence: Record Positions 1-3, CH, A; 14-15, CH, A;
6-13, CH, A

Data Sets:

	DDNAME	DSNAME	T/D/C/P	I/O	DISP	LRECL
<u>Step 1</u>	SEQMAST	EQMAST2	D	I	OLD,DELETE	300
	TEQMAST	EQMAST3	D	O	NEW,PASS	300
	BKVALU	BKVALU1	D	I	OLD,DELETE	13
	INDCOST	ICOST1	D	I	OLD,DELETE	63
	DIRCOST	DCOST1	D	I	OLD,DELETE	65
	SYSPRINT					
<u>Step 2</u>	SORTIN	*STEP1.GO.TEQMAST	D	I	OLD,KEEP	300
	SORTOUT	&&SMAST	D	O	NEW,PASS	300
<u>Step 3</u>	FEQMAST	*.STEP2.SORT.SORTOUT	D	I	OLD,DELETE	300
	YRDIR		P	O		132
	YRTOT		P	O		132
	LIFEDIR		P	O		132
	LIFETOT		P	O		132

Abstract for Program CTYSUMRY

1. Purpose: To produce past year and lifetime cost per mile/hour averages for each equipment class by districts and counties.
2. Procedure: Program CTYSUMRY is a two-step job. The first step sorts the master file produced in MAIN by equipment class, district, and county number. The second step reads records from the sorted file produced in step one. It keeps running totals of total direct costs, pieces of equipment, and miles/hours. When the equipment class, district number, or county number of the record just read differs from the previous record, the appropriate average cost per mile/hour calculations are made, the averages are printed, and appropriate running totals are reset to zero.
3. Configuration: IBM 370/145
One Printer
Six to Eight Cylinders of IBM 3330 Disk Space
4. Source Language: PL/1
5. Limitations: Cannot be run until MAIN has produced an appropriate master file. One hundred copies must be produced if all counties are to receive output.
6. Running Time: Approximately 10 minutes CPU time.

7. Additional Remarks: Make 100 copies of printed output.

8. Subroutines: None

Computer Operating Instructions for Program CTYSUMRY

Frequency of Run: Annually

Region: 100K

Source Language: PL/1

Run Time: Approximately 10 minutes CPU time.

Job Description: Sorts master file, produces past year
and lifetime cost per mile/hour averages
for each equipment class by districts and
counties.

Printer Used: Yes Punch Used: No

Printer Forms: Wide stock forms with standard alignment and
standard carriage control tape. Make 100
copies.

Sort Sequence: Record positions 14-15, CH, A; 4-5, CH, A; 1-3,
CH, A

Data Sets:

	DDNAME	DSNAME	T/D/C/P	I/O	DISP	LRECL
<u>Step 1</u>	SORTIN	EQMAST3	D	I	OLD,KEEP	300
	SORTOUT	&&COUNTY	D	O	NEW,PASS	300
<u>Step 2</u>	UEMF	*.STEP1.SORT.SORTOUT	D	I	OLD,DELETE	300
	OUT		P	O		132

Abstract for Program MFGAGE

1. Purpose: To produce past year and lifetime cost per mile/hour averages for five age groups in each equipment class by manufacturer.
2. Procedure: Program MFGAGE is a two-step job. The first step sorts the master file produced in MAIN by equipment class and manufacturer. The second step reads records from the sorted file produced in step one. This step keeps running totals of total direct costs, pieces of equipment, and miles/hours. After each record is read, the age of the piece of equipment contained on that record is calculated, and the direct costs and miles/hours on the record are added to the appropriate age grouping. Cost per mile/hour calculations are made and output produced whenever the current and past records have differing equipment class codes or manufacturer codes.
3. Configuration: IBM 370/145
One Printer
Six to Eight Cylinders of IBM 3330 Disk Space
4. Source Language: PL/1
5. Limitations: Cannot be run until MAIN has produced an appropriate master file. One hundred copies must be produced if all counties are to receive output.

6. Running Time: Approximately 12 minutes CPU time.
7. Additional Remarks: Make 100 copies of printed output.
8. Subroutines: None

Computer Operating Instructions for Program MFGAGE

Frequency of Run: Annually

Region: 100K

Source Language: PL/1

Run Time: Approximately 12 minutes CPU time

Job Description: Sorts master file, produces past year and lifetime averages cost per mile/hour for five groups in each equipment class by manufacturer.

Printer Used: Yes Punch Used: No

Printer Forms: Wide stock forms with standard alignment and standard carriage control tape. Make 100 copies.

Sort Sequence: Record Positions 14-15, CH, A; 18-20, CH, A

Data Sets:

	<u>DDNAME</u>	<u>DSNAME</u>	<u>T/D/C/P</u>	<u>I/O</u>	<u>DISP</u>	<u>LRECL</u>
<u>Step 1</u>	SORTIN	EQMAST3	D	I	OLD,KEEP	300
	SORTOUT	&&MFGAGE	D	O	NEW,PASS	300
<u>Step 2</u>	UEMF	*.STEP1.SORT.SORTOUT	D	I	OLD,DELETE	300
	OUT		P	O		132

Abstract for Program UPDATE

1. Purpose: To delete records of equipment disposed of in the past year from the master file and to produce a printed listing by county of the deleted records.
2. Procedure: Program UPDATE reads the master file created in MAIN. All records to be deleted were flagged for deletion by program UPDATEM by setting a flag in the disposal method field. All records with the disposal method field set to S, J, or T are removed from the master file and listed by county on the printer. The master file produced is written on magnetic tape and will be used as input in the year to follow.
3. Configuration: IBM 370/145
One Printer
One Tape Drive
Three to Four Cylinders of IBM 3330
Disk Space
4. Source Language: PL/1
5. Limitations: Cannot be run until MAIN, CTYSUMRY, and MFGAGE have been run.
6. Running Time: Approximately two minutes CPU time.
7. Additional Remarks: The new master tape file produced must be labeled with the current year to avoid confusing it with the previous year's master file. Make a back up magnetic tape copy of the new master file.
8. Subroutines: None

Computer Operating Instructions for Program UPDATE

Frequency of Run: Annually

Region: 100K

Source Language: PL/1

Run Time: Approximately two minutes CPU time

Job Description: Deletes records of equipment disposed of in the past year from the master file, produces printed output of deleted records, and writes new equipment master file to magnetic tape.

Printer Used: Yes Punch Used: No

Printer Forms: Wide stock forms with standard alignment and standard carriage control tape.

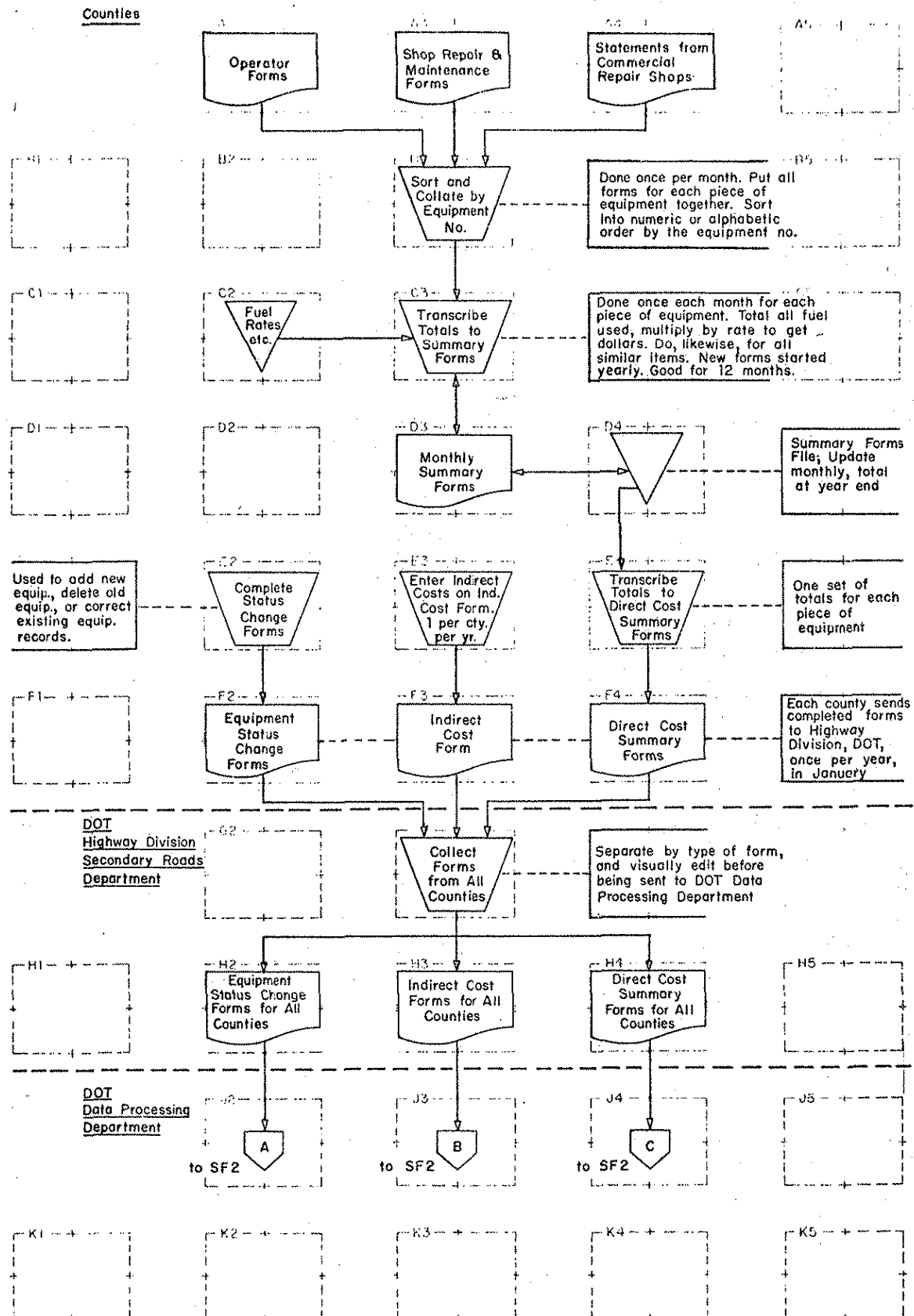
Data Sets:

<u>DDNAME</u>	<u>DSNAME</u>	<u>T/D/C/P</u>	<u>I/O</u>	<u>DISP</u>	<u>LRECL</u>
OLDMST	EQMAST3	D	I	OLD,DELETE	300
NEWMST	EQMAST1	T	O	NEW,KEEP	300
SYSPRINT					

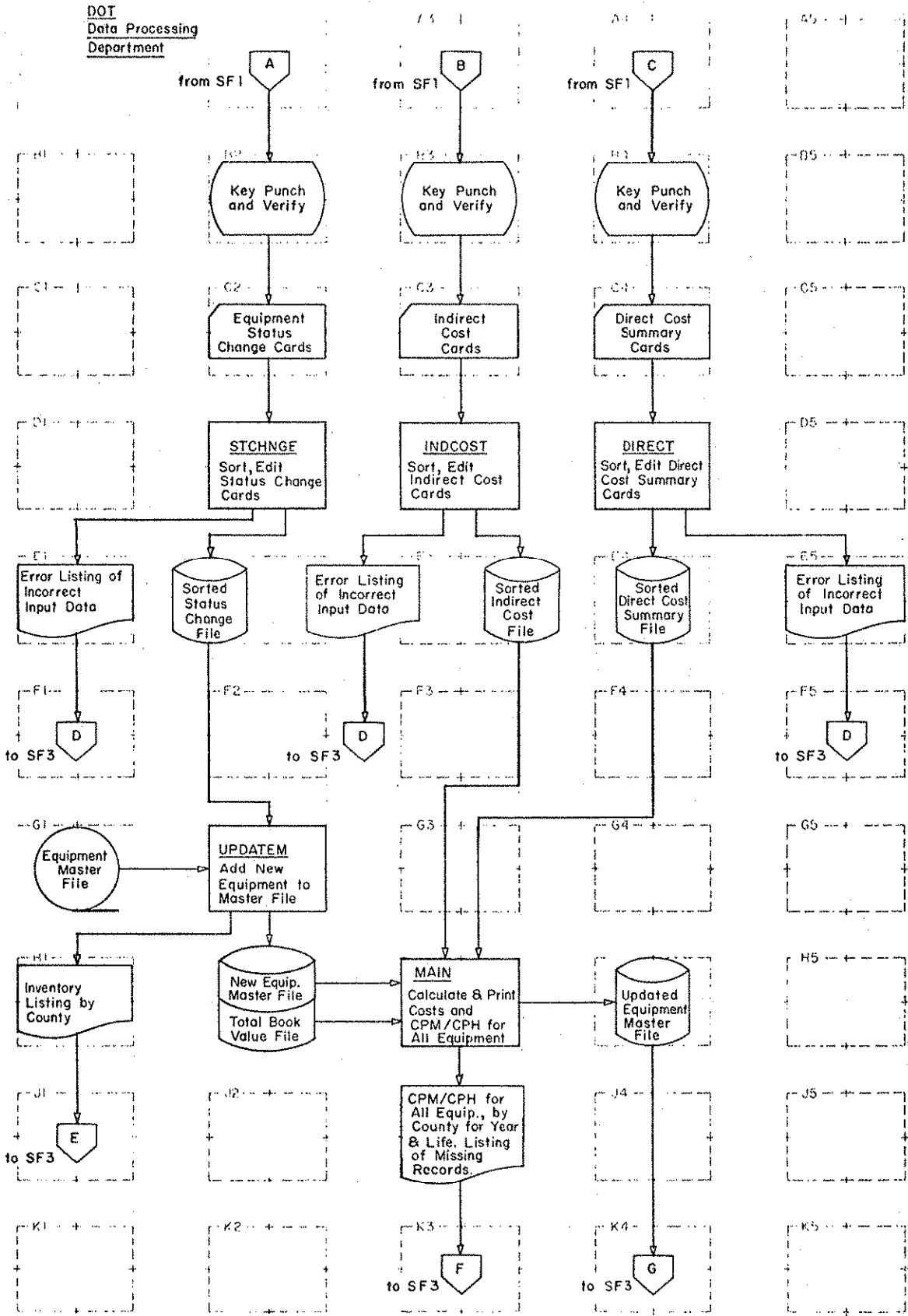
FLOWCHARTS

This section of Appendix D contains the system flowcharts and the eight computer program flowcharts. The system flowcharts give the general system information flow. The computer program flowcharts show the specific flow of data processing in each computer program. These program flowcharts correspond to the PL/1 listings given in the next section.

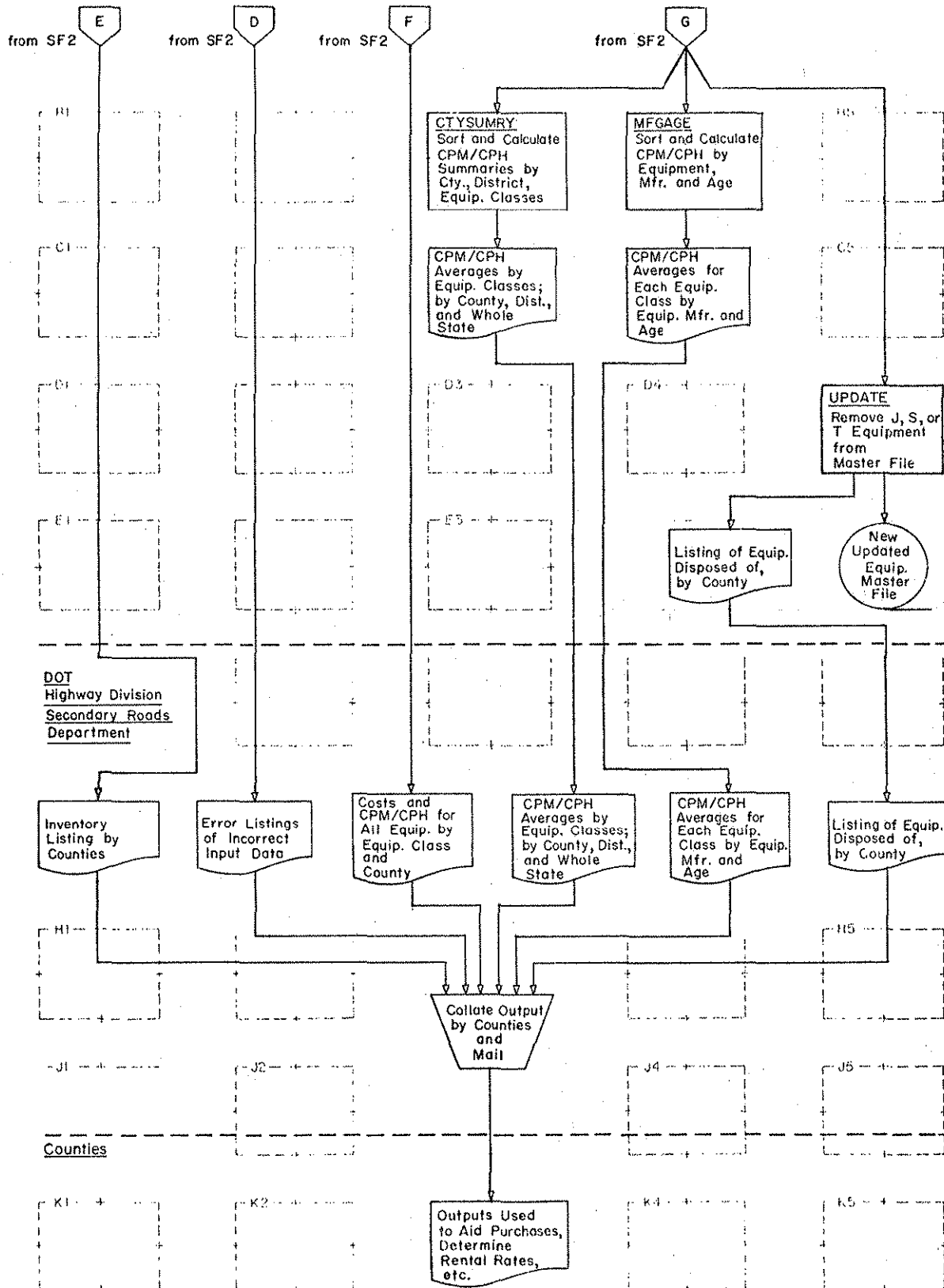
Programmer: _____ Program No.: _____ Date: _____ Page: _____
 Chart ID: SF1 Chart Name: System Flowchart Program Name: _____



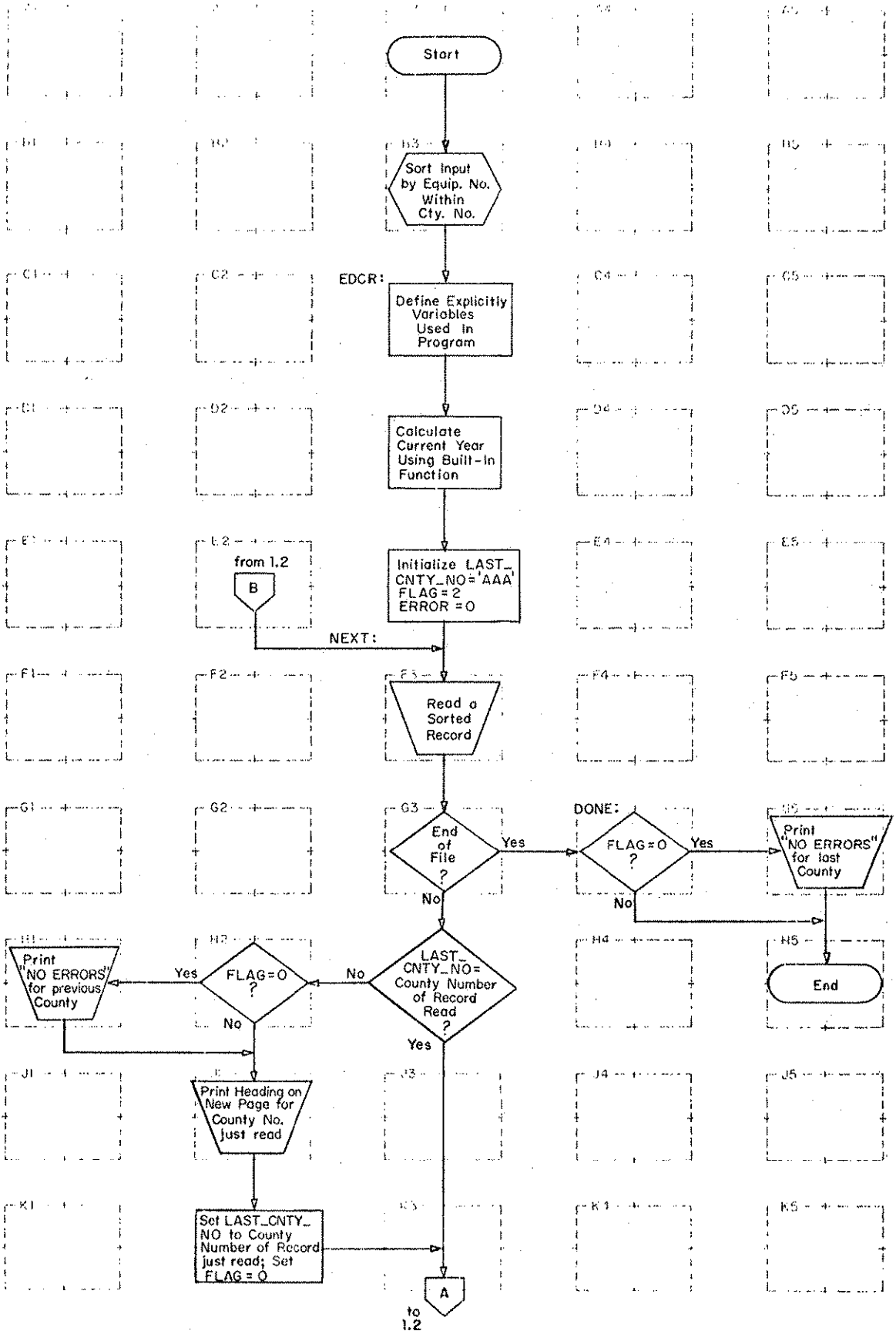
Programmer: _____ Program No.: _____ Date: _____ Page: _____
 Chart ID: SF2 Chart Name: _____ Program Name: _____



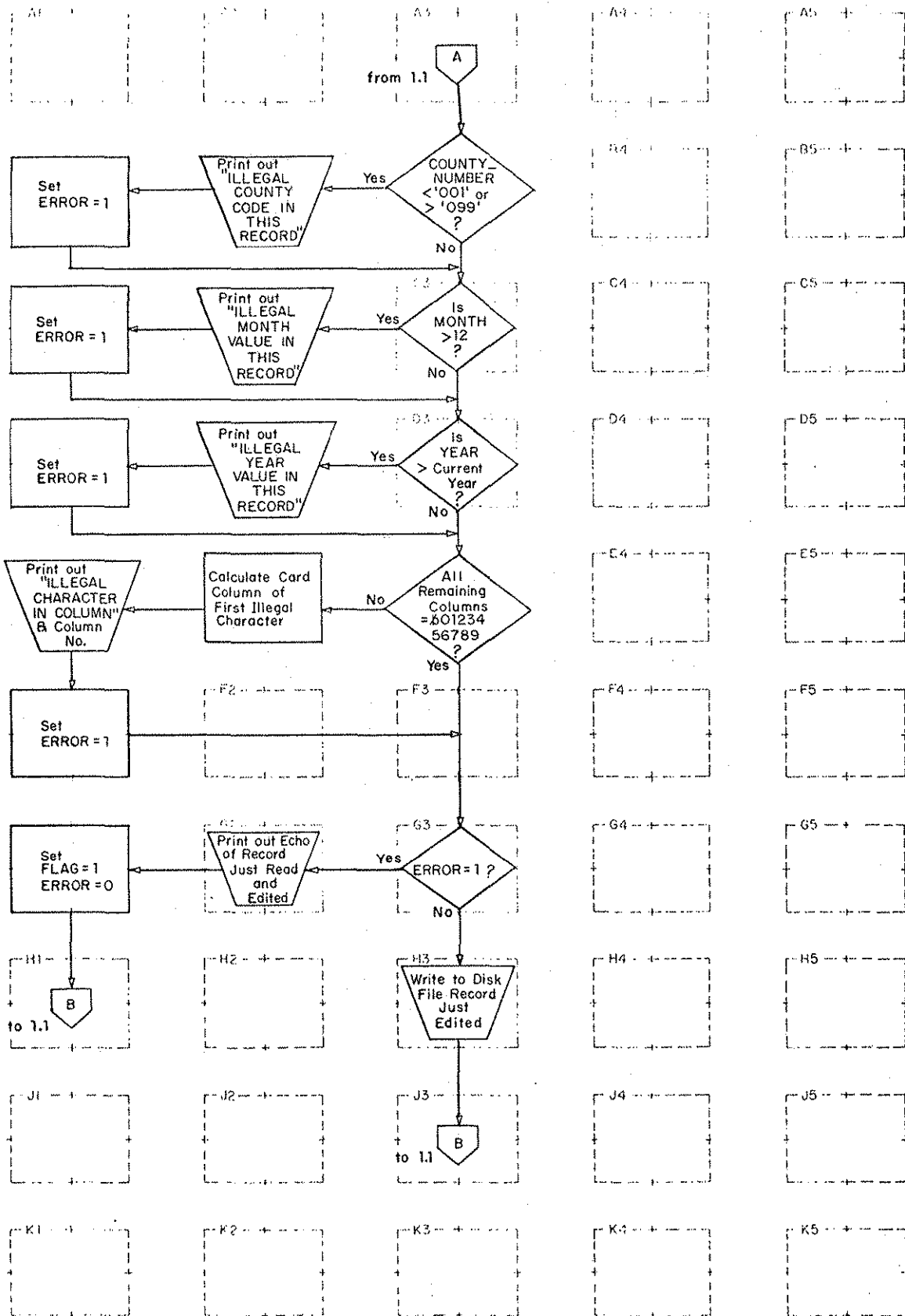
Programmer: _____ Program No.: _____ Date: _____ Page: _____
 Chart ID: SF3 Chart Name: _____ Program Name: _____



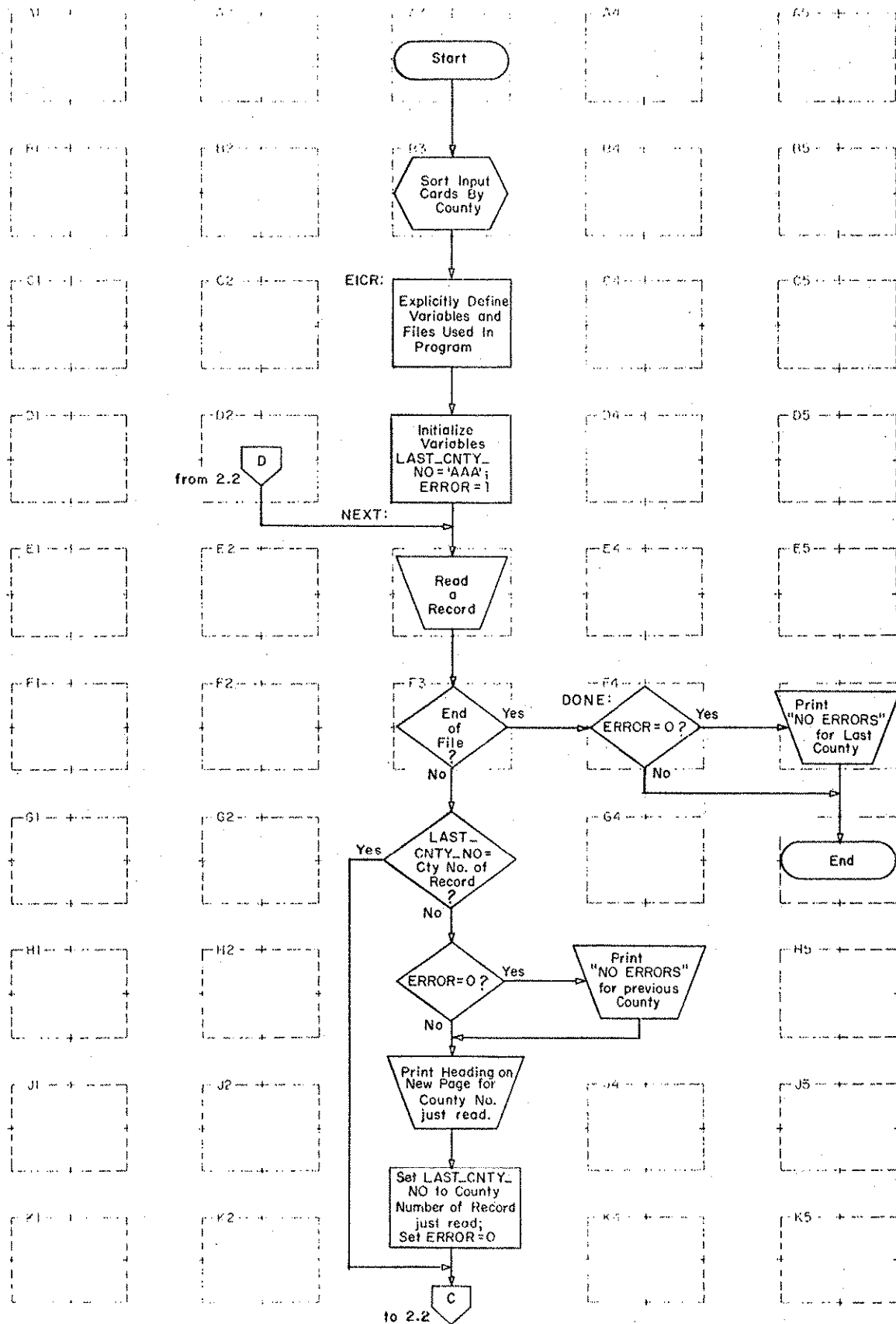
Programmer: Rolf McHenry Program No.: 1 Date: July 1975 Page: _____
 Chart ID: 1.1 Chart Name: Sort and Edit Direct Cost Records Program Name: DIRECT



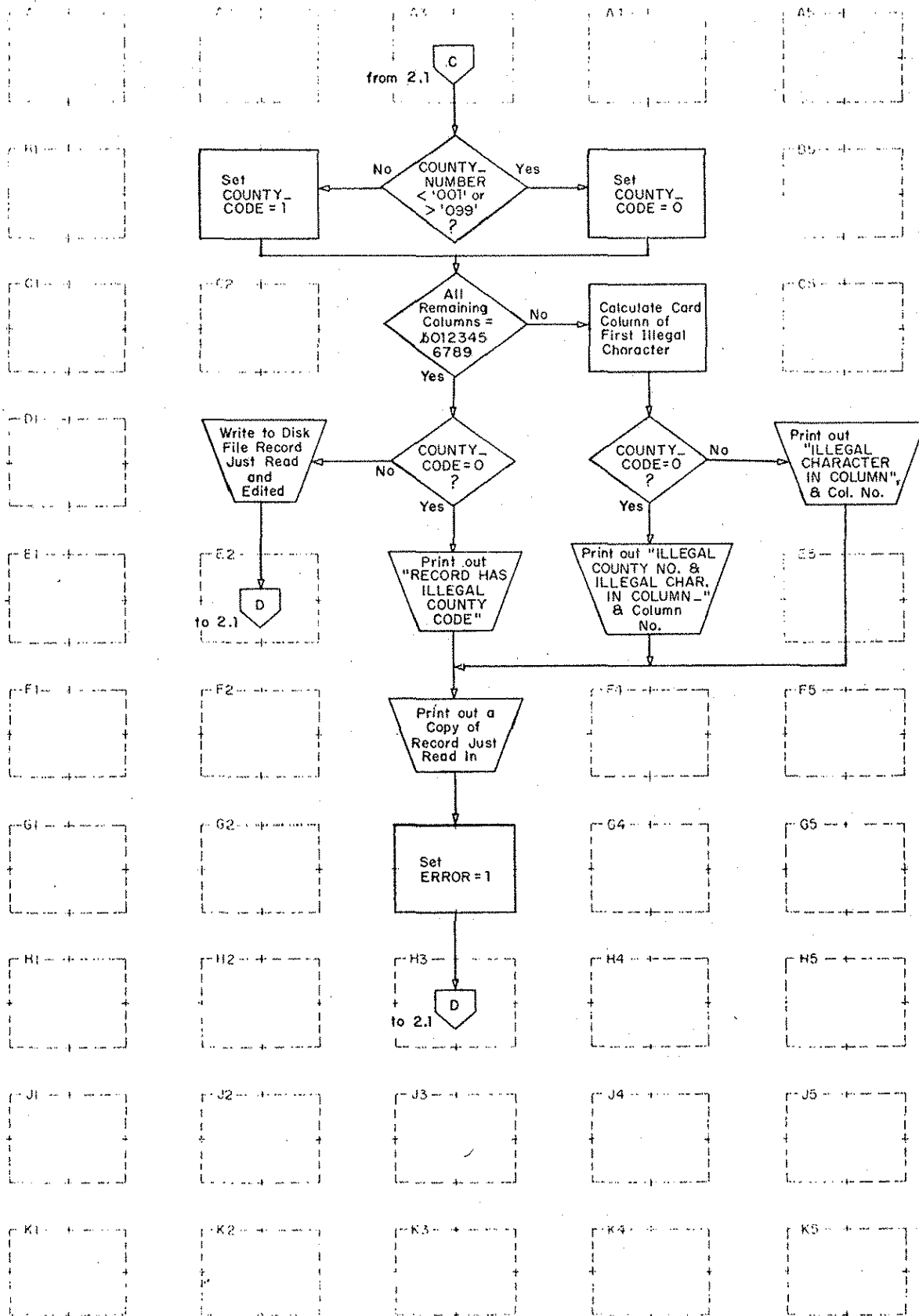
Programmer: Rolf McHenry Program No.: 1 Date: Page:
Chart ID: 1.2 Chart Name: Program Name: DIRECT



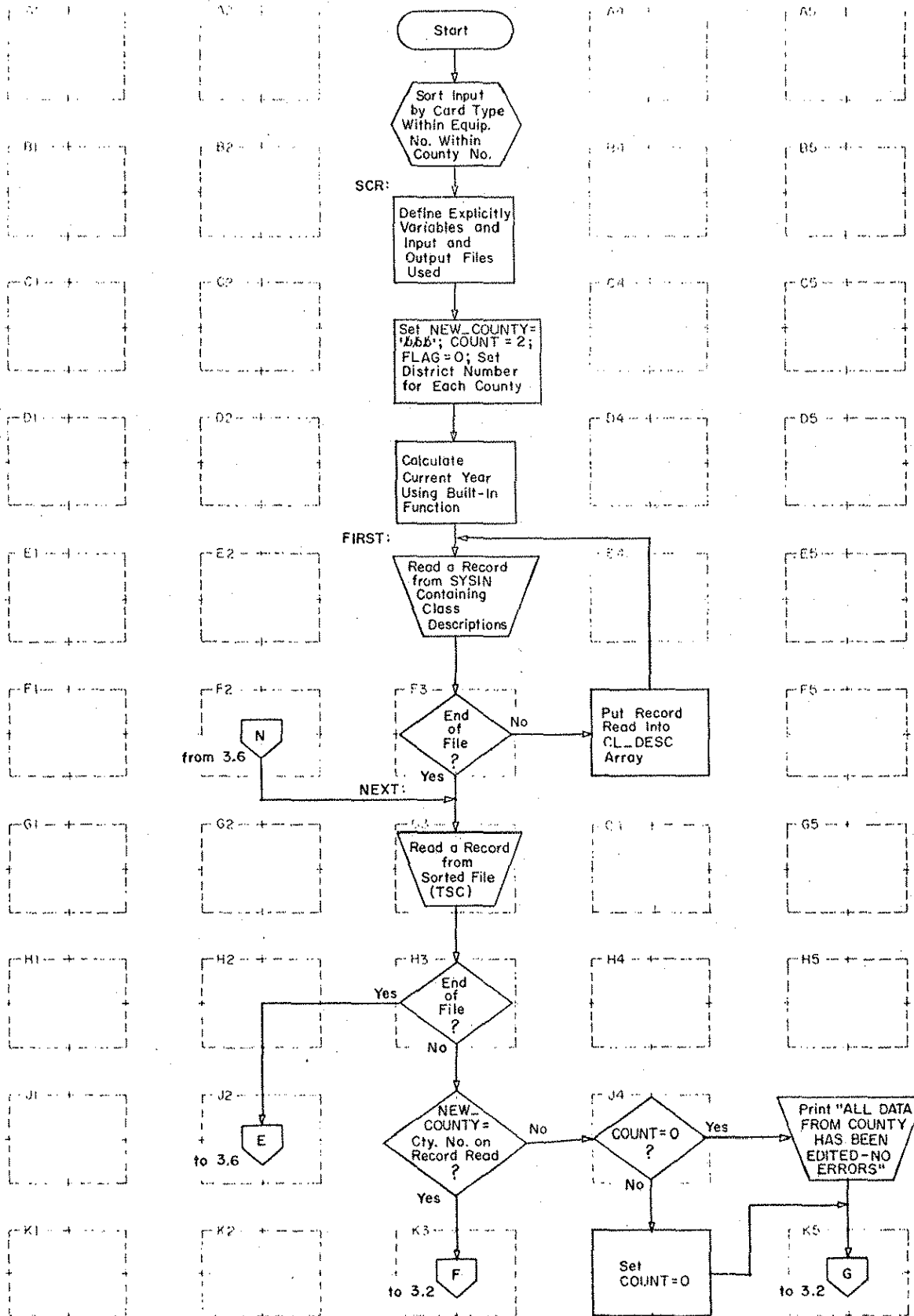
Programmer: Rolf McHenry Program No.: 2 Date: _____ Page: _____
Chart ID: 2.1 Chart Name: Sort and Edit Indirect Cost Records Program Name: INDCOST



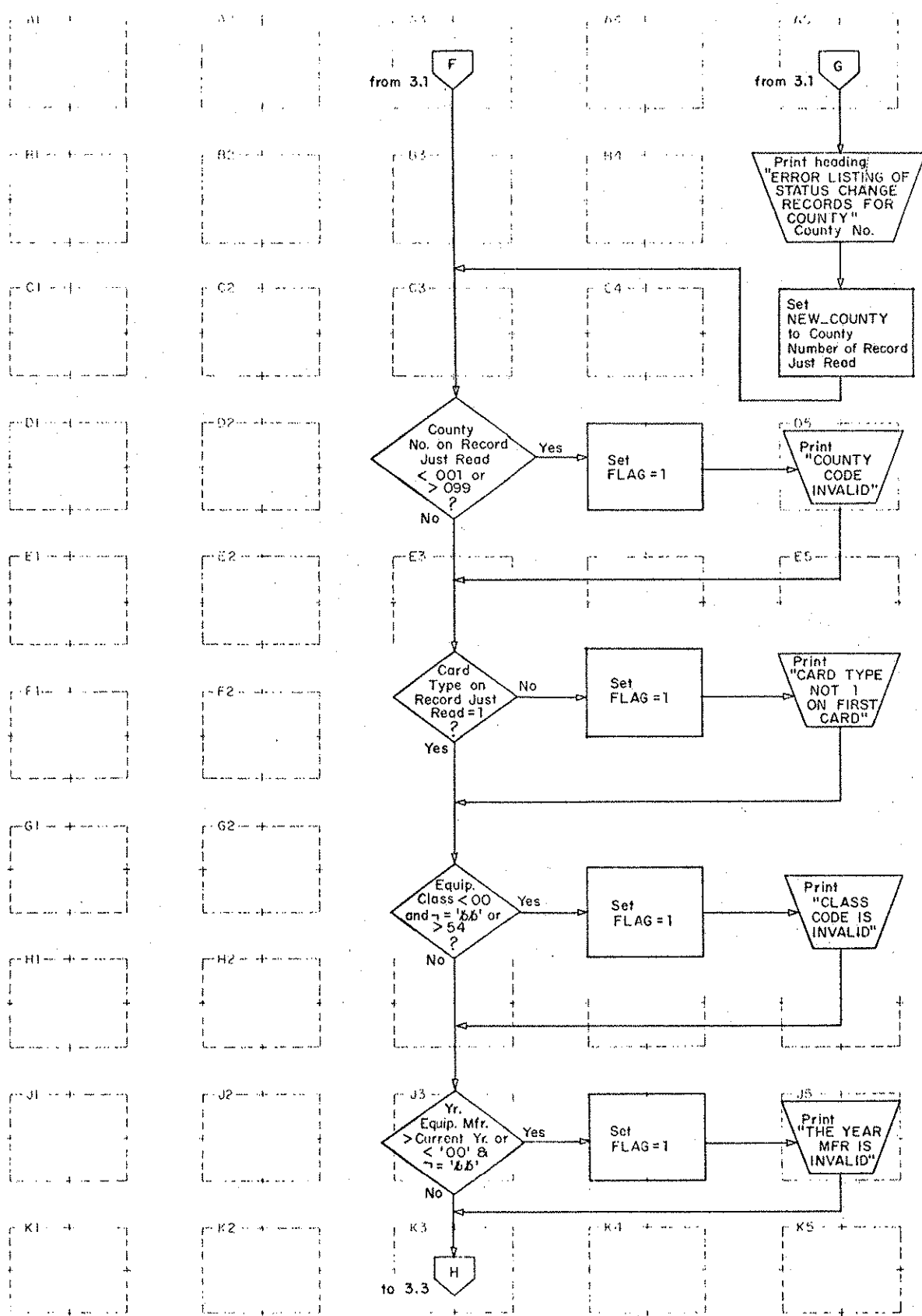
Programmer: Rolf McHenry Program No.: 2 Date: Page:
Chart ID: 2.2 Chart Name: Program Name: INDCOST



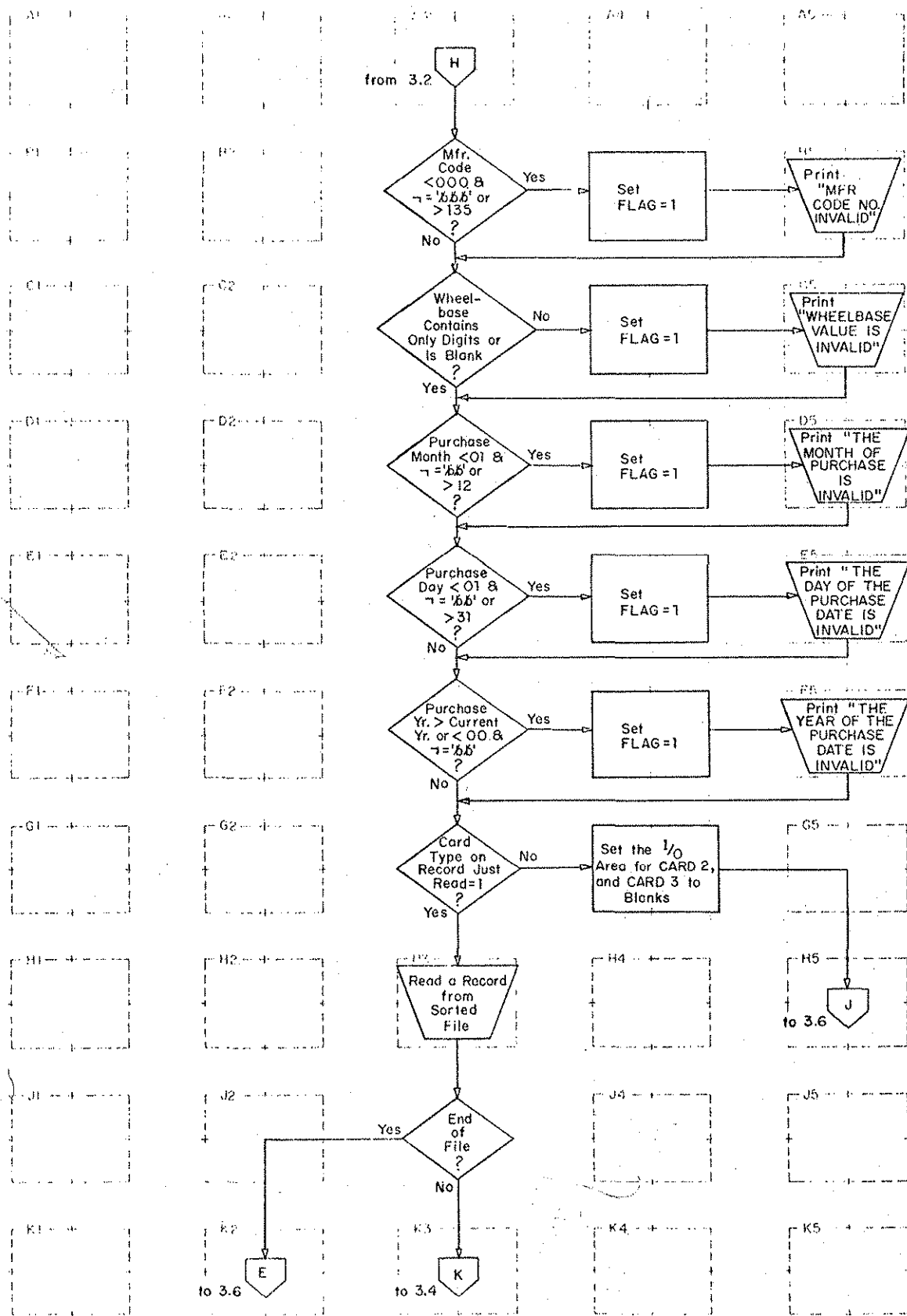
Programmer: Charles Sadoris Program No.: 3 Date: Page:
 Chart ID: 3.1 Chart Name: Sort and Edit Status Change Records Program Name: STCHNGE



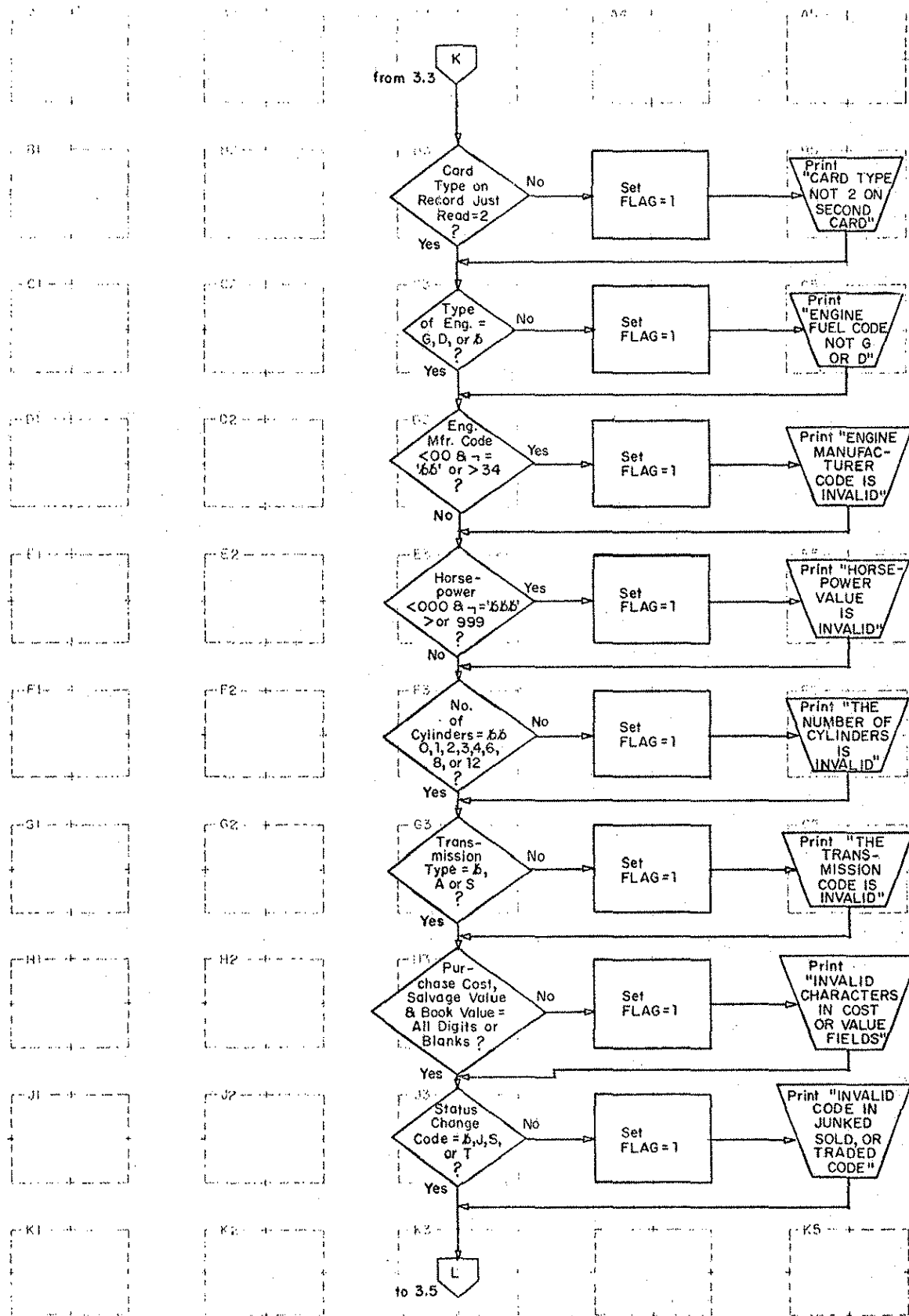
Programmer: Charles Sadoris Program No.: 3 Date: July 1975 Page: _____
 Chart ID: 3.2 Chart Name: _____ Program Name: STCHANGE

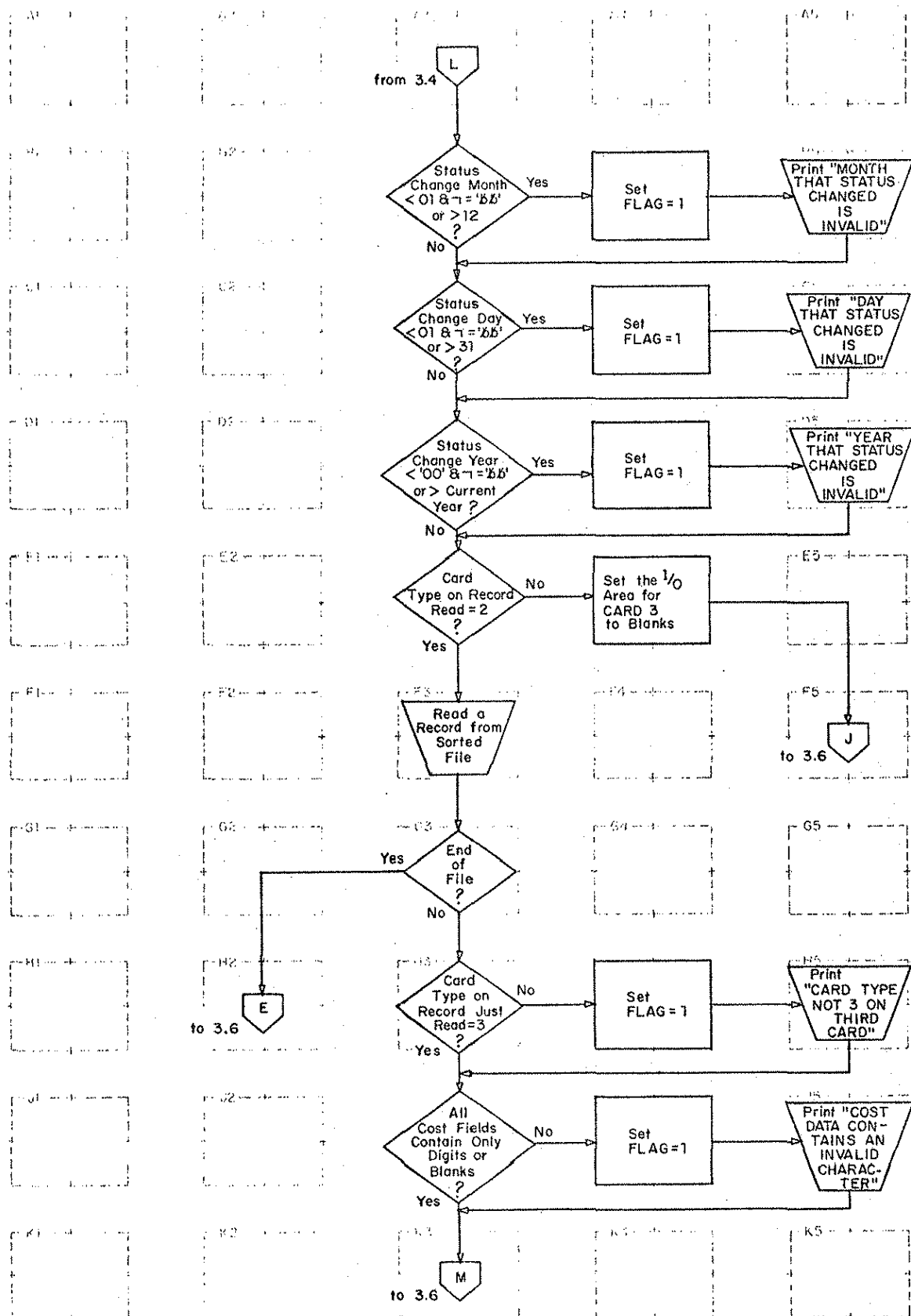


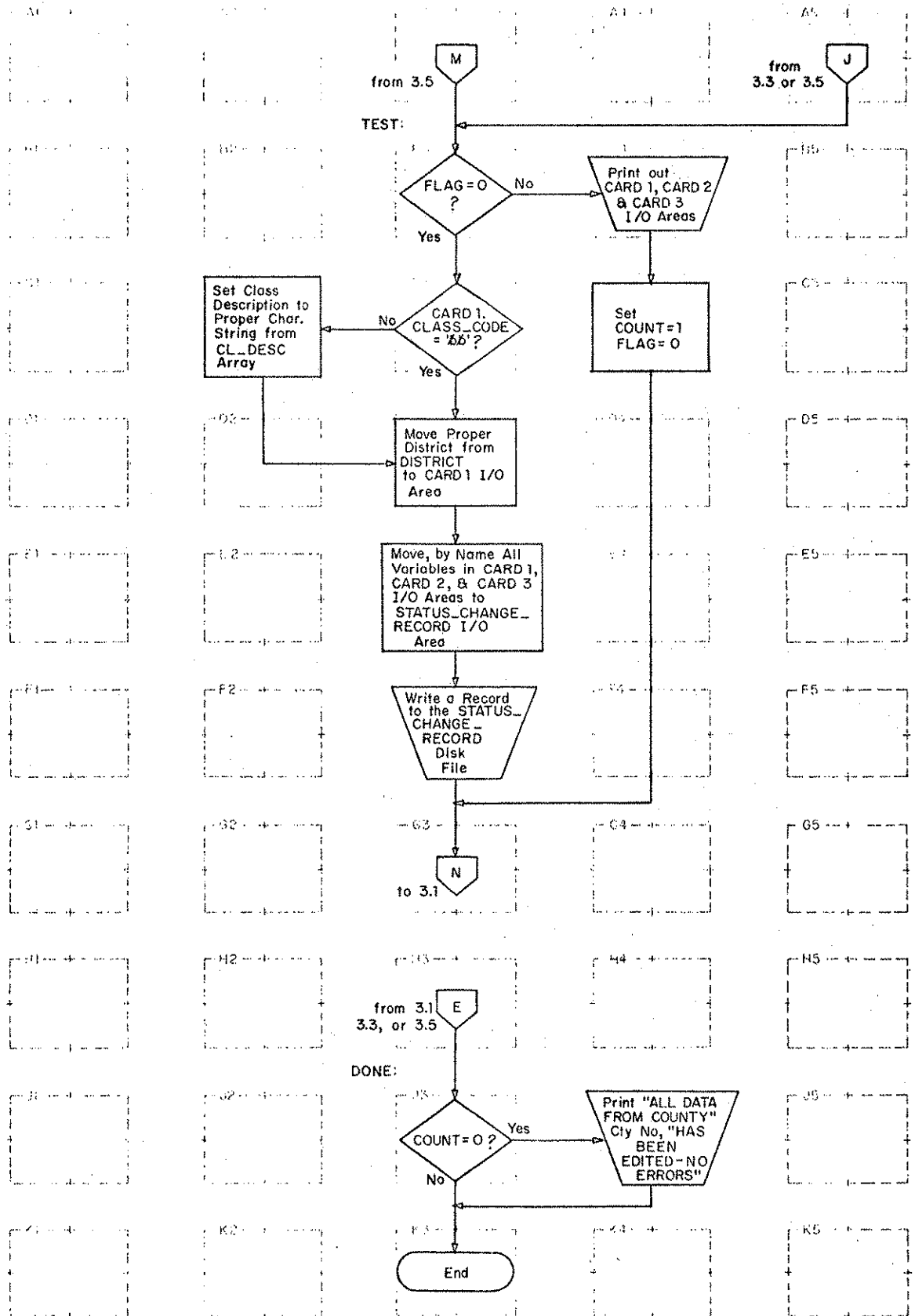
Programmer: Charles Sadoris Program No.: 3 Date: July 1975 Page: Chart ID: 3.3 Chart Name: Program Name: STCHNGE



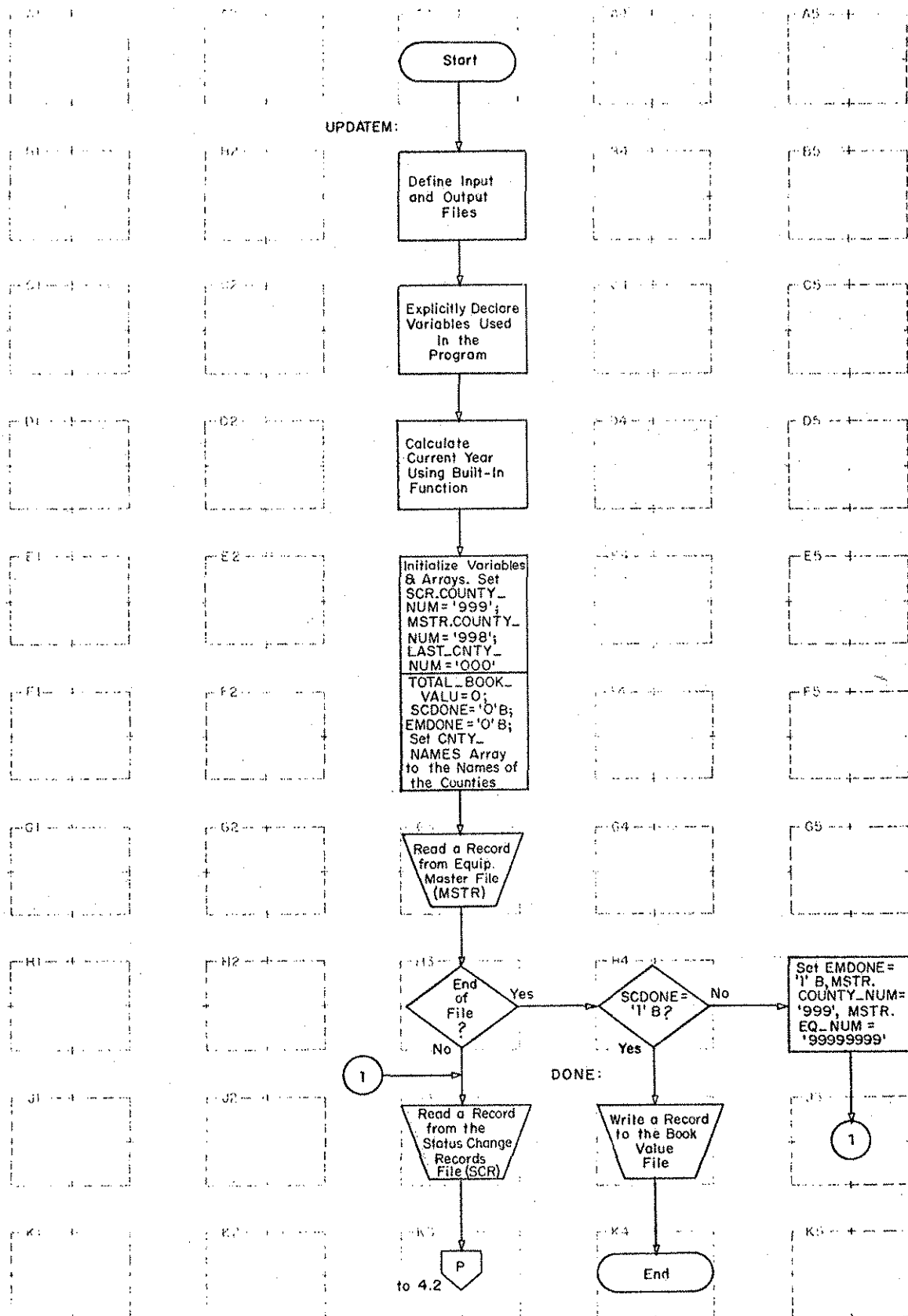
Programmer: Charles Sadoris Program No.: 3 Date: Page:
 Chart ID: 3.4 Chart Name: Program Name: STCHNGE

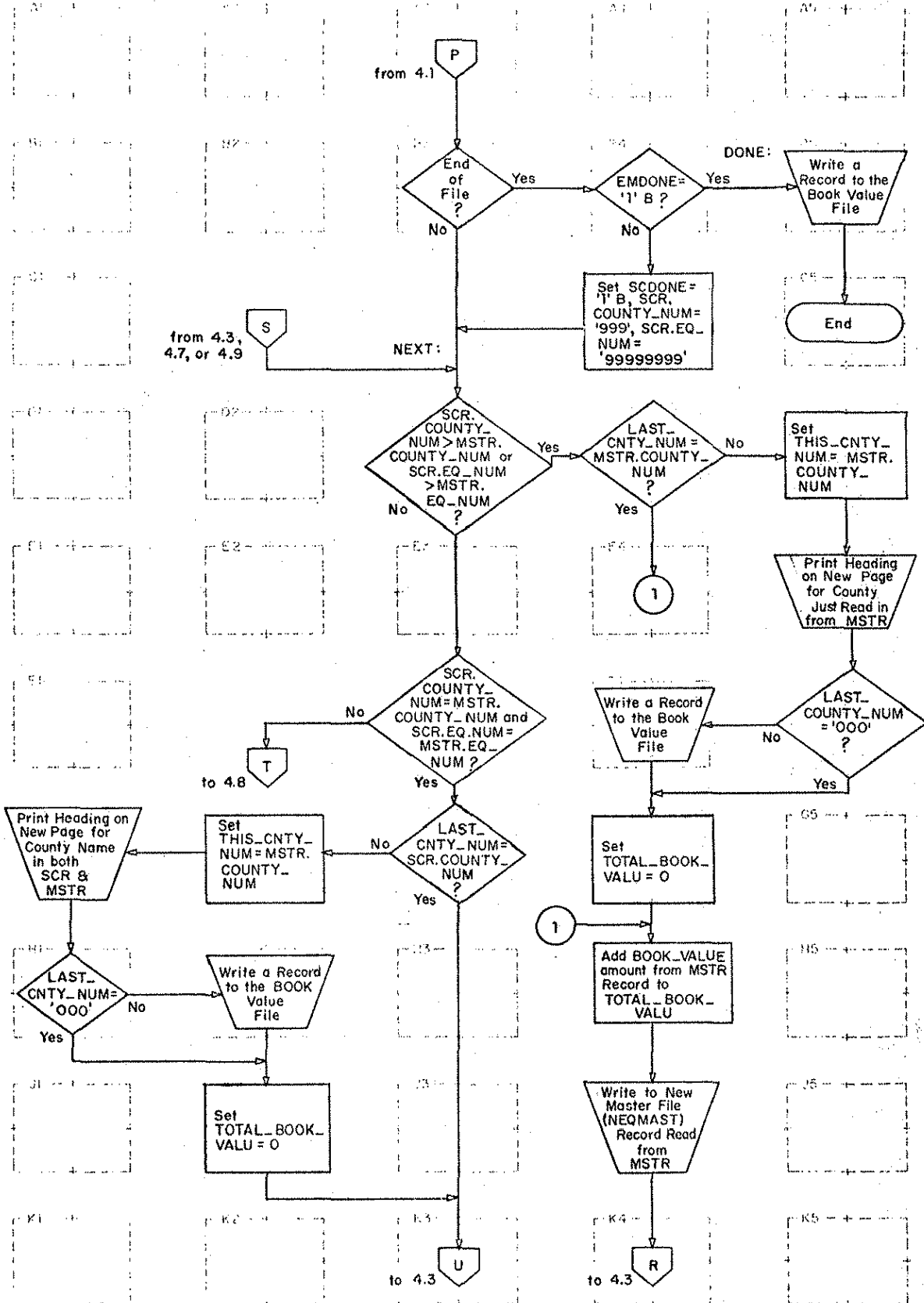


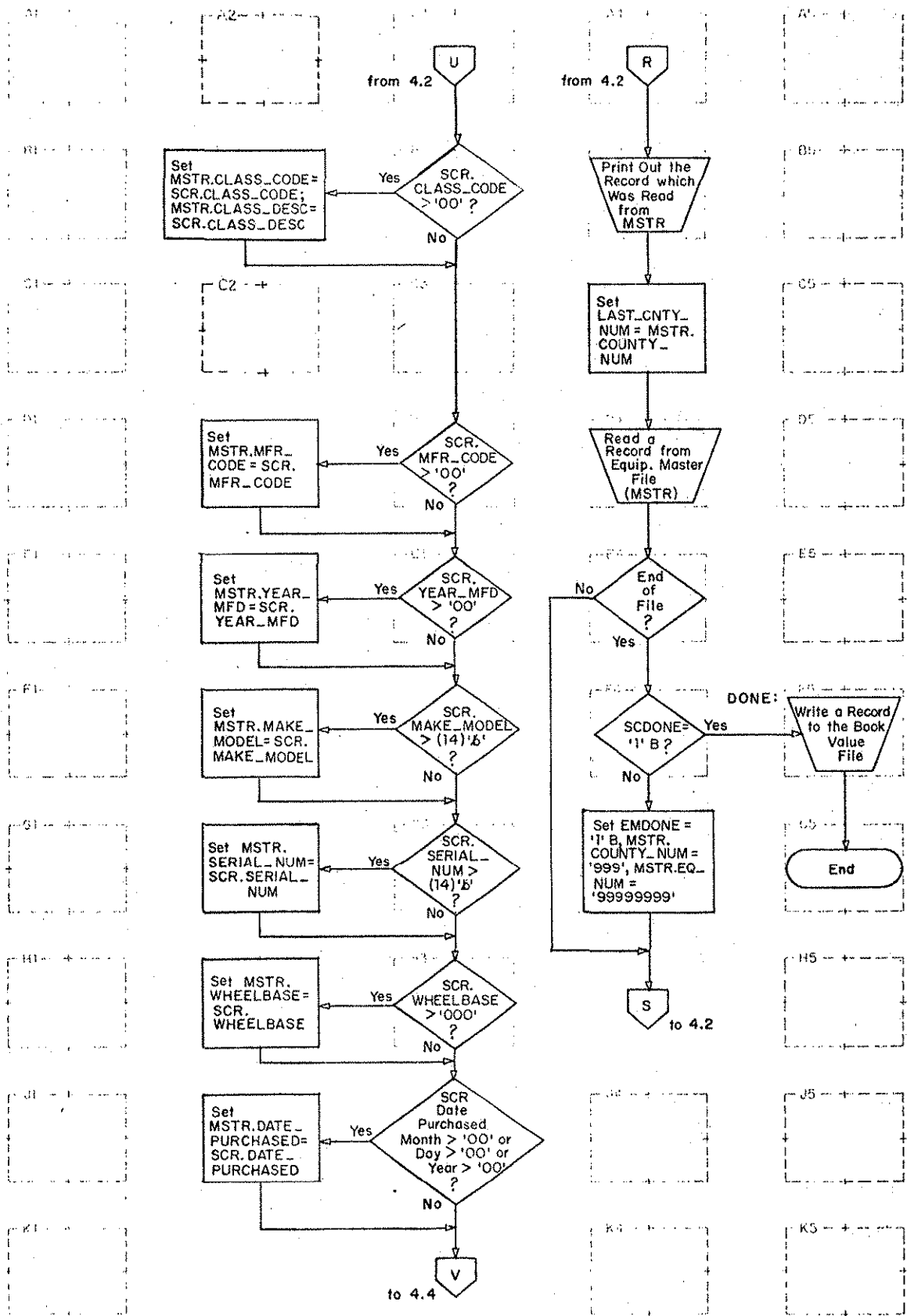


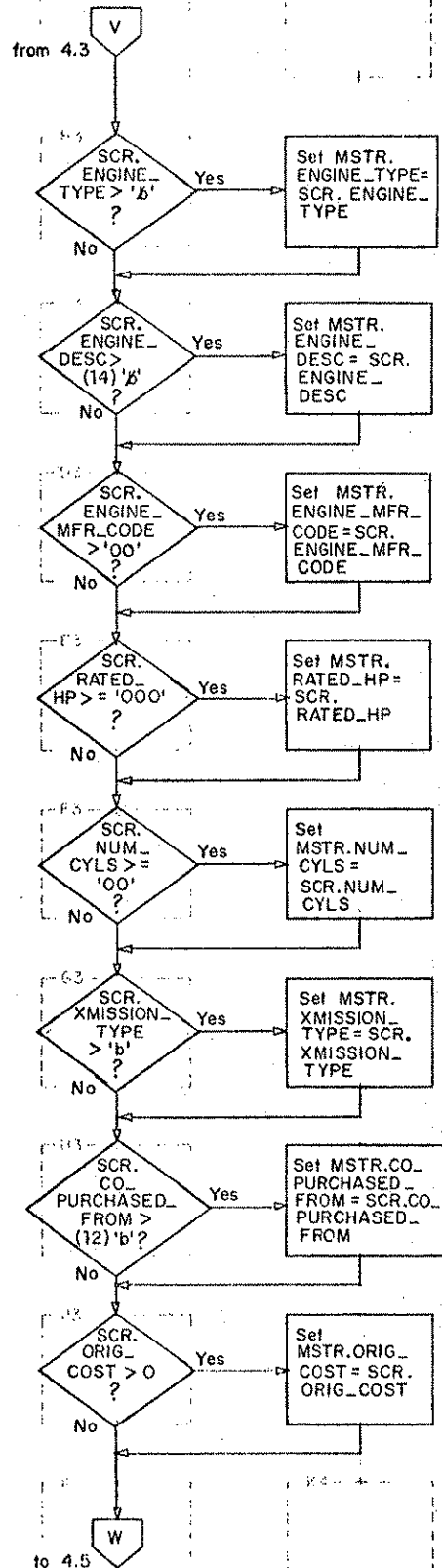


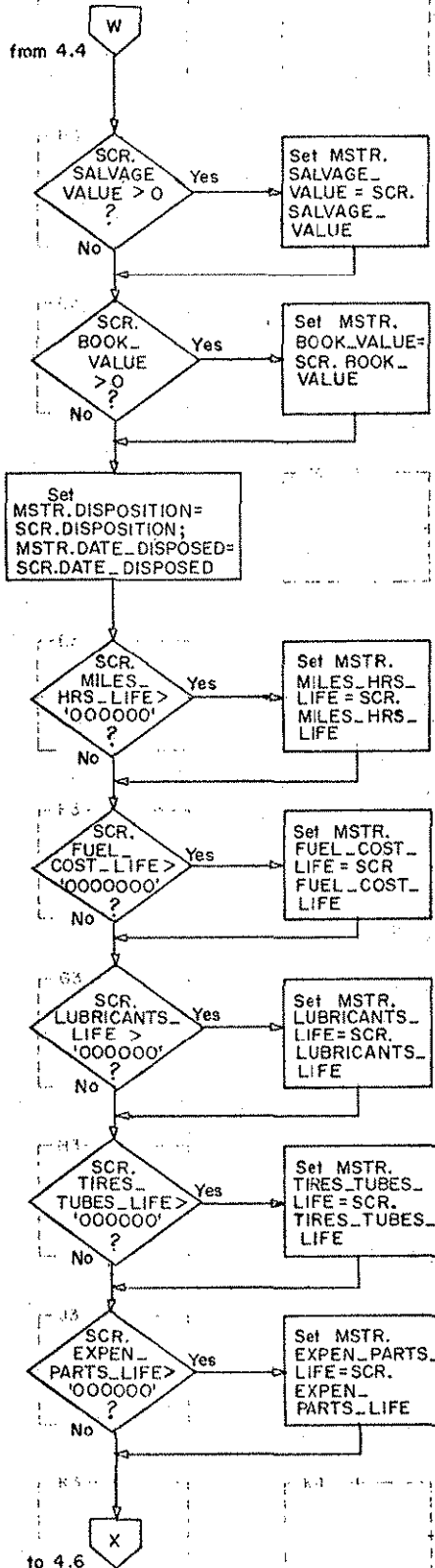
Programmer: Rolf McHenry Program No.: 4 Date: July 1975 Page: _____
Chart ID: 4.1 Chart Name: Add Status Change Records to Master File, and Calculate Book Value by County Program Name: UPDITEM

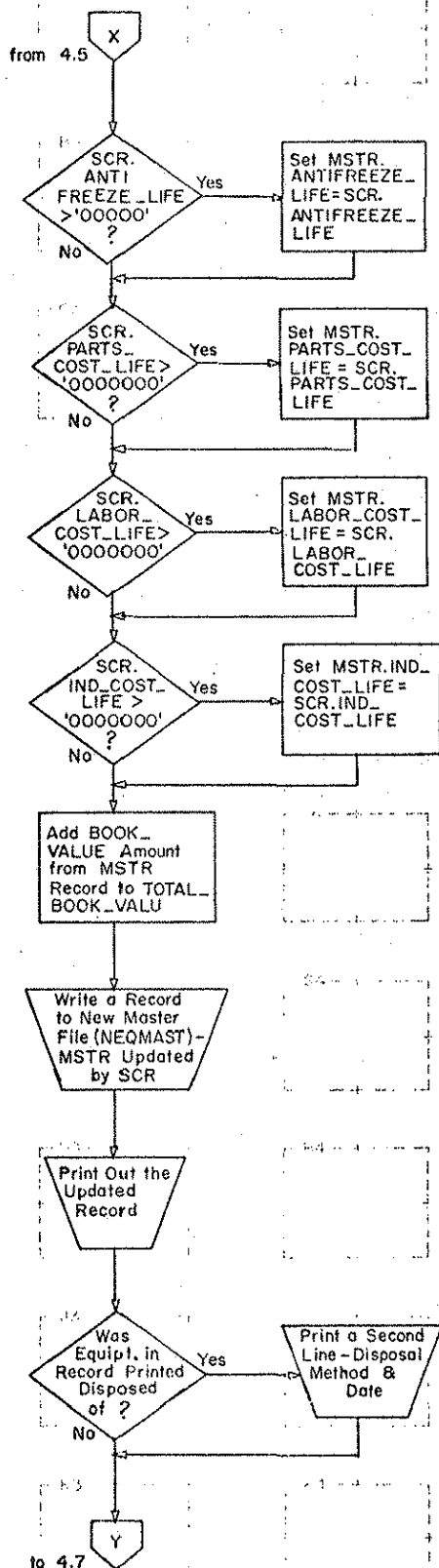




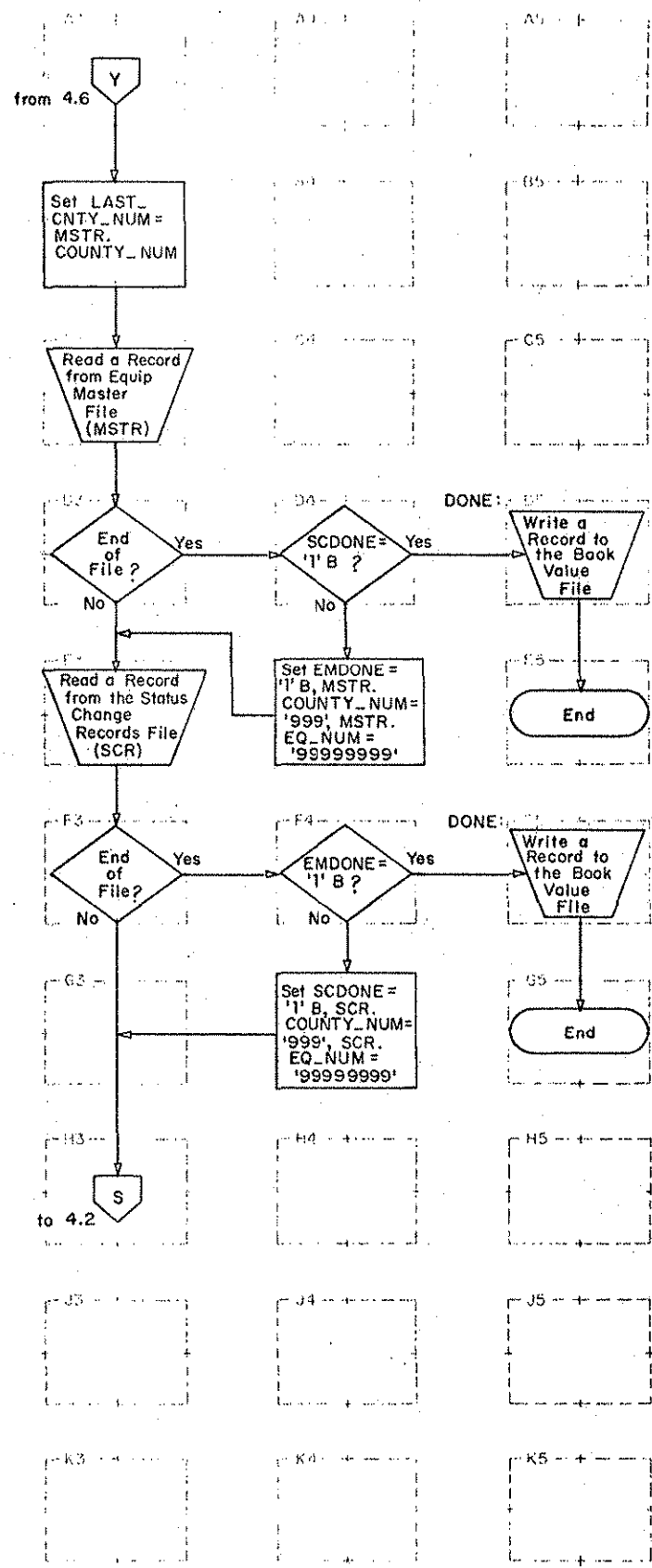


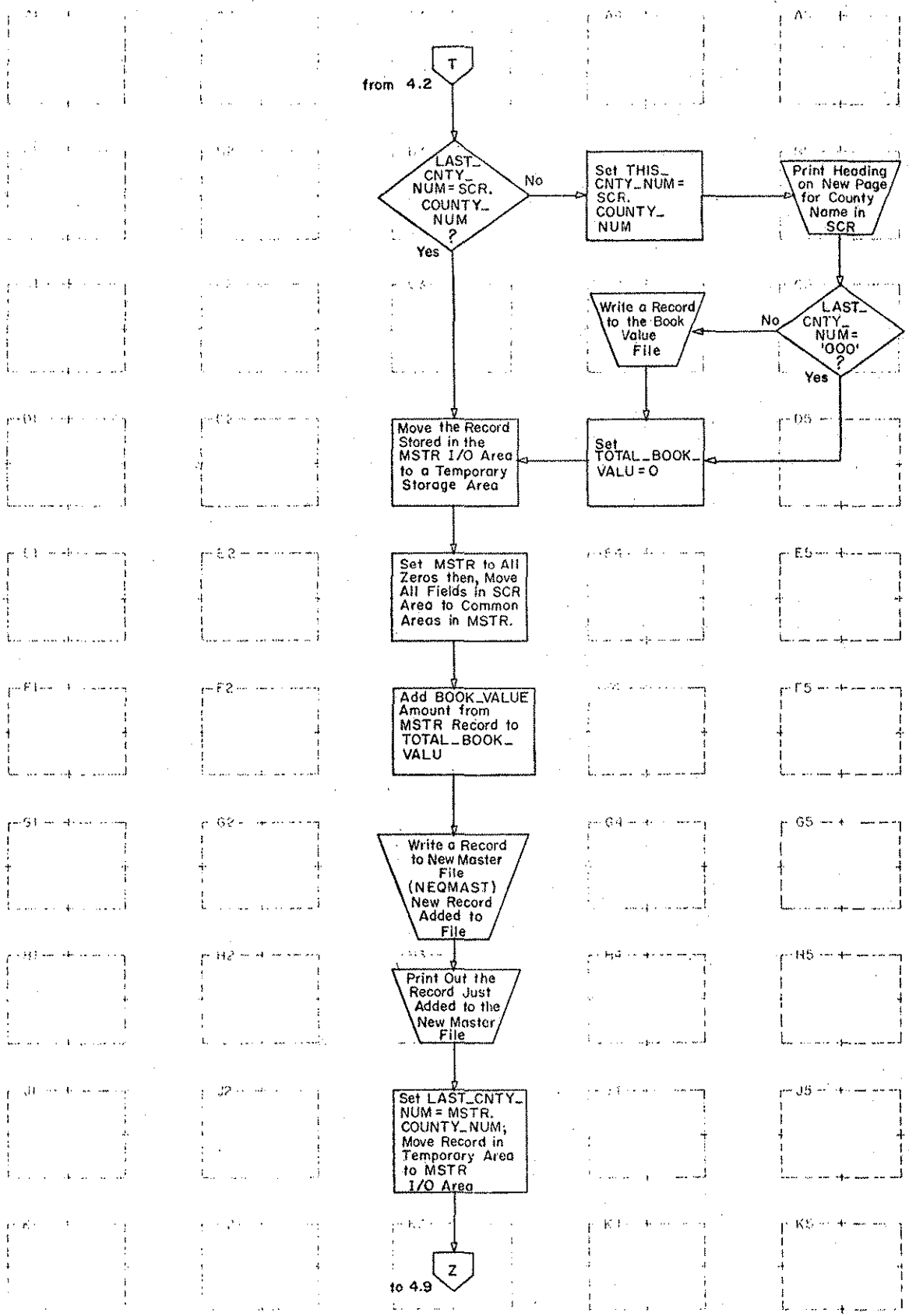




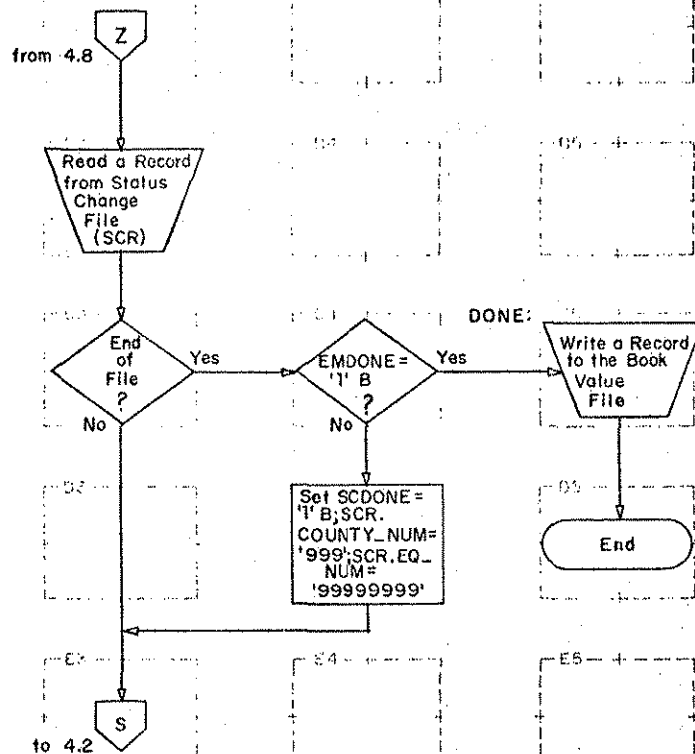


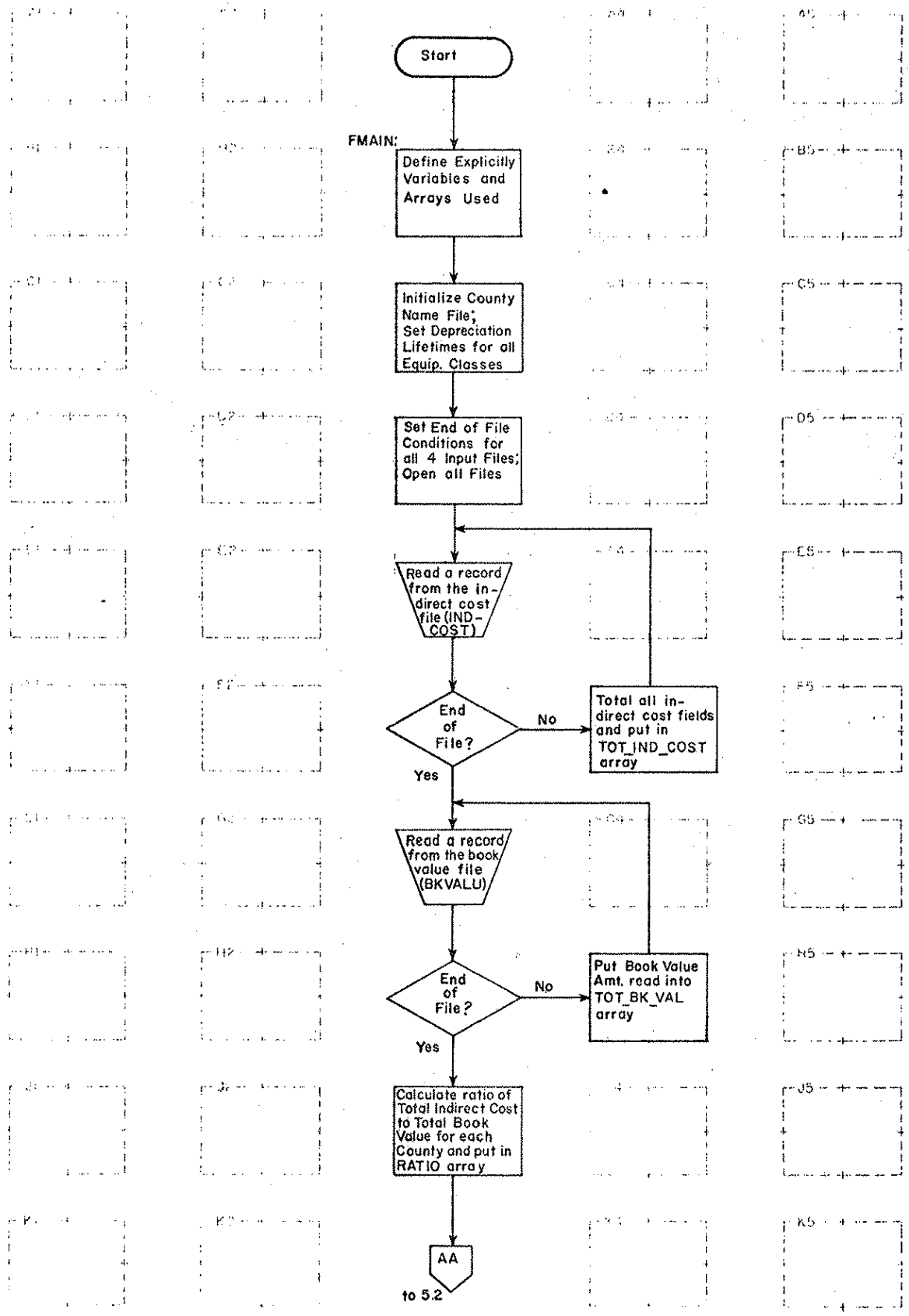
Programmer: Rolf McHenry Program No.: 4 Date: _____ Page: _____
Chart ID: 4.7 Chart Name: _____ Program Name: UPDATM

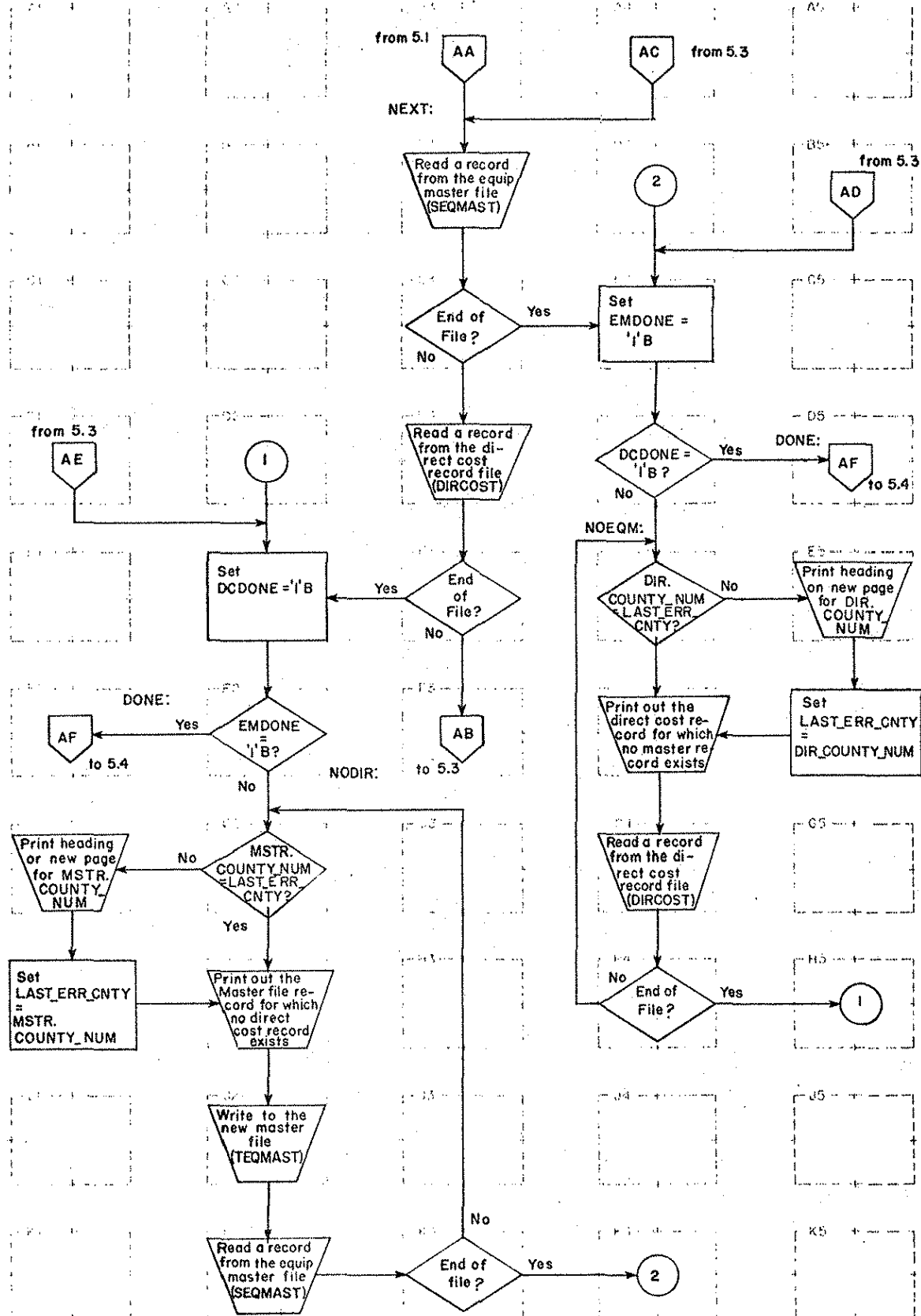




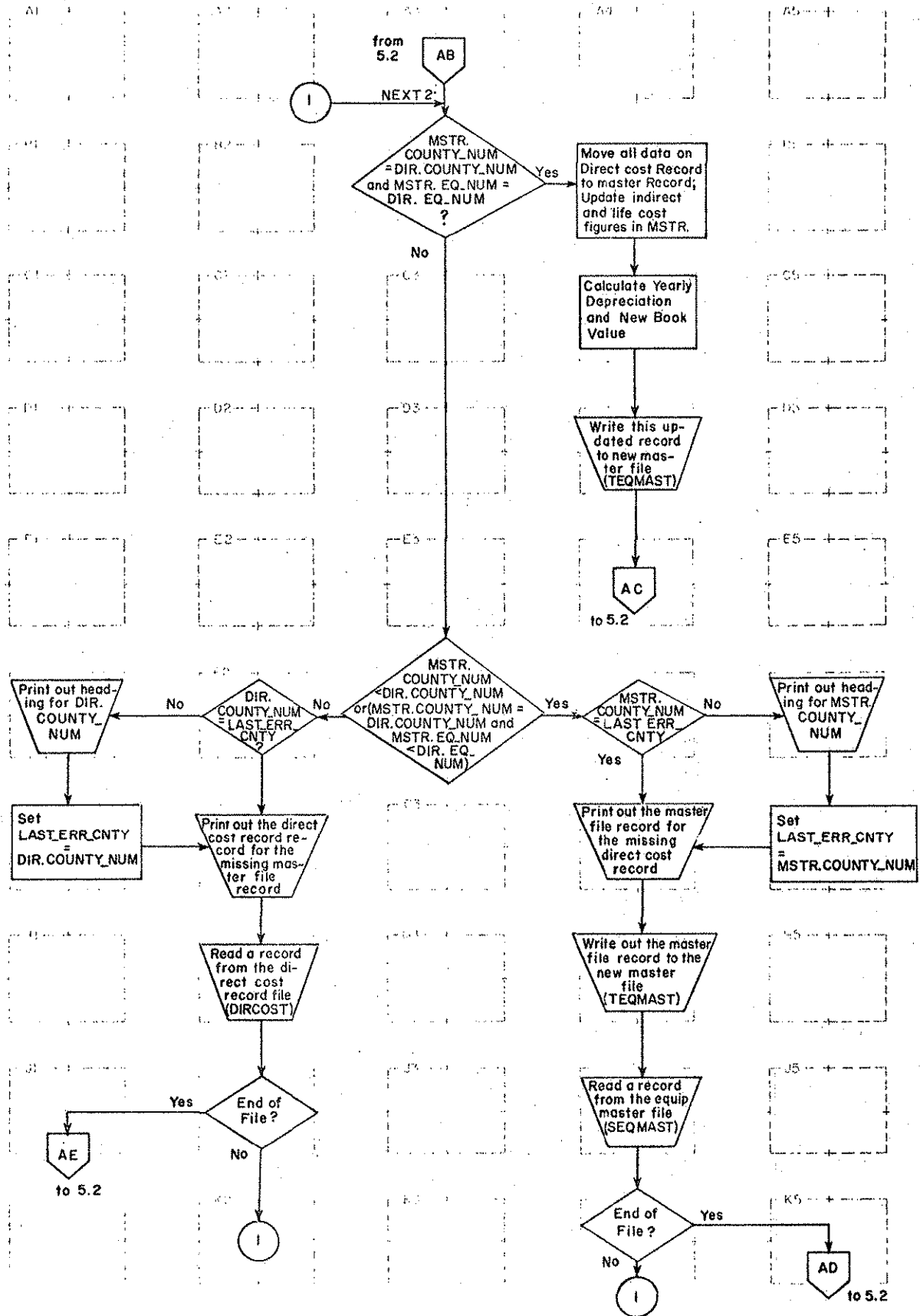
Programmer: Rolf McHenry Program No.: 4 Date: _____ Page: _____
Chart ID: 4.9 Chart Name: _____ Program Name: UPDATEM



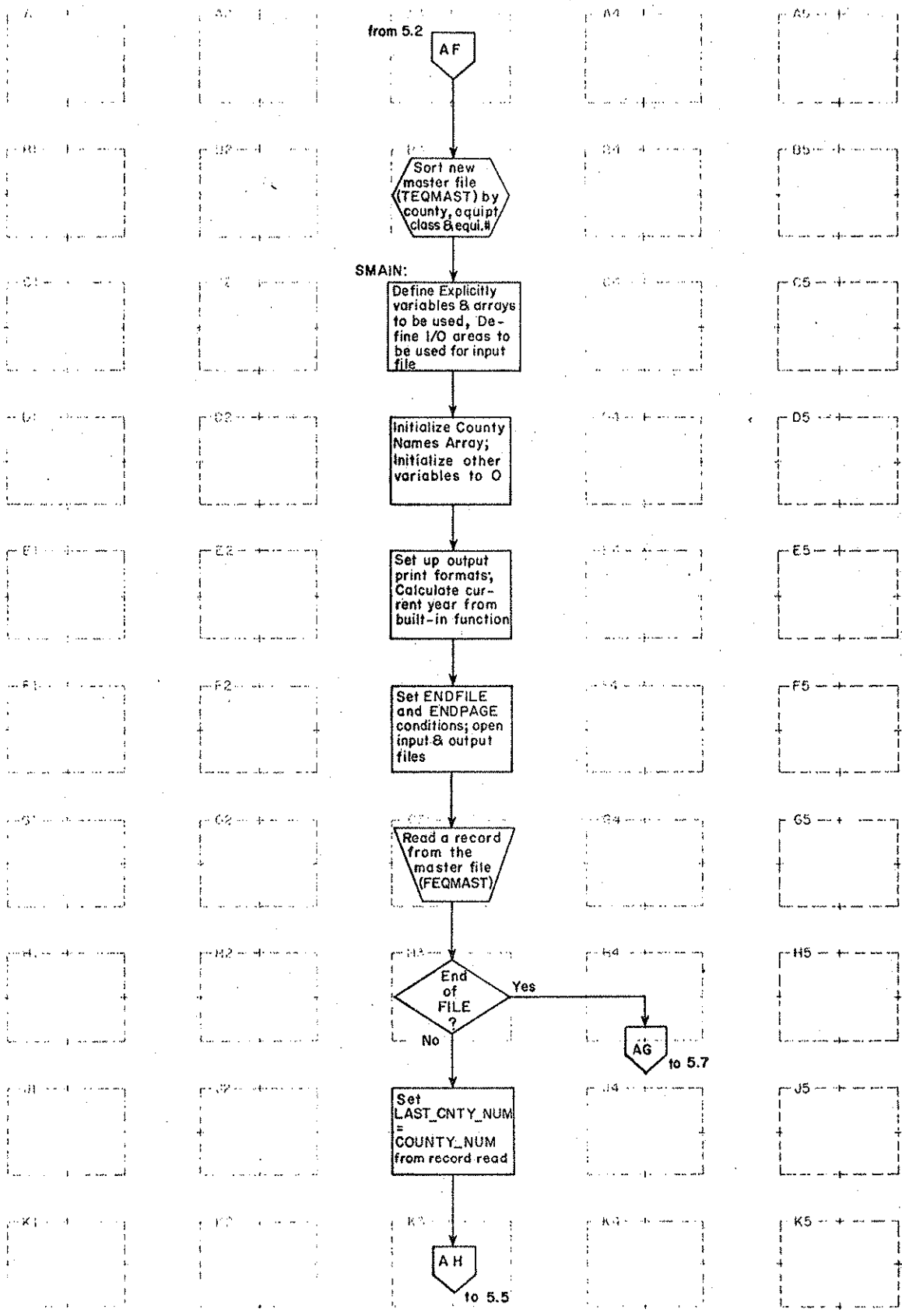


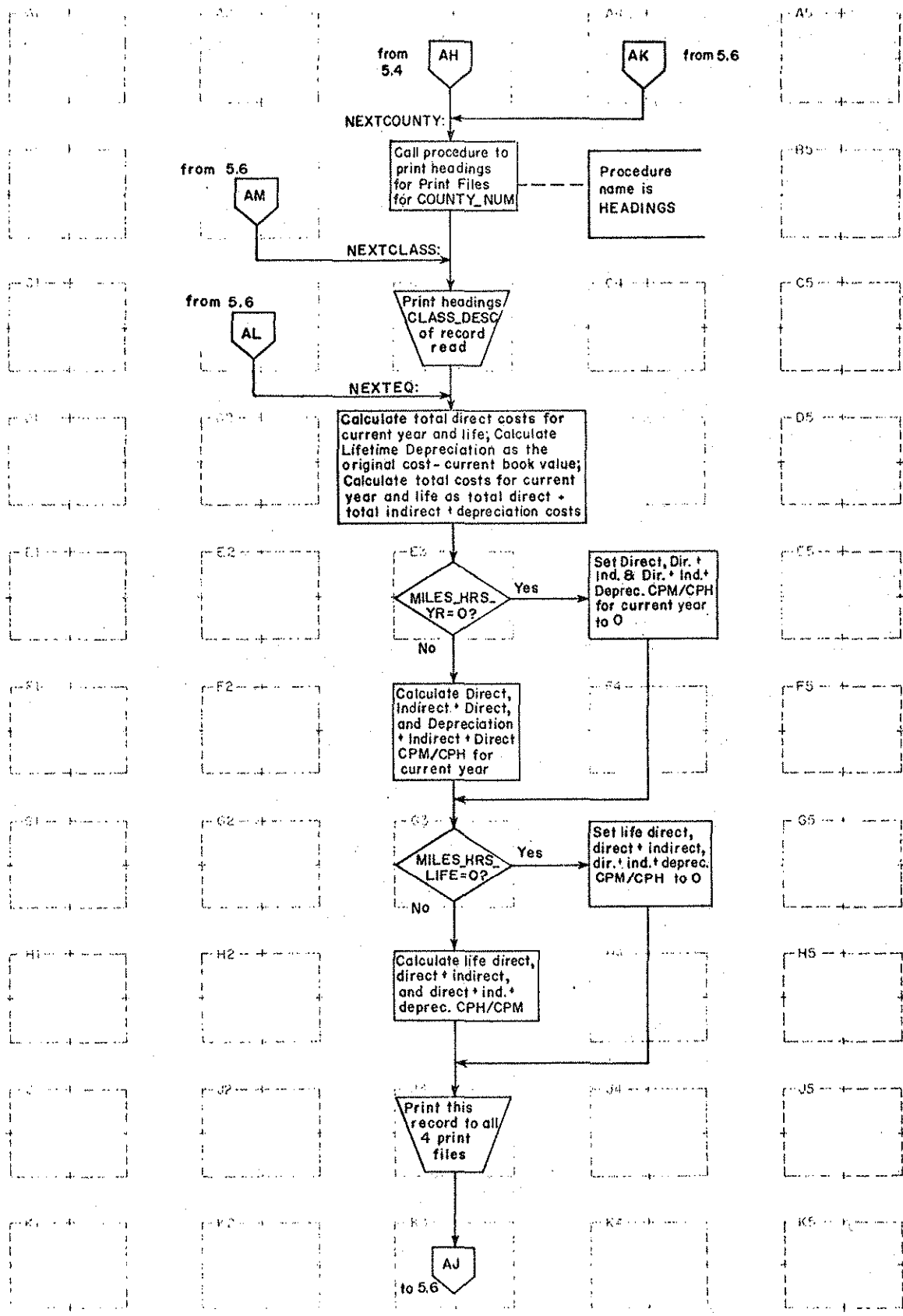


Programmer: Rolf McHenry Program No.: 5 Date: _____ Page: _____
 Chart ID: 5.3 Chart Name: _____ Program Name: MAIN

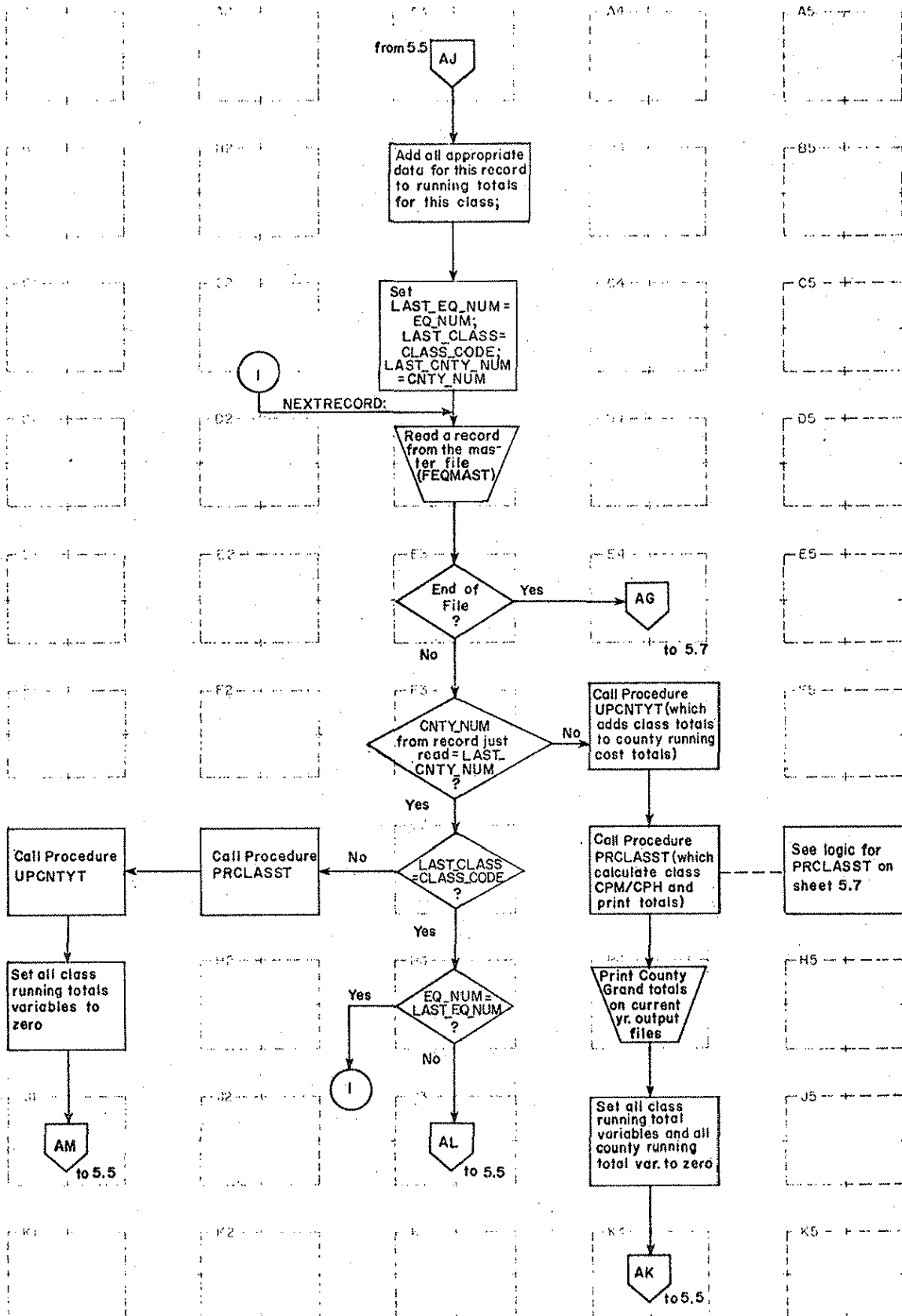


Programmer: Rolf McHenry Program No.: 5 Date: _____ Page: _____
Chart ID: 5.4 Chart Name: _____ Program Name: MAIN





Programmer: Rolf McHenry Program No.: 5 Date: _____ Page: _____
 Chart ID: 5.6 Chart Name: _____ Program Name: MAIN



Programmer: Rolf McHenry Program No.: 5 Date: _____ Page: _____
 Chart ID: 5.7 Chart Name: _____ Program Name: MAIN

Procedure
PRCLASST

Start

CLASS_MLHRS = 0?

Yes
Set the whole class direct cost, direct + indirect cost, & direct + indirect + depreciation CPM/CPH to 0

No
Calculate Class direct, direct + indirect, & direct + ind. + deprec. CPM/CPH for the current year

CLASS_MLHRS_L = 0?

Yes
Set the class direct cost, direct + indirect cost, & direct + indirect + deprec. cost CPM/CPH life figures to 0

No
Calculate class direct + indirect, & direct + indirect + deprec. CPM/CPH for the life figures

Print Class Totals and CPM/CPH on all four output print files

End

from 5.4 & 5.6
AG

DONE:

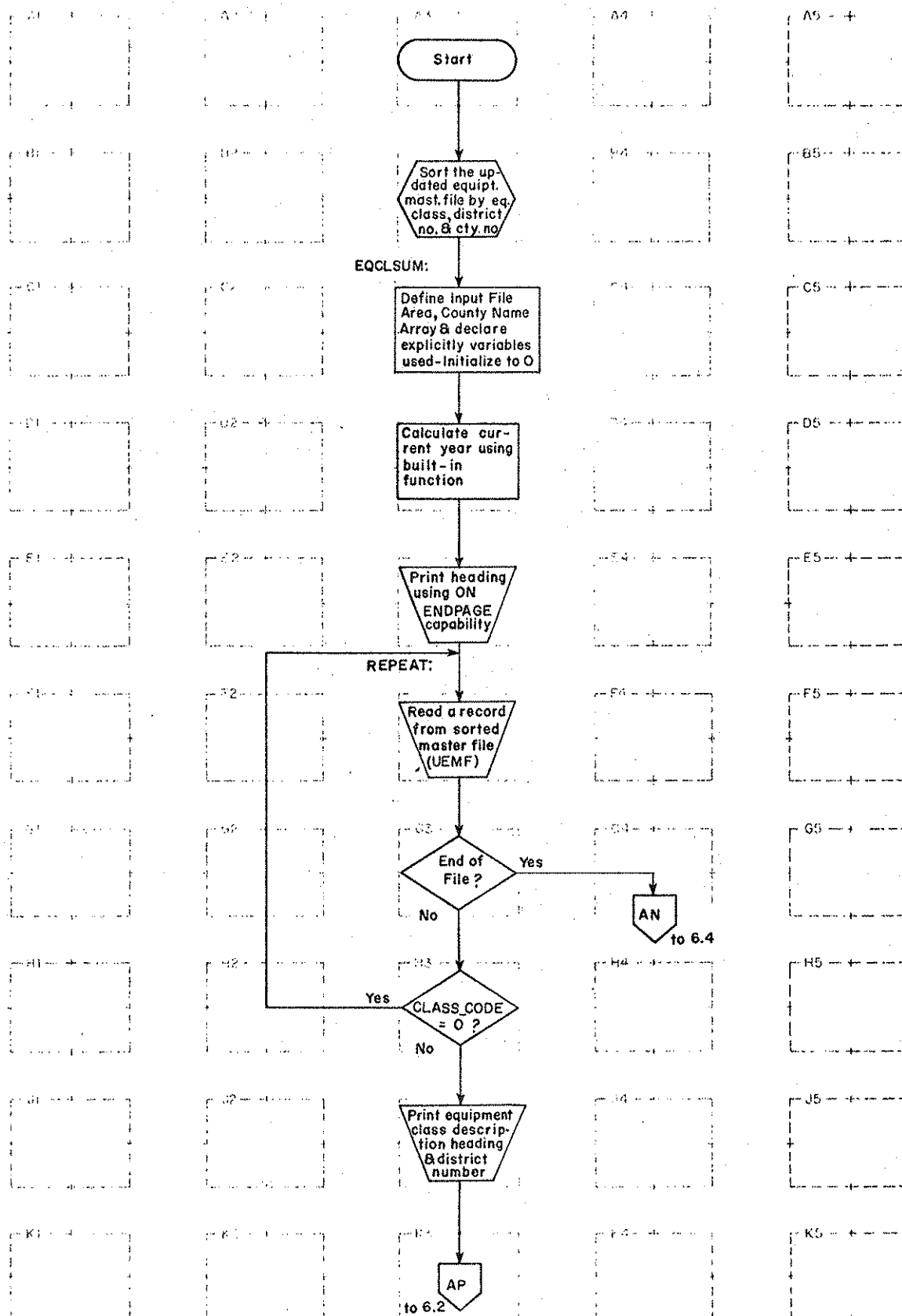
Call Procedure UPCNTYT

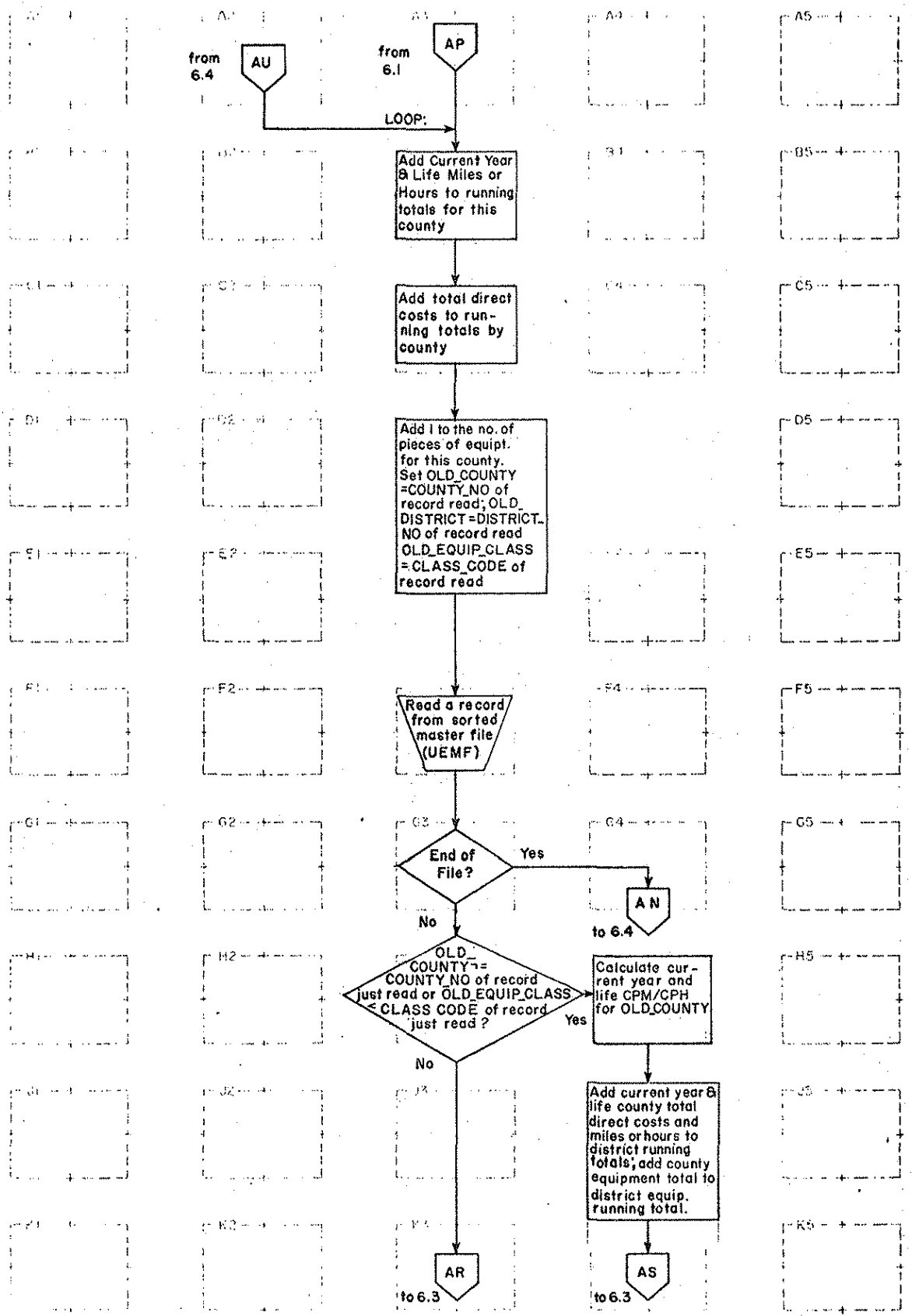
Call Procedure PRCLASST

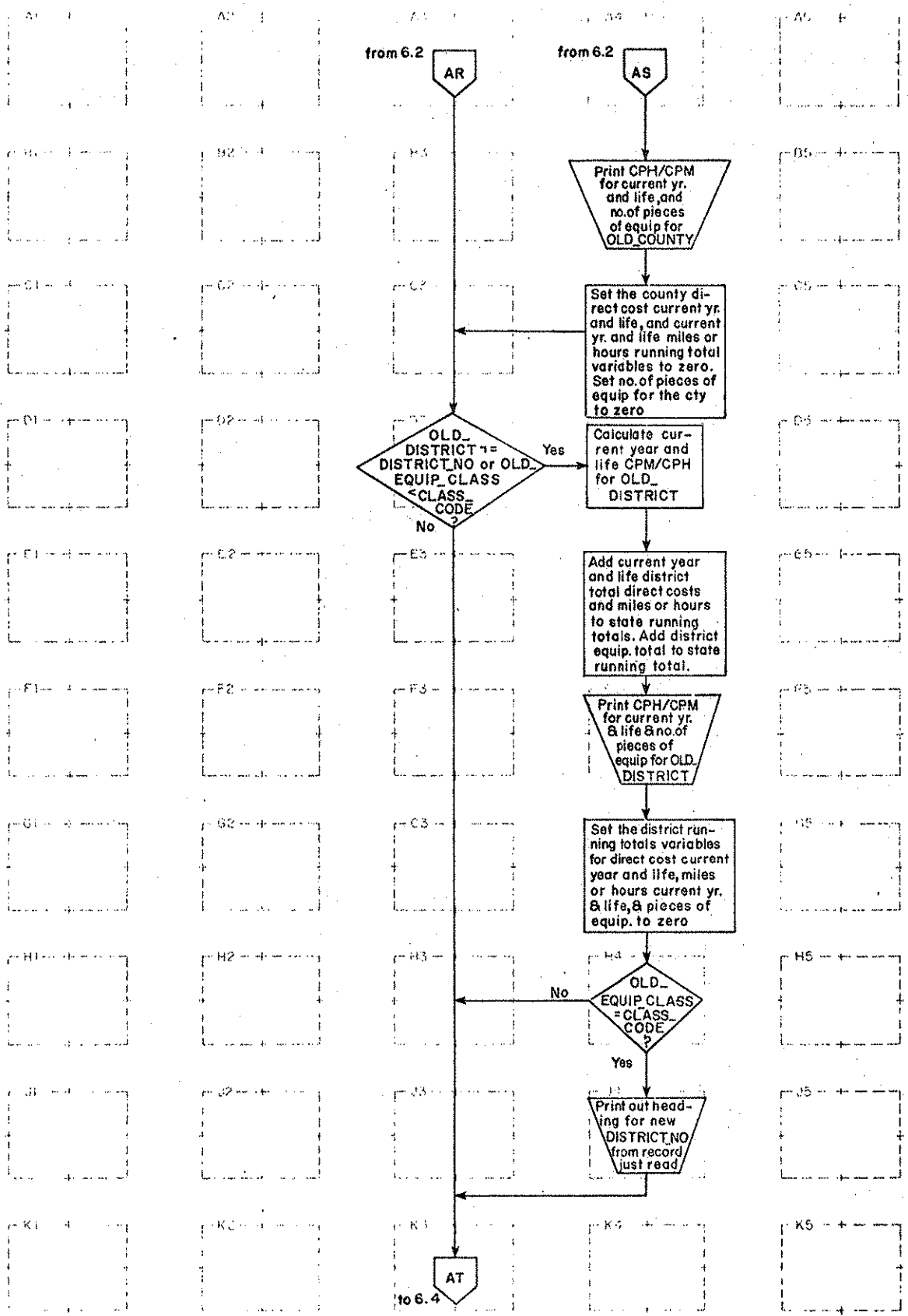
Print County Grand Totals on current yr. output files

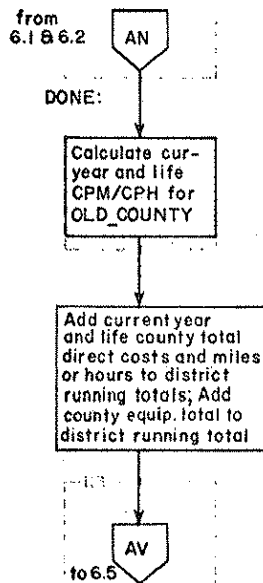
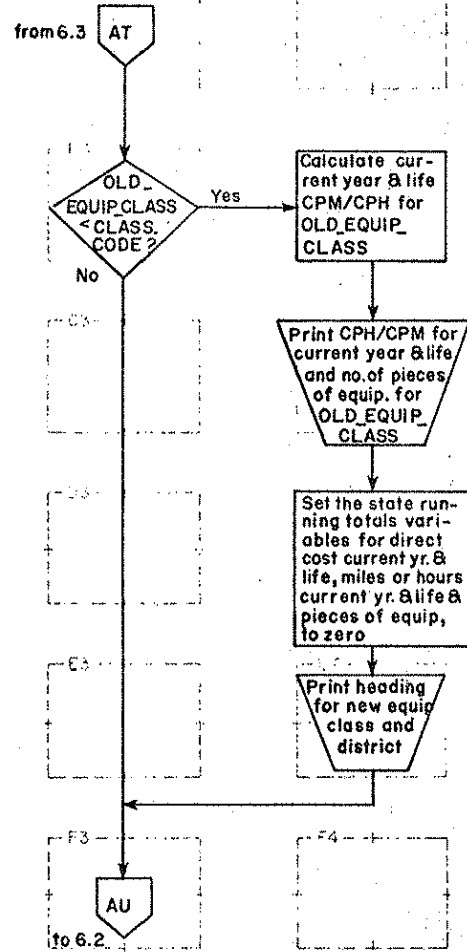
End

Programmer: John Poyzer Program No.: 6 Date: _____ Page: _____
Chart ID: 6.1 Chart Name: CPM/CPH Summary by Equipment Class District and County Program Name: CTYSUMRY

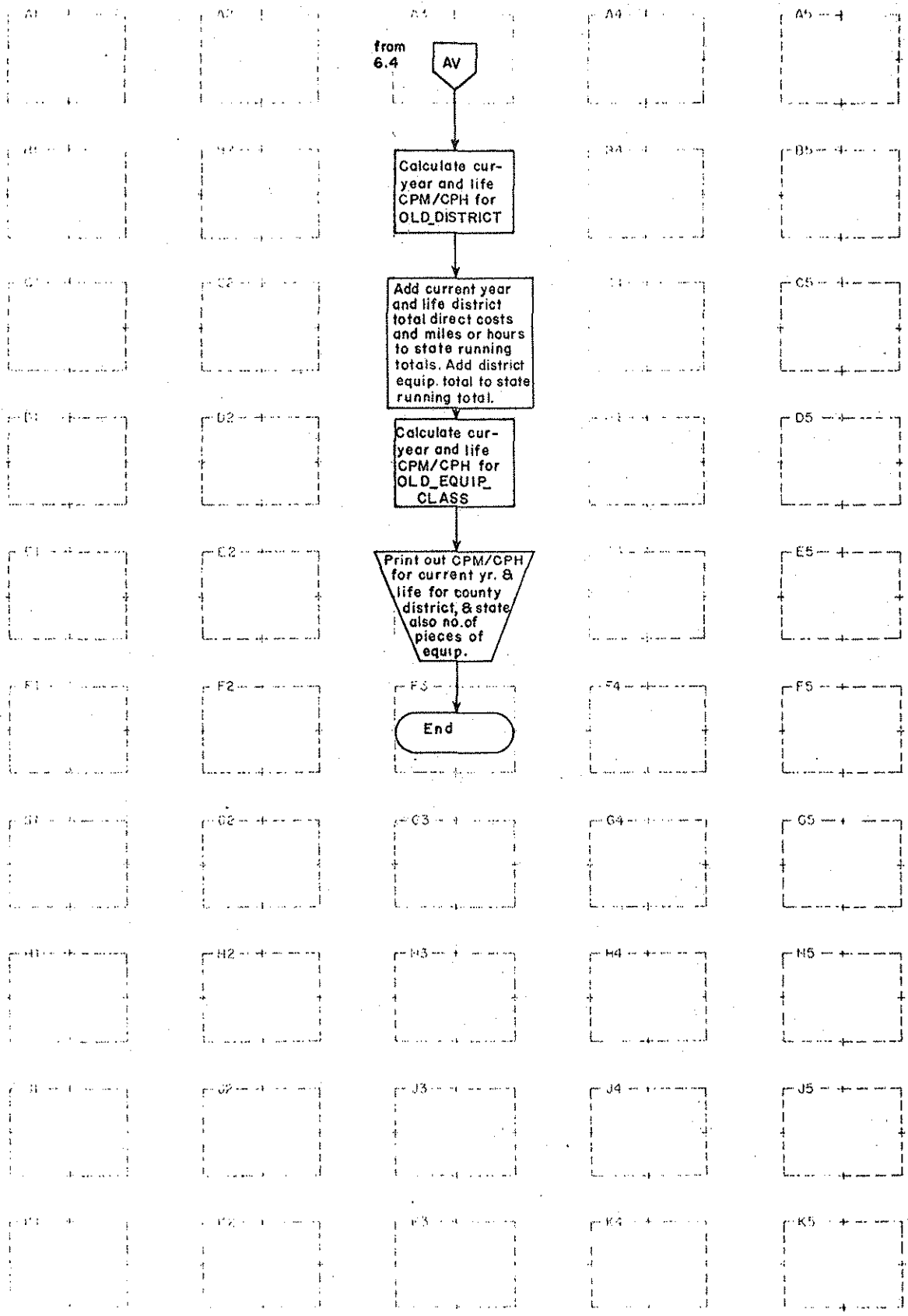


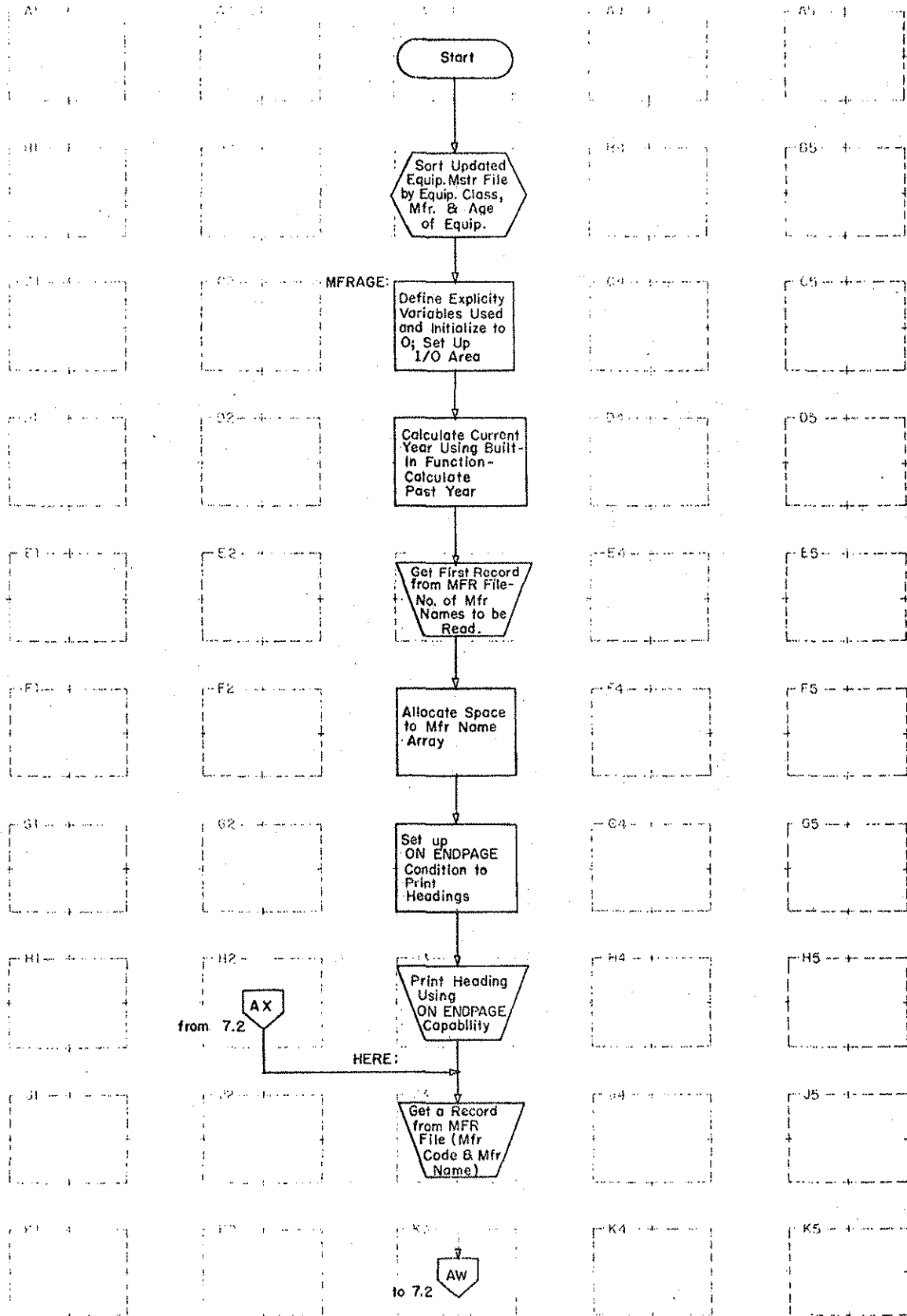




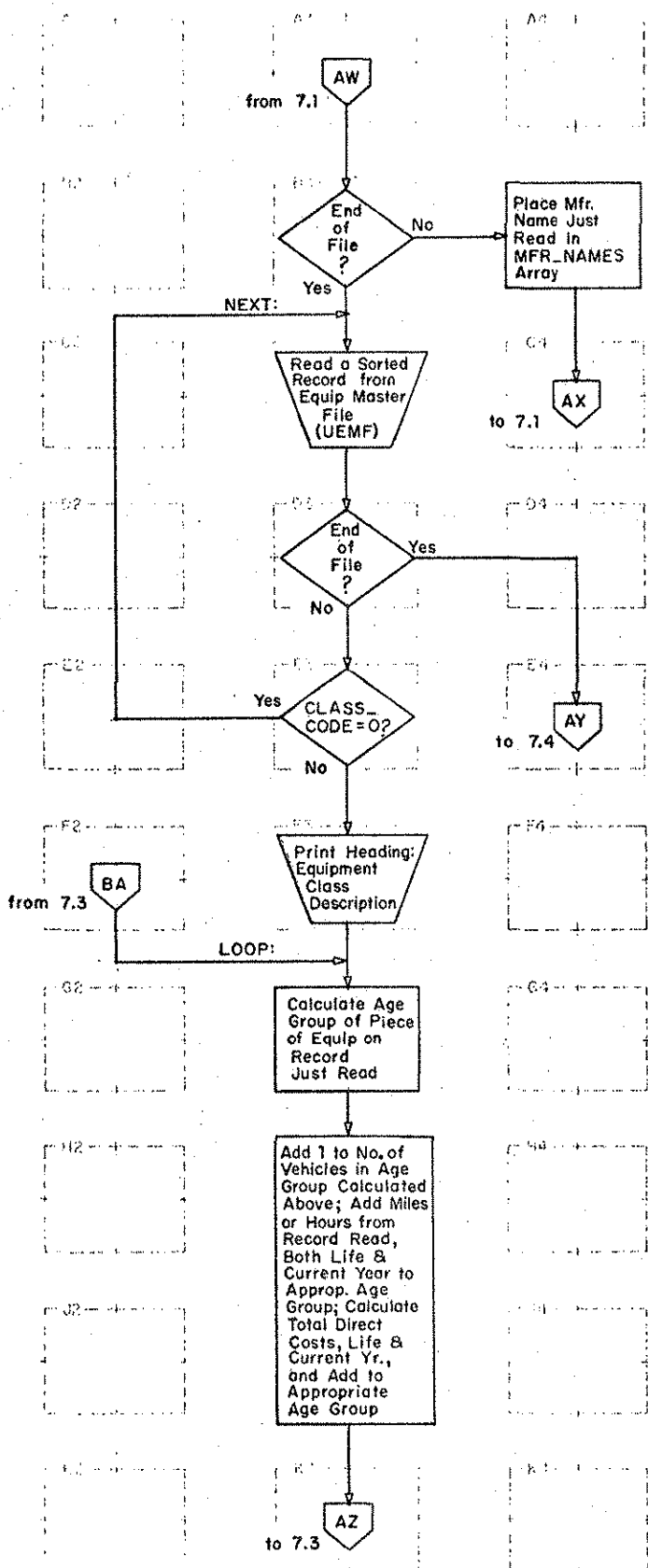


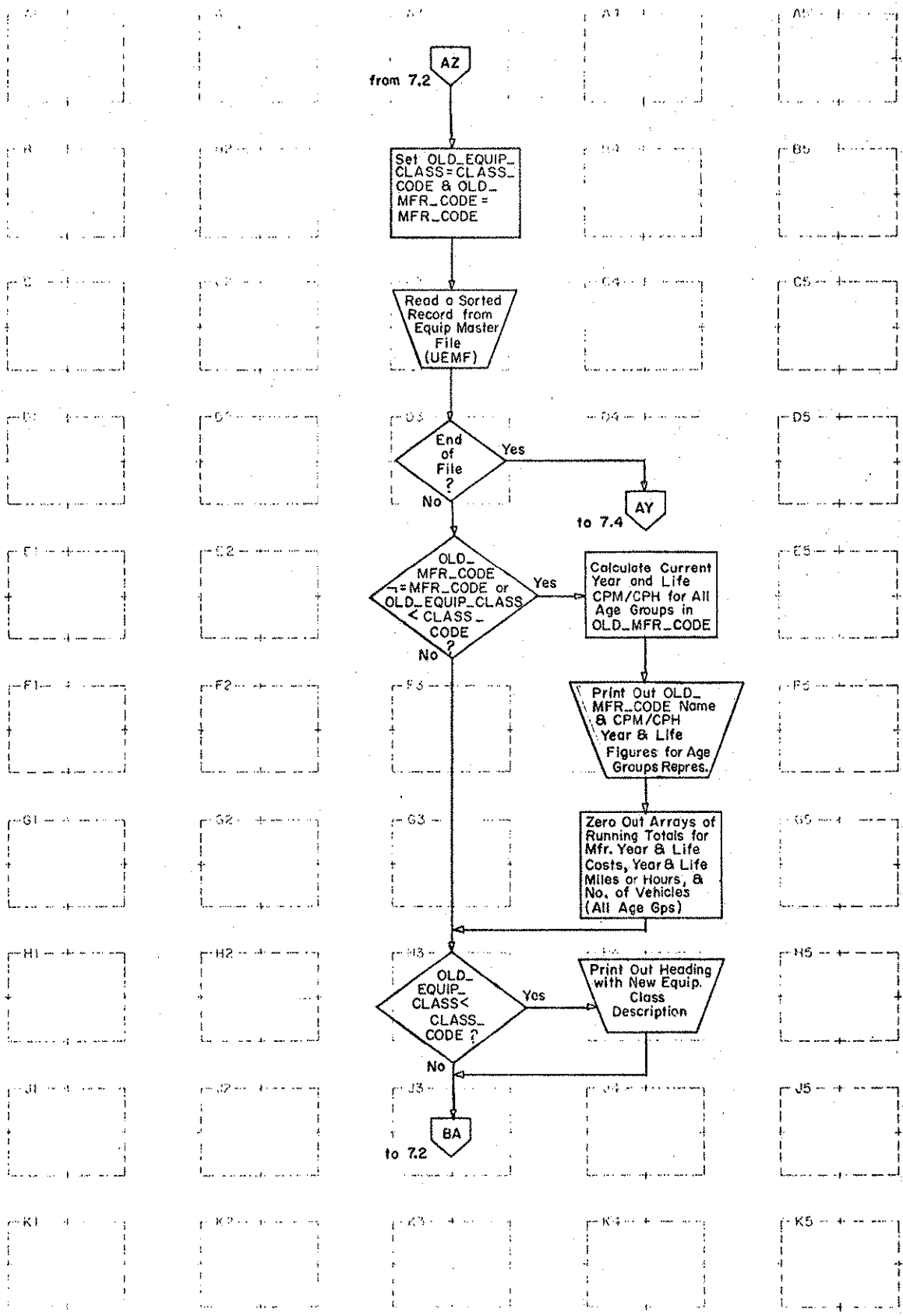
Programmer: John Poyzer Program No.: 6 Date: _____ Page: _____
Chart ID: 6.5 Chart Name: _____ Program Name: CTYSUMRY

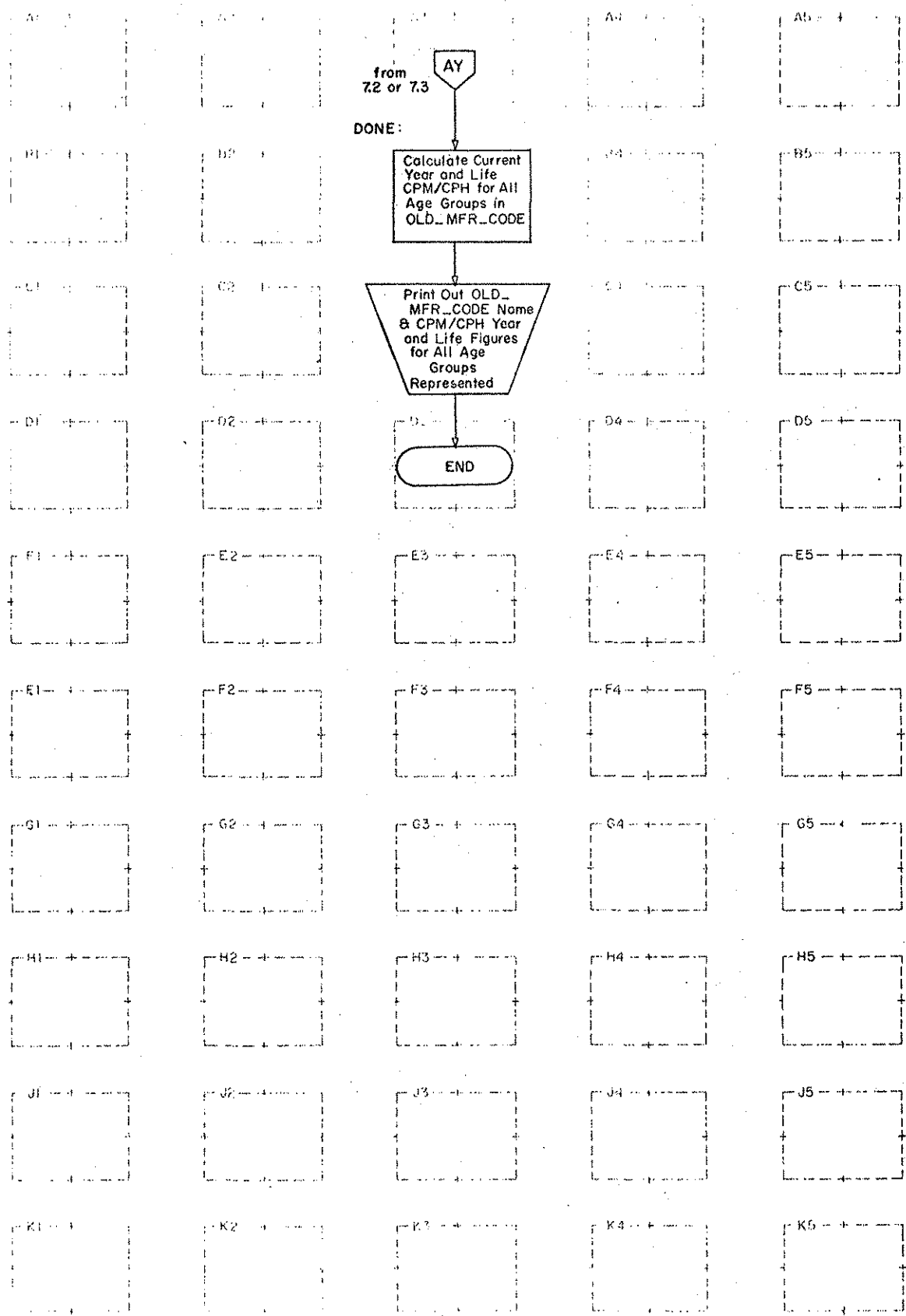




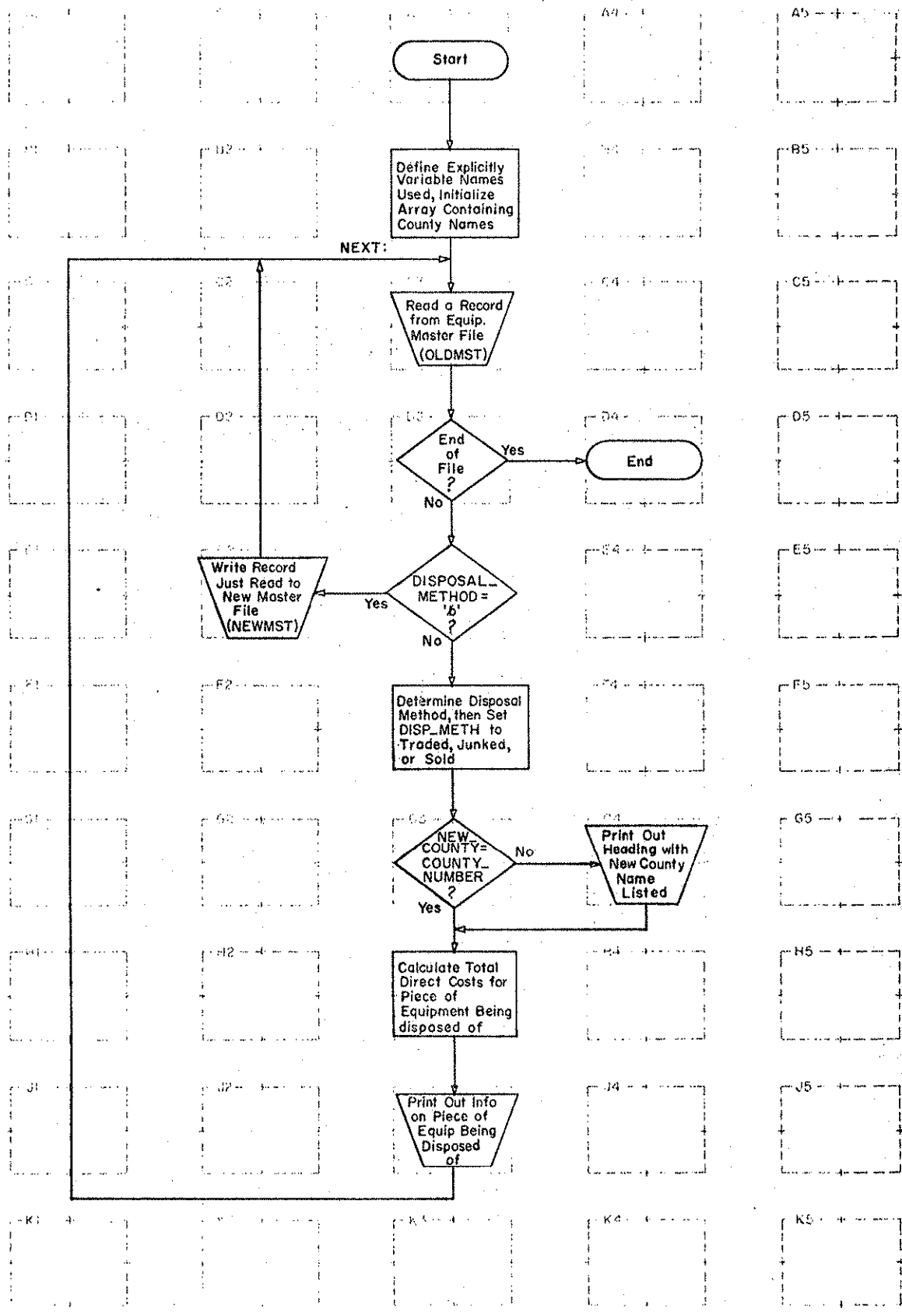
Programmer: John Poyzer Program No.: 7 Date: _____ Page: _____
Chart ID: 7.2 Chart Name: _____ Program Name: MFGAGE







Programmer: Charles Sadoris Program No.: B Date: Page:
Chart ID: 8.1 Chart Name: Remove Disposed Equipment From Master File Program Name: UPDATE



COMPUTER PROGRAM LISTINGS

This section contains the computer program listing of each of the eight computer programs used in this system. All programs are written in the PL/1 programming language. DIRECT, INDCOST, STCHNGE, CTYSUMRY, and MFGAGE are two-step jobs, the first step being a sort performed by a utility sort program. MAIN is a three step job, the first step and third step are programs written in PL/1, with the second step being a utility sort program. The sort routine portions of these programs are not given in these listings, however, the necessary sort sequences are given in each of the Computer Operating Instructions in the first section of this Appendix.

Listing of Program DIRECT

STAT LEVEL NEST

1

ECCR: PROC OPTICNS(MAIN):

```
/* PROGRAM DIRECT */
/*
/* THIS PROGRAM READS ALL DIRECT COST SUMMARY FORM CARDS,
/* CHECKS THEM FOR ERRORS, AND OUTPUTS THE CORRECT RECORDS
/* TO THE DIRECT COST FILE. ERRORS FOR EACH COUNTY ARE
/* LISTED.
/*
/* THIS PROGRAM IS ONE OF EIGHT COMPUTER PROGRAMS
/* WHICH FORM THE "COMPUTER BASED INFORMATION SYSTEM
/* FOR COUNTY EQUIPMENT COST RECORDS."
/*
/* WRITTEN BY
/*
/* SYSTEMS DIVISION
/* COLLEGE OF ENGINEERING
/* THE UNIVERSITY OF IOWA
/* IOWA CITY, IOWA
/* JULY, 1975
/*
```

SYMT LEVEL NEST

```
2 1 DECLARE SDCOST FILE RECORD SEQUENTIAL;
3 1 DECLARE ERROR FIXED DEC(5) INIT(0);
4 1 DECLARE FLAG FIXED DEC(5) INIT(2);
5 1 DECLARE LAST_CNTY_NO CHAR(3) INIT('AAA');

6 1 DCL 1 DIRECT_COST_SUMMARY_RECORDS,
      2 COUNTY_NUMBER CHAR(3),
      2 EQUIPMENT_NUMBER CHAR(8),
      2 F_DATE,
        3 MONTH CHAR(2),
        3 YEAR CHAR(2),
      2 DIRECT_COSTS CHAR(50);

7 1 DECLARE DCOST FILE RECORD SEQUENTIAL;
8 1 DECLARE COLUMN REAL FIXED BIN(15,0);
9 1 DECLARE 1 C_DATE,
      2 C_YEAR CHAR(2),
      2 MON_DAY CHAR(4);

10 1 ON ENDFILE(SDCOST) GO TO DONE;

12 1 OPEN FILE(SDCOST) INPUT;
13 1 OPEN FILE(DCOST) OUTPUT;
14 1 C_DATE = DATE;
```

STMT LEVEL NEST

```

15      1      NEXT:
        REAC FILE(SDCOST) INTO (DIRECT_COST_SUMMARY_RECORDS);

16      1      IF LAST_CNTY_NO = COUNTY_NUMBER
17      1      THEN DO;
18      1      1      IF FLAG = 0
19      1      1      THEN PUT SKIP(3) EDIT ('NO ERRORS FOUND')(X(5),A);
20      1      1      PUT PAGE EDIT('ERROR LISTING FOR DIRECT COST SUMMARY FORMS',
        ' - COUNTY CODE ',COUNTY_NUMBER)(CCL(32),A,A,A(3));
21      1      1      PUT SKIP(3);
22      1      1      LAST_CNTY_NO = COUNTY_NUMBER;
23      1      1      FLAG = 0;
24      1      1      END;

25      1      IF COUNTY_NUMBER < '001' | COUNTY_NUMBER > '099'
26      1      THEN DO;
27      1      1      PUT SKIP EDIT('THE FOLLOWING RECORD HAS AN ILLEGAL COUNTY CODE.')
        (X(5),A);
28      1      1      ERROR = 1;
29      1      1      END;

30      1      IF MONTH > '12'
31      1      THEN DO;
32      1      1      PUT SKIP EDIT('THE FOLLOWING RECORD HAS AN INVALID MONTH VALUE')
        (X(5),A);
33      1      1      ERROR = 1;
34      1      1      END;

35      1      IF YEAR > C_YEAR
36      1      THEN DO;
37      1      1      PUT SKIP EDIT('THE FOLLOWING RECORD HAS AN INVALID YEAR VALUE.')
        (X(5),A);
38      1      1      ERROR = 1;
39      1      1      END;

40      1      COLUMN = VERIFY(DIRECT_COSTS,' 0123456789') + 15;

41      1      IF COLUMN = 15
42      1      THEN DO;
43      1      1      PUT SKIP EDIT('THE FOLLOWING RECORD HAS AN ILLEGAL CHARACTER',
        ' AT POSITION ',COLUMN,'.')(X(5),A,A,F(2),A);
44      1      1      ERROR = 1;
45      1      1      END;

46      1      IF ERROR = 1
47      1      THEN DO;
48      1      1      PUT SKIP(2) EDIT (DIRECT_COST_SUMMARY_RECORDS)(X(3),(5) A);
49      1      1      PUT SKIP(2);
50      1      1      FLAG = 1;
51      1      1      ERROR = 0;
52      1      1      END;
53      1      ELSE
        WRITE FILE(DCOST) FROM(DIRECT_COST_SUMMARY_RECORDS);

54      1      GO TO NEXT;

```

STMT LEVEL NEST

```
55 1  DONE: IF FLAG = 0 THEN PUT SKIP(3) EDIT('NO ERRORS')(X(5),A1;  
57 1  END ECCR;
```


Listing of Program INDCOST

STMT LEVEL NEST

1

/* EDIT INDIRECT COST RECORDS */
EICR: PRCC OPTICKS(MAIN);

```
/* PROGRAM INDCOST */
/*
/* THIS PROGRAM READS ALL INDIRECT COST FORMS, CHECKS
/* THEM FOR ERRORS, AND OUTPUTS THE CORRECT RECORDS TO
/* THE INDIRECT COST FILE. ERRORS FOR EACH COUNTY ARE
/* LISTED.
/*
/* THIS PROGRAM IS ONE OF EIGHT COMPUTER PROGRAMS
/* WHICH FORM THE "COMPUTER BASED INFORMATION SYSTEM
/* FOR COUNTY EQUIPMENT COST RECORDS."
/*
/* WRITTEN BY
/*
/* SYSTEMS DIVISION
/* COLLEGE OF ENGINEERING
/* THE UNIVERSITY OF IOWA
/* IOWA CITY, IOWA
/* JULY, 1975
/*
/*
```

SYMT LEVEL NEST

```

2      1      DECLARE SICOST FILE RECORD SEQUENTIAL,
           1 INDIRECT_COST_RECORD,
           2 COUNTY_NUMBER CHAR(3),
           2 INDIRECT_COST CHAR(60);
3      1      DECLARE ICOST FILE RECORD SEQUENTIAL;
4      1      DECLARE COUNTY_CODE REAL FIXED BIN(15,0),
           COLUMN_REAL_REAL FIXED BIN(15,0);
5      1      DECLARE LAST_CNTRY_CD CHAR(3) INIT('AAA');
6      1      DECLARE ERROR FIXED DEC(5) INIT(1);

7      1      ON ENDFILE(SICOST) GO TO DCNE;

9      1      OPEN FILE(SICOST) INPUT;
10     1      OPEN FILE(ICOST) OUTPUT;
    
```

STMT LEVEL NEST

```

11      1      NEXT:
        READ FILE(SICCST) INTO (INDIRECT_COST_RECORD);

12      1      IF LAST_CNTY_NO = COUNTY_NUMBER
13      1      THEN DO:
14      1      1      IF ERROR = 0 THEN PUT SKIP EDIT ('NO ERRORS')(X(5),A);
16      1      1      PUT PAGE EDIT ('ERROR LISTING FOR INDIRECT COST FORM',
        ' - COUNTY CODE ',COUNTY_NUMBER)(COL(32),A,A,A(2));
17      1      1      PUT SKIP(3);
18      1      1      ERROR = 0;
19      1      1      LAST_CNTY_NO = COUNTY_NUMBER;
20      1      1      END;

21      1      IF COUNTY_NUMBER < '001' | COUNTY_NUMBER > '099'
22      1      THEN COUNTY_CODE = 0;
23      1      ELSE COUNTY_CODE = 1;
24      1      COLUMN = VERIFY(INDIRECT_COST,' 0123456789') + 3;

25      1      IF COLUMN = 3
26      1      THEN IF COUNTY_CODE = 0
27      1      THEN DO:
28      1      1      PUT SKIP EDIT('THE FOLLOWING RECORD HAS AN ILLEGAL ',
        'COUNTY CODE ') (X(5),A,A);
29      1      1      PUT SKIP EDIT(INDIRECT_COST_REC'D) (X(10),A,A);
30      1      1      ERROR = 1;
31      1      1      END;
32      1      ELSE DO:
33      1      1      WRITE FILE(ICOST) FROM (INDIRECT_COST_RECORD);
34      1      1      END;

35      1      ELSE IF COUNTY_CODE = 0
36      1      THEN DO:
37      1      1      PUT SKIP EDIT('THE FOLLOWING RECORD HAS AN ILLEGAL COUNTY ',
        'CODE AND AN ILLEGAL CHARACTER AT POSITION ',
        COLUMN) (X(5),A,A,F(2));
38      1      1      PUT SKIP EDIT (INDIRECT_COST_RECORD) (X(10),A,A);
39      1      1      ERROR = 1;
40      1      1      END;
41      1      ELSE DO:
42      1      1      PUT SKIP EDIT('THE FOLLOWING RECORD HAS AN ILLEGAL ',
        'CHARACTER AT POSITION ',COLUMN)(X(5),A,A,F(2));
43      1      1      PUT SKIP EDIT(INDIRECT_COST_RECORD) (X(10),A,A);
44      1      1      ERROR = 1;
45      1      1      END;

46      1      GO TO NEXT;

47      1      DCNE:
48      1      IF ERROR=0 THEN PUT SKIP EDIT ('NO ERRORS')(X(5),A);
49      1      END EICR;
    
```

Listing of Program STCHNGE

SYMT LEVEL NEST

1

SCR: PROC OPTICNS(MAIN);

```

/*      PROGRAM STCHNGE                                     */
/*      THIS PROGRAM EDITS EQUIPMENT STATUS CHANGE CARDS.  */
/*      INVALID INPUT CARDS ARE LISTED ON PRINTER OUTPUT.  */
/*      THE THREE INPUT CARDS PER PIECE OF EQUIPMENT ARE COMBINED */
/*      INTO ONE LOGICAL RECORD AND WRITTEN TO AN OUTPUT FILE. */
/*      THIS PROGRAM IS ONE OF EIGHT COMPUTER PROGRAMS WHICH FORM */
/*      THE "COMPUTER BASED INFORMATION SYSTEM FOR COUNTY EQUIPMENT */
/*      CCST RECORDS".                                       */
/*      WRITTEN BY                                           */
/*      SYSTEMS DIVISION                                     */
/*      COLLEGE OF ENGINEERING                               */
/*      THE UNIVERSITY OF IOWA                               */
/*      IOWA CITY, IOWA                                     */
/*      JULY 1975                                           */
/*      DECLARE THE STRUCTURES TO HOLD THE INPUT CARDS     */

```

2 1

```

DECLARE TSC FILE RECORD SEQUENTIAL,
1 CARD1,

```

```

2 COUNTY_NUMBER          CHAR(3),
2 EQUIPMENT_NUMBER      CHAR(8),
2 CARD_TYPE1            CHAR(1),
2 DISTRICT              CHAR(2),
2 CLASS_CODE            CHAR(2),
2 CLASS_DESCRIPTION     CHAR(20),
2 MANUFACTURER_CODE    CHAR(3),
2 YEAR_EQ_MANUFACTURED  CHAR(2),
2 MAKE_MODEL_SERIAL_NO CHAR(28),
2 WHEELBASE             CHAR(3),
2 DATE_PURCHASED,
3 MONTH                 CHAR(2),
3 DAY                   CHAR(2),
3 YEAR                  CHAR(2),
2 FILLER                CHAR(2);

```

3 1

```

DECLARE
1 CARD2,

```

```

2 CTY_NO_EQ_NO          CHAR(11),
2 CARD_TYPE2            CHAR(1),
2 TYPE_OF_ENGINE        CHAR(1),
2 ENGINE_MAKE_MODEL_DSC CHAR(14),
2 ENG_MAF_CODE          CHAR(2),
2 RATED_HCRSEPOWER     CHAR(3),
2 NO_CYLINDERS          CHAR(2),
2 TRANS_TYPE            CHAR(1),
2 COMPANY_PUR_FROM     CHAR(12),
2 COST_VALUE            CHAR(23),
2 CHNG_STATUS_CODE     CHAR(1),
2 DATE_CHNG_STATUS,
3 MONTH                 CHAR(2),
3 DAY                   CHAR(2);

```

STMT LEVEL NEST

		3 YEAR	CHAR(2),
		2 FILLER	CHAR(3);
4	1	DECLARE	
		1 CARD3,	
		2 CTY_NO_EQ_NO2	CHAR(11),
		2 CARD_TYPE3	CHAR(1),
		2 LIFE_COST	CHAR(57),
		2 FILLER	CHAR(11);

STMT LEVEL NEST

```

/*      DECLARE THE STRUCTURE TO HOLD THE OUTPUT RECORD      */
5      1      DECLARE SCHNGE FILE RECORD SEQUENTIAL,
          1 STATUS_CHANGE_RECORD,
            2 COUNTY_NUMBER          CHAR(3),
            2 EQUIPMENT_NUMBER       CHAR(8),
            2 DISTRICT               CHAR(2),
            2 CLASS_CODE             CHAR(2),
            2 CLASS_DESCRIPTION      CHAR(20),
            2 MANUFACTURER_CODE      CHAR(3),
            2 YEAR_EQ_MANUFACTURED   CHAR(2),
            2 MAKE_MODEL_SERIAL_NO   CHAR(28),
            2 WHEELBASE              CHAR(3),
            2 DATE_PURCHASED,
              3 MONTH                CHAR(2),
              3 DAY                  CHAR(2),
              3 YEAR                 CHAR(2),
            2 TYPE_OF_ENGINE         CHAR(1),
            2 ENGINE_MAKE_MODEL_DSC  CHAR(14),
            2 ENG_MAF_CODE           CHAR(2),
            2 RATED_HORSEPOWER       CHAR(3),
            2 NO_CYLINDERS           CHAR(2),
            2 TRANS_TYPE             CHAR(1),
            2 COMPANY_PUR_FROM       CHAR(12),
            2 COST_VALUE             CHAR(23),
            2 CHNG_STATUS_CODE       CHAR(1),
            2 DATE_CHNG_STATUS,
              3 MONTH                CHAR(2),
              3 DAY                  CHAR(2),
              3 YEAR                 CHAR(2),
            2 LIFE_COST              CHAR(5);

6      1      DECLARE DISTRICT(99) CHAR(2) INIT('04','04','02','05','04','06','02',
          '01','02','06','03','02','03','03','04','06','02','03','02','05','03',
          '02','06','03','04','05','05','06','05','03','06','02','02','02','02',
          '04','01','01','04','01','02','01','03','05','02','02','03','06','06',
          '01','05','06','06','05','02','05','06','05','05','03','04','05','05',
          '01','04','02','03','05','04','05','03','03','04','03','03','03','01',
          '04','01','04','03','06','04','03','01','01','04','04','05','05',
          '05','05','05','01','02','02','03','02','02');

7      1      DECLARE NEW_COUNTY CHAR(3) INIT(' ');
8      1      DECLARE CCNTY FIXED BIN(15,0) INIT(2);
9      1      DECLARE FLAG FIXED BIN(15,0) INIT(0);
10     1      DECLARE BLANKS CHAR(80) INIT((80) ' ');
11     1      DECLARE 1 TODAYS_DATE,
          2 C_YEAR          CHAR(2),
          2 C_MONTH_DAY     CHAR(4);

12     1      TODAYS_DATE = DATE;
13     1      DECLARE DESCRIPTIONS CHAR(20);
14     1      DECLARE CL_DESC(0:54) CHAR(20);
15     1      DECLARE CODES FIXED DEC(2);

16     1      OPEN FILE(TSC) INPUT;
    
```


STMT LEVEL NEST

17	1	OPEN FILE(SCHANGE) OUTPUT;
18	1	ON ENDFILE(TSC) GO TO DONE;
20	1	ON ENDFILE(SYSIN) GO TO NEXT;

STMT LEVEL NEST

```

/*      READ IN CLASS DESCRIPTIONS FROM CARDS      */
22      1      FIRST: GET EDIT(CODES,DESCRIPTIONS)(COL(0),F(2),A(20));
23      1      CL_DESC(CODES) = DESCRIPTIONS;
24      1      GO TO FIRST;

/*      READ AND EDIT THE FIRST CARD OF AN EQUIPMENT STATUS      */
/*      CHANGE RECORD.      */

25      1      NEXT: READ FILE(ISC) INTO(CARD1);

26      1      IF NEW_COUNTY ^= CARD1.COUNTY_NUMBER
27      1      THEN DO:
28      1      1      IF COUNT = 0 THEN PUT SKIP EDIT('ALL DATA SUBMITTED BY COUNTY ',
                NEW_COUNTY, ' HAS BEEN EDITED AND NO ERRORS WERE FOUND.')
                (CCL(30),A,A,A);
30      1      1      ELSE COUNT = 0;
31      1      1      PUT PAGE EDIT('ERROR LISTING OF STATUS CHANGE RECORDS FOR COUNTY ',
                CARD1.COUNTY_NUMBER)(CCL(35),A,A);
32      1      1      PUT SKIP(3);
33      1      1      NEW_COUNTY = CARD1.COUNTY_NUMBER;
34      1      1      END;

35      1      IF ((CARD1.COUNTY_NUMBER < '000') | (CARD1.COUNTY_NUMBER > '099'))
36      1      THEN DO:
37      1      1      FLAG = 1;
38      1      1      PUT SKIP EDIT('THE COUNTY CODE IS INVALID')(X(5),A);
39      1      1      END;

40      1      IF CARD_TYPE1 ^= '1'
41      1      THEN DO:
42      1      1      FLAG = 1;
43      1      1      PUT SKIP EDIT('CARD TYPE NOT 1 ON FIRST CARD')(X(5),A);
44      1      1      END;

45      1      IF ((CARD1.CLASS_CODE < '00') & (CARD1.CLASS_CODE ^= ' ') |
                (CARD1.CLASS_CODE > '54'))
46      1      THEN DO:
47      1      1      FLAG = 1;
48      1      1      PUT SKIP EDIT('CLASS CODE IS INVALID')(X(5),A);
49      1      1      END;

50      1      IF (CARD1.YEAR_EQ_MANUFACTURED > C_YEAR) |
                ((CARD1.YEAR_EQ_MANUFACTURED < '00') &
                (CARD1.YEAR_EQ_MANUFACTURED ^= ' '))
51      1      THEN DO:
52      1      1      FLAG = 1;
53      1      1      PUT SKIP EDIT('YEAR EQUIPMENT MANUFACTURED IS INVALID')(X(5),A);
54      1      1      END;

55      1      IF ((CARD1.MANUFACTURER_CODE < '000') & (CARD1.MANUFACTURER_CODE ^=
                ' ') | (CARD1.MANUFACTURER_CODE > '135'))
56      1      THEN DO:
57      1      1      FLAG = 1;
58      1      1      PUT SKIP EDIT('THE MANUFACTURER CODE NUMBER IS INVALID')(X(5),A);

```

STMT LEVEL NEST

```

55      1      1      END;
60      1          IF VERIFY(CARD1.WHEELBASE,'1234567890') ~= 0
61      1          THEN DO;
62      1      1      FLAG = 1;
63      1      1      PUT SKIP EDIT('THE WHEELBASE VALUE IS INVALID.')(X(5),A);
64      1      1      END;

65      1          IF ((CARD1.DATE_PURCHASED.MONTH < '01') & (CARD1.DATE_PURCHASED.MONTH
66      1          ~= ' ')) | (CARD1.DATE_PURCHASED.MONTH > '12')
67      1      1      THEN DO;
68      1      1      FLAG = 1;
69      1      1      PUT SKIP EDIT('THE MONTH OF THE DATE PURCHASED IS INVALID.')(X(5),A);
70      1      1      END;

71      1          IF ((CARD1.DATE_PURCHASED.DAY < '01') & (CARD1.DATE_PURCHASED.DAY ~=
72      1          ' ')) | (CARD1.DATE_PURCHASED.DAY > '31')
73      1      1      THEN DO;
74      1      1      FLAG = 1;
75      1      1      PUT SKIP EDIT('THE DAY OF THE DATE PURCHASED IS INVALID.')(X(5),A);
76      1      1      END;

77      1          IF (CARD1.DATE_PURCHASED.YEAR > C_YEAR) |
78      1          ((CARD1.DATE_PURCHASED.YEAR < '00') & (CARD1.DATE_PURCHASED.YEAR
79      1          ~= ' '))
80      1          THEN DO;
81      1      1      FLAG = 1;
82      1      1      PUT SKIP EDIT('THE YEAR OF THE DATE PURCHASED IS INVALID.')(X(5),A);
83      1      1      END;

84      1          IF CARD_TYPE1 ~= '1'
85      1          THEN DO;
86      1      1      CARD2 = BLANKS;
87      1      1      CAPD3 = BLANKS;
88      1      1      GO TO TEST;
89      1      1      END;

```

STMT LEVEL NEST

```

/*      READ AND EDIT THE SECOND CARD OF AN EQUIPMENT STATUS      */
/*      CHANGE RECORD                                              */
86      1      READ FILE(TSC) INTO(CARD2);
87      1      IF CARD_TYPE2 ^= '2'
88      1      THEN DO:
89      1      1      FLAG = 1;
90      1      1      PUT SKIP EDIT('THE CARD TYPE FOR THE SECOND CARD IS NOT TYPE 2.')(X(5),A);
91      1      1      END;

92      1      IF VERIFY(CARD2.TYPE_OF_ENGINE,' GD') ^= 0
93      1      THEN DO:
94      1      1      FLAG = 1;
95      1      1      PUT SKIP EDIT('THE ENGINE FUEL CCDE IS NOT A 'G' OR 'D'.')(X(5),A);
96      1      1      END;

97      1      IF ((CARD2.ENG_MAF_CODE < '00') & (CARD2.ENG_MAF_CODE ^= ' ') |
98      1      (CARD2.ENG_MAF_CODE > '34'))
99      1      1      THEN DO:
100     1      1      FLAG = 1;
101     1      1      PUT SKIP EDIT('THE ENGINE MANUFACTURER CODE IS INVALID.')(X(5),A);
102     1      1      END;

103     1      IF ((CARD2.RATED_HCRSEPOWER < '000') &
104     1      (CARD2.RATED_HORSEPOWER ^= ' ') |
105     1      (CARD2.RATED_HORSEPOWER > '999'))
106     1      1      THEN DO:
107     1      1      FLAG = 1;
108     1      1      PUT SKIP EDIT('THE RATED HORSEPOWER IS INVALID.')(X(5),A);
109     1      1      END;

110     1      IF (CARD2.NO_CYLINDERS ^= '02') & (CARD2.NO_CYLINDERS ^= '04') &
111     1      (CARD2.NO_CYLINDERS ^= '06') & (CARD2.NO_CYLINDERS ^= '08') &
112     1      (CARD2.NO_CYLINDERS ^= '12') & (CARD2.NO_CYLINDERS ^= '01') &
113     1      (CARD2.NO_CYLINDERS ^= '00') & (CARD2.NO_CYLINDERS ^= ' ')
114     1      1      THEN DO:
115     1      1      FLAG = 1;
116     1      1      PUT SKIP EDIT('THE NUMBER OF CYLINDERS IS INVALID.')(X(5),A);
117     1      1      END;

118     1      IF VERIFY(CARD2.TRANS_TYPE,' AS') ^= 0
119     1      THEN DO:
120     1      1      FLAG = 1;
121     1      1      PUT SKIP EDIT('THE TRANSMISSION CODE IS INVALID.')(X(5),A);
122     1      1      END;

123     1      IF VERIFY(CARD2.COST_VALUE,' 1234567890') ^= 0
124     1      THEN DO:
125     1      1      FLAG = 1;
126     1      1      PUT SKIP EDIT('THE ORIGINAL PURCHASE PRICE, SALVAGE VALUE, OR ',
127     1      'BOOK VALUE CONTAINS INVALID CHARACTER(S).')(X(5),A,A);
128     1      1      END;

```

STMT LEVEL NEST

```

122 1      IF VERIFY(CARD2.CHNG_STATUS_CCDE, 'JST') = 0
123 1      THEN DO;
124 1 1      FLAG = 1;
125 1 1      PUT SKIP EDIT('THE SOLD, JUNKED, OR TRADED CCDE FIELD CONTAINS AN ',
126 1 1      'INVALID CHARACTER.')(X(5),A,A);
127 1      END;
127 1      IF ((CARD2.DATE_CHNG_STATUS.MONTH < '01') &
128 1      (CARD2.DATE_CHNG_STATUS.MONTH = ' ')) |
129 1      (CARD2.DATE_CHNG_STATUS.MONTH > '12')
130 1      THEN DO;
130 1 1      FLAG = 1;
131 1 1      PUT SKIP EDIT('THE MONTH OF DATE SOLD, JUNKED, OR TRADED ',
132 1 1      'IS INVALID.')(X(5),A,A);
133 1      END;
132 1      IF ((CARD2.DATE_CHNG_STATUS.DAY < '01') &
134 1      (CARD2.DATE_CHNG_STATUS.DAY = ' ')) |
135 1      (CARD2.DATE_CHNG_STATUS.DAY > '31')
136 1      THEN DO;
136 1 1      FLAG = 1;
137 1 1      PUT SKIP EDIT('THE DAY OF DATE SOLD, JUNKED, OR TRADED IS INVALID.')(
138 1 1      X(5),A);
139 1      END;
137 1      IF (CARD2.DATE_CHNG_STATUS.YEAR > C_YEAR) |
140 1      ((CARD2.DATE_CHNG_STATUS.YEAR < '00') &
141 1      (CARD2.DATE_CHNG_STATUS.YEAR = ' '))
142 1      THEN DO;
143 1 1      FLAG = 1;
144 1 1      PUT SKIP EDIT('THE YEAR OF THE DATE SOLD, JUNKED, OR TRADED IS ',
145 1 1      'INVALID.')(X(5),A,A);
146 1 1      END;
142 1      IF CARD_TYPE2 = '2'
143 1      THEN DO;
144 1 1      CARD3 = BLANKS;
145 1 1      GO TO TEST;
146 1 1      END;

```

STMT LEVEL NEST

```

/*      READ AND EDIT THE THIRD CARD OF AN EQUIPMENT STATUS      */
/*      CHANGE RECCRD                                             */
147   1      READ FILE(TSC) INTO(CARD3);
148   1      IF CARD_TYPE3 ^= '3'
149   1      THEN DO;
150   1      1      FLAG = 1;
151   1      1      PUT SKIP EDIT('THE CARD TYPE FOR THE THIRD CARD IS INVALID. ')
152   1      1      (X(5),A);
152   1      END;
153   1      IF VERIFY(CARD3.LIFE_COST,'1234567890') ^= 0
154   1      THEN DO;
155   1      1      FLAG = 1;
156   1      1      PUT SKIP EDIT('THE LIFE INFORMATION CONTAINS AT LEAST ONE INVALID ',
157   1      1      'CHARACTER.')(X(5),A,A);
157   1      1      END;
158   1      TEST: IF FLAG = 0
159   1      THEN DO: /* WRITE A RECORD WITH NO ERRORS TO OUTPUT FILE. */
160   1      1      CARD1.DISTRICT = DISTRICT(CARD1.COUNTY_NUMBER);
161   1      1      IF CARD1.CLASS_CODE ^= ' ' THEN
162   1      1      CARD1.CLASS_DESCRIPTION = CL_DESC(CARD1.CLASS_CODE);
163   1      1      STATUS_CHANGE_RECCRD = CARD1, BY NAME;
164   1      1      STATUS_CHANGE_RECCRD = CARD2, BY NAME;
165   1      1      STATUS_CHANGE_RECCRD = CARD3, BY NAME;
166   1      1      WRITE FILE(SCHANGE) FROM(STATUS_CHANGE_RECCRD);
167   1      1      GO TO NEXT;
168   1      1      END;
169   1      ELSE DO: /* PRINT A RECORD CONTAINING ERRORS ON THE PRINTER. */
170   1      1      PUT SKIP(2) EDIT(CARD1) (X(1),(13)A);
171   1      1      PUT SKIP EDIT(CARD2)(X(1),(14)A);
172   1      1      PUT SKIP EDIT(CARD3)(X(1),(13)A);
173   1      1      PUT SKIP(2);
174   1      1      COUNT = 1;
175   1      1      FLAG = 0;
176   1      1      GO TO NEXT;
177   1      1      END;
178   1      DONE: IF CCUNT = 0
179   1      THEN PUT SKIP EDIT('ALL DATA SUBMITTED BY CCUNTY ',
180   1      'NEW_CCUNTY,' HAS BEEN EDITED AND NO ERRORS WERE FOUND. ')
180   1      (COL(30),A,A,A);
180   1      END SCR;

```

Listing of Program UPDATES

STMT LEVEL NEST
1

UPDATEM: PROC OPTICNS(MAIN);

```
/*      PROGRAM UPDATEM      */
/*      */
/*      THIS PROGRAM INPUTS NEW STATUS CHANGE RECORDS AND      */
/*      UPDATES EXISTING RECORDS WITH NEW DATA OR ADDS NEW      */
/*      RECORDS TO THE EQUIPMENT MASTER FILE. TOTAL COUNTY      */
/*      EQUIPMENT BOOK VALUES ARE CALCULATED. THE EQUIPMENT      */
/*      INVENTORY LIST FOR THE PREVIOUS YEAR IS OUTPUT.          */
/*      */
/*      THIS PROGRAM IS ONE OF EIGHT COMPUTER PROGRAMS          */
/*      WHICH FORM THE "COMPUTER BASED INFORMATION SYSTEM        */
/*      FOR COUNTY EQUIPMENT COST RECORDS."                      */
/*      */
/*      WRITTEN BY                                               */
/*      */
/*      SYSTEMS DIVISION                                         */
/*      COLLEGE OF ENGINEERING                                    */
/*      THE UNIVERSITY OF IOWA                                   */
/*      IOWA CITY, IOWA                                         */
/*      JULY, 1975                                              */
/*      */
```


STMT LEVEL NEST

```

2      1      DCL STCHNC FILE RECORD SEQUENTIAL;
3      1      DCL 1 SCR,
                2 COUNTY_NUM          CHAR(3) INIT('999'),
                2 EC_NUM              CHAR(8),
                2 DISTRICT            CHAR(2),
                2 CLASS_CODE          CHAR(2),
                2 CLASS_DESC          CHAR(20),
                2 MFR_CODE            CHAR(3),
                2 YEAR_MFD            CHAR(2),
                2 MAKE_MODEL          CHAR(14),
                2 SERIAL_NUM          CHAR(14),
                2 WHEELBASE           CHAR(3),
                2 DATE_PURCHASED,
                    3 PCHN            CHAR(2),
                    3 PDAY            CHAR(2),
                    3 PYR             CHAR(2),
                2 ENGINE_TYPE         CHAR(1),
                2 ENGINE_DESC         CHAR(14),
                2 ENGINE_MFR_CODE     CHAR(2),
                2 RATED_HP            CHAR(3),
                2 NUM_CYLS            CHAR(2),
                2 XMISSION_TYPE       CHAR(1),
                2 CD_PURCHASED_FRM    CHAR(12),
                2 ORIG_COST           PIC'999999V99',
                2 SALVAGE_VALUE       PIC'999999V99',
                2 BOOK_VALUE          PIC'999999V99',
                2 DISPOSITION         CHAR(1),
                2 DATE_DISPOSED,
                    3 DMCN            CHAR(2),
                    3 DDAY            CHAR(2),
                    3 DYR             CHAR(2),
                2 MILES_HRS_LIFE      CHAR(6),
                2 FUEL_COST_LIFE      CHAR(7),
                2 LUBRICANTS_LIFE     CHAR(6),
                2 TIRES_TUBES_LIFE    CHAR(6),
                2 EXPEN_PARTS_LIFE    CHAR(6),
                2 ANTIFREEZE_LIFE     CHAR(5),
                2 PARTS_COST_LIFE     CHAR(7),
                2 LABOR_COST_LIFE     CHAR(7),
                2 IND_COST_LIFE       CHAR(7);
    
```

STAT LEVEL NEST

```

4      1      DCL  EQMAST FILE RECORD SEQUENTIAL,
          1 MSTR,
          2 COUNTY_NUM          CHAR(3) INIT('998'),
          2 DISTRICT            CHAR(2),
          2 EC_NUM              CHAR(8),
          2 CLASS_CODE          CHAR(2),
          2 YEAR_MFD            CHAR(2),
          2 MFR_CODE            CHAR(3),
          2 ENGINE_MFR_CODE     CHAR(2),
          2 NUM_CYLS            CHAR(2),
          2 ENGINE_TYPE         CHAR(1),
          2 XMISSION_TYPE       CHAR(1),
          2 ORIG_COST           PIC'9999999V99',
          2 SALVAGE_VALUE       PIC'9999999V99',
          2 BOOK_VALUE          PIC'9999999V99',
          2 CURRENT_COSTS       CHAR(6),
          2 MILES_HRS_LIFE      CHAR(6),
          2 FUEL_COST_LIFE      CHAR(7),
          2 LUBRICANTS_LIFE     CHAR(6),
          2 TIRES_TUBES_LIFE    CHAR(6),
          2 EXPEN_PARTS_LIFE    CHAR(6),
          2 ANTI_FREEZE_LIFE    CHAR(5),
          2 PARTS_COST_LIFE     CHAR(7),
          2 LABOR_COST_LIFE     CHAR(7),
          2 IND_COST_LIFE       CHAR(7),
          2 CLASS_DESC          CHAR(20),
          2 MAKE_MODEL          CHAR(14),
          2 SERIAL_NUM          CHAR(14),
          2 ENGINE_DESC         CHAR(14),
          2 CC_PURCHASED_FRM    CHAR(12),
          2 DATE_PURCHASED,
            3 PMCN              CHAR(2),
            3 PDAY              CHAR(2),
            3 PYR               CHAR(2),
          2 DATE_DISPOSED,
            3 DMCN              CHAR(2),
            3 DDAY              CHAR(2),
            3 CYR               CHAR(2),
          2 DISPOSITION         CHAR(1),
          2 WHEELBASE           CHAR(3),
          2 REPAIRS             CHAR(13),
          2 RATED_HP            CHAR(3),
          2 BLANKS              CHAR(24);
    
```

STMT LEVEL NEST

```

5      1      DCL  1 CNTY_BCK_VALU,
           2 LAST_CNTY_NUM      CHAR(3) INIT('000'),
           2 TOTAL_BCK_VALU    PIC'99999999V99' INIT((10)'0');
6      1      DCL DISP CHAR(6) VARYING;
7      1      DCL LETTERS FIXED DEC(5);
8      1      DCL THIS_CNTY_NUM      CHAR(3);

9      1      DCL SCDCNE      BIT(1)      INIT('0'B);
10     1      DCL EMDJNE      BIT(1)      INIT('0'B);

11     1      DCL  1 TODAYS_DATE,
           2 C_YEAR          CHAR(2),
           2 C_MON_DAY      CHAR(4);
12     1      TODAYS_DATE = DATE;
13     1      DCL LAST_YEAR      FIXED DEC(4);
14     1      LAST_YEAR = 1899 + C_YEAR;

15     1      DCL  1 SAVMSTR LIKE MSTR;
16     1      DCL NECMASD FILE RECORD SEQUENTIAL;
17     1      DCL BKVALU FILE RECORD SEQUENTIAL;

18     1      EQ_INV_LIST:  FORMAT(COL(3),A,COL(13),A,CCL(28),A,COL(43),A,
           CCL(57),(2)A,
           COL(63),A,COL(69),A,COL(74),A,COL(78),A,CCL(82),A,COL(87),A,
           CCL(103),(5)A,COL(112),P'ZZZZZV.99',COL(122),P'ZZZZZV.99');

19     1      DCL  CNTY_NAMES(99) CHAR(13) VARYING
           INIT('ADAIR','ADAMS','ALLAMAKEE','APPANOOSE','AUDUBON','BENTON',
           'BLACK HAWK','BCCNE','BREMEN','BUCHANAN','BUENA VISTA','EUTLER',
           'CAIHOUN','CARPELL','CASS','CEDAR','CERRC GORDO','CHERCKEE',
           'CHICKASAW','CLARKE','CLAY','CLAYTON','CLINTON','CRAWFCRC','DALLAS',
           'DAVIS','DECATUR','DELAWARE','DES MOINES','DICKINSON','DLRUQUE',
           'EMMET','FAYETTE','FLOYD','FRANKLIN','FREMONT','GREENE','GRUNDY',
           'GUTHRIE','HAMILTON','HANCOCK','HARDIN','HARPISON','HENRY',
           'HOWARD','HUMBLOD','IDA','TCHWA','JACKSON','JASPER','JEFFERSON',
           'JOHNSON','JONES','KEOKUK','KOSOUTH','LEE','LINN','LOUISA','LUCAS',
           'LYON','MADISON','MAHASKA','MARION','MARSHALL','MILLS','MITCHELL',
           'MONONA','MONRCE','MONTGOMERY','MUSCATINE','C'BRIEN','OSCEOLA',
           'PAGE','PALO ALTO','PLYMOUTH','POCAHONTAS','PCLK','POTTAWATTAMIE',
           'POWESHIEK','RINGOLD','SAC','SCOTT','SHELBY','SIOUX','STORY',
           'TAMA','TAYLOR','UNION','VAN BUREN','WAPELLO','WARREN',
           'WASHINGTON','WAYNE','WEBSTER','WINNEBAGO','WINNESHIEK','WOODBURY',
           'WORTH','WRIGHT');

```

STMT LEVEL NEST

```
20      1      ON  ENDFILE(EQMAST) BEGIN;
22      2      IF   SCDCNE = '1'B
23      2      THEN GO TO DCNE;
24      2      ELSE DO;
25      2      1      EMDONE = '1'B;
26      2      1      MSTR.COUNTY_NUM = '999';
27      2      1      MSTR.EQ_NUM = '99999999';
28      2      1      END;
29      2      END;

30      1      ON  ENDFILE(STCHNG)  BEGIN;
32      2      IF   EMDONE = '1'B
33      2      THEN GO TO DCNE;
34      2      ELSE DO;
35      2      1      SCDCNE = '1'B;
36      2      1      SCR.COUNTY_NUM = '999';
37      2      1      SCR.EQ_NUM = '99999999';
38      2      1      END;
39      2      END;

40      1      OPEN FILE(EQMAST) INPUT;
41      1      OPEN FILE(STCHNG) INPUT;
42      1      OPEN FILE(NEQMAST) OUTPUT;
43      1      OPEN FILE(BKVALU) OUTPUT;
44      1      OPEN FILE(SYSPRINT) LINESIZE(132) PAGESIZE(60);
45      1      READ FILE(EQMAST) INTO (MSTR);
46      1      READ FILE(STCHNG) INTO (SCR);
```

STMT LEVEL NEST

```

47      1      NEXT:
48      1      IF SCR.COUNTY_NUM > MSTR.COUNTY_NUM | SCR.EQ_NUM > MSTR.EQ_NUM
49      1      1      THEN DO: /* COPY MASTER FILE RECORD */
50      1      1      IF LAST_CNTY_NUM = MSTR.COUNTY_NUM
51      1      1      THEN DO:
52      1      2      THIS_CNTY_NUM = MSTR.COUNTY_NUM;
53      1      2      CALL HEADING;
54      1      2      IF LAST_CNTY_NUM = '000'
55      1      2      THEN WRITE FILE(BKVALU) FROM(CNTY_BOOK_VALU);
56      1      2      ELSE:
57      1      2      TOTAL_BOOK_VALU = (10)'0';
58      1      2      END;
59      1      1      ELSE:
60      1      1      TOTAL_BOOK_VALU = TOTAL_BOOK_VALU + MSTR.BOOK_VALUE;
61      1      1      WRITE FILE(EQMAST) FROM (MSTR);
62      1      1      PUT SKIP EDIT(MSTR.EQ_NUM,MSTR.MAKE_MODEL,MSTR.SERIAL_NUM,
63      1      1      MSTR.CO_PURCHASED_FFCM,'19',MSTR.YEAR_MFD,MSTR.WHEELBASE,
64      1      1      MSTR.XMISSION_TYPE,MSTR.NUM_CYLS,MSTR.ENGINE_TYPE,
65      1      1      MSTR.RATED_HP,MSTR.ENGINE_DESC,MSTR.DATE_PURCHASEC.PMON,'-',
66      1      1      MSTR.DATE_PURCHASEC.PDAY,'-',MSTR.CATE_PURCHASEC.PYR,
67      1      1      MSTR.ORIG_COST,MSTR.BOOK_VALUE)(R(EC_INV_LIST));
68      1      1      LAST_CNTY_NUM = MSTR.COUNTY_NUM;
69      1      1      READ FILE(EQMAST) INTO (MSTR);
70      1      1      GO TO NEXT;
71      1      1      END;

```

STMT LEVEL NEST

```

66      1      ELSE IF SCR.COUNTY_NUM = MSTR.COUNTY_NUM & SCR.EQ_NUM = MSTR.EQ_NUM
67      1      THEN DO: /* UPDATE EXISTING RECORD ON MASTER FILE */
68      1      1      IF LAST_CNTY_NUM = SCR.COUNTY_NUM
69      1      1      THEN DO:
70      1      2      THIS_CNTY_NUM = SCR.COUNTY_NUM;
71      1      2      CALL HEADING;
72      1      2      IF LAST_CNTY_NUM = '000'
73      1      2      THEN WRITE FILE(BKVALU) FROM (CNTY_BOOK_VALU);
74      1      2      WRITE FILE(BKVALU) FROM (CNTY_BOOK_VALU);
75      1      2      TOTAL_BOOK_VALU = (10)'0';
76      1      2      END;
77      1      1      ELSE:

78      1      1      IF SCR.CLASS_CODE >= '00'
79      1      1      THEN DO:
80      1      2      MSTR.CLASS_CODE = SCR.CLASS_CODE;
81      1      2      MSTR.CLASS_DESC = SCR.CLASS_DESC;
82      1      2      END;

83      1      1      IF SCR.MFR_CODE > '00'
84      1      1      THEN MSTR.MFR_CODE = SCR.MFR_CODE;

85      1      1      IF SCR.YEAR_MFD > '00'
86      1      1      THEN MSTR.YEAR_MFD = SCR.YEAR_MFD;

87      1      1      IF SCR.MAKE_MODEL > (14)' '
88      1      1      THEN MSTR.MAKE_MODEL = SCR.MAKE_MODEL;

89      1      1      IF SCR.SERIAL_NUM > (14)' '
90      1      1      THEN MSTR.SERIAL_NUM = SCR.SERIAL_NUM;

91      1      1      IF SCR.WHEELBASE > '000'
92      1      1      THEN MSTR.WHEELBASE = SCR.WHEELBASE;

93      1      1      IF SCR.DATE_PURCHASED.PMON > '00'
          1      1      | SCR.DATE_PURCHASED.PDAY > '00'
          1      1      | SCR.DATE_PURCHASED.PYR > '00'
94      1      1      THEN MSTR.DATE_PURCHASED = SCR.DATE_PURCHASED;

95      1      1      IF SCR.ENGINE_TYPE > ' '
96      1      1      THEN MSTR.ENGINE_TYPE = SCR.ENGINE_TYPE;

97      1      1      IF SCR.ENGINE_DESC > (14)' '
98      1      1      THEN MSTR.ENGINE_DESC = SCR.ENGINE_DESC;

99      1      1      IF SCR.ENGINE_MFR_CODE > '00'
100     1      1      THEN MSTR.ENGINE_MFR_CODE = SCR.ENGINE_MFR_CODE;

101     1      1      IF SCR.RATED_HP >= '000'
102     1      1      THEN MSTR.RATED_HP = SCR.RATED_HP;

103     1      1      IF SCR.NUM_CYLS >= '00'
104     1      1      THEN MSTR.NUM_CYLS = SCR.NUM_CYLS;

105     1      1      IF SCR.XMISSION_TYPE > ' '

```

STMT LEVEL NEST

```

106      1      1      THEN MSTR.XMISSION_TYPE = SCR.XMISSION_TYPE;
107      1      1      IF   SCR.CC_PURCHASED_FROM > (12)' '
108      1      1      THEN MSTR.CC_PURCHASED_FRCM = SCR.CC_PURCHASED_FRCM;
109      1      1      IF   SCR.ORIG_COST > 0
110      1      1      THEN MSTR.ORIG_COST = SCR.ORIG_COST;
111      1      1      IF   SCR.SALVAGE_VALUE > 0
112      1      1      THEN MSTR.SALVAGE_VALUE = SCR.SALVAGE_VALUE;
113      1      1      IF   SCR.BOOK_VALU > 0
114      1      1      THEN MSTR.BOOK_VALUE = SCR.BOOK_VALUE;
115      1      1      MSTR.DISPOSITION = SCR.DISPOSITION;
116      1      1      MSTR.DATE_DISPOSED = SCR.DATE_DISPOSED;
117      1      1      IF   SCR.MILES_HRS_LIFE > '00000'
118      1      1      THEN MSTR.MILES_HRS_LIFE = SCR.MILES_HRS_LIFE;
119      1      1      IF   SCR.FUEL_COST_LIFE > '000000'
120      1      1      THEN MSTR.FUEL_COST_LIFE = SCR.FUEL_COST_LIFE;
121      1      1      IF   SCR.LUBRICANTS_LIFE > '00000'
122      1      1      THEN MSTR.LUBRICANTS_LIFE = SCR.LUBRICANTS_LIFE;
123      1      1      IF   SCR.TIRES_TUBES_LIFE > '00000'
124      1      1      THEN MSTR.TIRES_TUBES_LIFE = SCR.TIRES_TUBES_LIFE;
125      1      1      IF   SCR.EXPEN_PARTS_LIFE > '00000'
126      1      1      THEN MSTR.EXPEN_PARTS_LIFE = SCR.EXPEN_PARTS_LIFE;
127      1      1      IF   SCR.ANTIFREEZE_LIFE > '00000'
128      1      1      THEN MSTR.ANTIFREEZE_LIFE = SCR.ANTIFREEZE_LIFE;
129      1      1      IF   SCR.PARTS_COST_LIFE > (7)'0'
130      1      1      THEN MSTR.PARTS_COST_LIFE = SCR.PARTS_COST_LIFE;
131      1      1      IF   SCR.LABOR_COST_LIFE > (7)'0'
132      1      1      THEN MSTR.LABOR_COST_LIFE = SCR.LABOR_COST_LIFE;
133      1      1      IF   SCR.IND_COST_LIFE > (7)'0'
134      1      1      THEN MSTR.IND_COST_LIFE = SCR.IND_COST_LIFE;
135      1      1      TOTAL_BCKK_VALU = TOTAL_BCKK_VALU + MSTR.BOOK_VALUE;
136      1      1      WRITE FILE(NEOMAST) FROM (MSTR);
137      1      1      PUT SKIP FCIT(MSTR.EQ_NUM,MSTR.MAKE_MODEL,MSTR.SERIAL_NUM,
MSTR.CO_PURCHASED_FRCM,'19',MSTR.YEAR_MFC,
MSTR.WHEELBASE,MSTR.XMISSION_TYPE,MSTR.NUM_CYLS,
MSTR.ENGINE_TYPE,MSTR.RATED_HP,
MSTR.ENGINE_DESC,MSTR.DATE_PURCHASED.PMON,'-',
MSTR.DATE_PURCHASED.PDAY,'-',MSTR.DATE_PURCHASED.PYR,
MSTR.ORIG_COST,MSTR.BOOK_VALUE)(REQ_INV_LIST);
138      1      1      IF   MSTR.DISPOSITION > ' '

```

STAT LEVEL NEST

```

139     1     1           THEN DO;
140     1     2           IF MSTR.DISPOSITION = 'J'
141     1     2           THEN DISP = 'JUNKED';
142     1     2           IF MSTR.DISPOSITION = 'T'
143     1     2           THEN DISP = 'TRADED';
144     1     2           IF MSTR.DISPOSITION = 'S'
145     1     2           THEN DISP = 'SOLD';
146     1     2           PUT SKIP EDIT ('*',MSTR.EQ_NUM,'AS ',DISP,' CN ',
MSTR.DATE_DISPOSED.DMON,'-',MSTR.DATE_DISPOSED.DDAY,
'-',MSTR.DATE_DISPOSED.DYR)(CCL(2),(10)A);
147     1     2           END;

148     1     1           LAST_CNTY_NUM = MSTR.COUNTY_NUM;
149     1     1           READ FILE(EQMAST) INTO (MSTR);
150     1     1           READ FILE(STCHNG) INTO (SCR);
151     1     1           GO TO NEXT;
152     1     1           END;

```


STMT LEVEL NEST

```

153 1 ELSE DO; /* ADD NEW RECORD TO MASTER FILE */
154 1 1 IF LAST_CNTY_NUM = SCR.CCNTY_NUM
155 1 1 THEN DO;
156 1 2 THIS_CNTY_NUM = SCR.CCNTY_NUM;
157 1 2 CALL HEADING;
158 1 2 IF LAST_CNTY_NUM = '000'
159 1 2 THEN WRITE FILE(BKVALU) FROM(CNTY_BOOK_VALU);
160 1 2 TOTAL_BOOK_VALU = (10)'0';
161 1 2 END;
162 1 1 ELSE;

163 1 1 SAVMSTR = MSTR, BY NAME;
164 1 1 MSTR = (300)'0';
165 1 1 MSTR = SCR, BY NAME;
166 1 1 TOTAL_BOOK_VALU = TOTAL_BOOK_VALU + MSTR.BOOK_VALUE;
167 1 1 WRITE FILE (INCOMAST) FROM (MSTR);

168 1 1 PUT SKIP EDIT(MSTR.EQ_NUM,MSTR.MAKE_MODEL,MSTR.SERIAL_NUM,
MSTR.CO_PURCHASED_FR CM,'19',MSTR.YEAR_MFD,
MSTR.WHEELBASE,MSTR.XMISSION_TYPE,MSTR.NUM_CYLS,
MSTR.ENGINE_TYPE,MSTR.RATED_HP,
MSTR.ENGINE_DESC,MSTR.DATE_PURCHASED.PMON,'-',
MSTR.DATE_PURCHASED.PDAY,'-',MSTR.DATE_PURCHASED.PYR,
MSTR.ORIG_COST,MSTR.BOOK_VALUE)(R(EQ_INV_LIST));

169 1 1 IF MSTR.DISPOSITION > ' '
170 1 1 THEN DO;
171 1 2 IF MSTR.DISPOSITION = 'J'
172 1 2 THEN DISP = 'JUNKED';
173 1 2 IF MSTR.DISPOSITION = 'T'
174 1 2 THEN DISP = 'TRADED';
175 1 2 IF MSTR.DISPOSITION = 'S'
176 1 2 THEN DISP = 'SOLD';
177 1 2 PUT SKIP EDIT ('*',MSTR.EQ_NUM,' WAS ',DISP,' CN ',
MSTR.DATE_DISPOSED.DMON,'-',MSTR.DATE_DISPOSED.DDAY,
'-',MSTR.DATE_DISPOSED.DYR)(CCL(2),(10)A);
178 1 2 END;

179 1 1 LAST_CNTY_NUM = MSTR.COUNTY_NUM;
180 1 1 MSTR = SAVMSTR, BY NAME;
181 1 1 READ FILE(STCHNG) INTO (SCR);
182 1 1 GO TO NEXT;
183 1 1 END;

```

STMT LEVEL NEST

```

184 1 HEADING: PRCC;
185 2 LETTERS = LENGTH(CNTY_NAMES(THIS_CNTY_NUM));
186 2 PUT PAGE EDIT(CNTY_NAMES(THIS_CNTY_NUM),' CCOUNTY')
(COL((125-LETTERS)/2),A,A);
187 2 PUT SKIP(2) EDIT (LAST_YEAR,' EQUIPMENT INVENTORY LIST')
(COL(51),F(4),A);
188 2 PUT SKIP(3) EDIT('EQUIPMENT','YEAR WPS TRAN','ENGINE MAKE',
'PURCHASE PURCHASE BOOK')(COL(3),A,CCL(57),A,COL(88),A,
CCL(103),A);
189 2 PUT SKIP EDIT('NUMBER MAKE & MODEL SERIAL NUMBER DEALER',
'MFRD INS TYPE CYL FUEL HP AND MODEL',
'DATE COST VALUE')(COL(4),A,COL(57),A,COL(105),A);
190 2 PUT SKIP(2);
191 2 RETURN;
192 2 END HEADING;

193 1 DCNE:
WRITE FILE(BKVALU) FROM (CNTY_BOOK_VALU);
194 1 END UPDATEM;
    
```

Listing of Program MAIN

Step 1

STMT LEVEL NEST
1

FMAIN: PROC OPTIONS(MAIN):

```

/*      PROGRAM MAIN                                */
/*      */                                           */
/*      THIS PROGRAM CALCULATES COST/PILE OR COST/HOUR FOR ALL */
/*      PIECES OF EQUIPMENT, BOTH FOR PAST YEAR AND LIFETIME */
/*      COSTS.                                          */
/*      */                                           */
/*      OUTPUT CONSISTS OF YEARLY AND LIFETIME DIRECT */
/*      OPERATING COSTS AND YEARLY AND LIFETIME TOTAL */
/*      OPERATING COSTS.                               */
/*      */                                           */
/*      THIS PROGRAM IS ONE OF EIGHT COMPUTER PROGRAMS */
/*      WHICH FORM THE "COMPUTER BASED INFORMATION SYSTEM */
/*      FOR COUNTY EQUIPMENT COST RECORDS."           */
/*      */                                           */
/*      WRITTEN BY                                     */
/*      */                                           */
/*      SYSTEMS DIVISION                             */
/*      COLLEGE OF ENGINEERING                       */
/*      THE UNIVERSITY OF IOWA                       */
/*      IOWA CITY, IOWA                             */
/*      JULY, 1975                                   */
/*      */                                           */

```

STMT LEVEL NEST

```

2 1 DCL SEQMAST FILE RECORD SEQUENTIAL,
1 MSTR,
2 COUNTY_NUM PIC'999',
2 DISTRICT CHAR(2),
2 EQ_NUM CHAR(8),
2 CLASS_CODE PIC'99',
2 MISC1 CHAR(11),
2 ORIG_COST PIC'999999V99',
2 SALVAGE_VALUE PIC'999999V99',
2 BOOK_VALUE PIC'999999V99',
2 DEPRECIATION PIC'999999V99',
2 MILES_HRS_YR PIC'999999',
2 FUEL_COST_YR PIC'999999V99',
2 LUBRICANTS_YR PIC'999999',
2 TIRES_TUBES_YR PIC'999999V99',
2 EXPEN_PARTS_YR PIC'999999V99',
2 ANTI_FREEZE_YR PIC'999999',
2 PARTS_COST_YR PIC'999999V99',
2 LABOR_COST_YR PIC'999999V99',
2 IND_COST_YR PIC'999999V99',
2 MILES_HRS_LIFE PIC'999999',
2 FUEL_COST_LIFE PIC'999999V99',
2 LUBRICANTS_LIFE PIC'999999V99',
2 TIRES_TUBES_LIFE PIC'999999V99',
2 EXPEN_PARTS_LIFE PIC'999999V99',
2 ANTI_FREEZE_LIFE PIC'999999',
2 PARTS_COST_LIFE PIC'999999V99',
2 LABOR_COST_LIFE PIC'999999V99',
2 IND_COST_LIFE PIC'999999V99',
2 CLASS_DESC CHAR(2),
2 MAKE_MODEL CHAR(14),
2 SERIAL_NUM CHAR(14),
2 MISC2 CHAR(42),
2 DOWN_TIME_YR PIC'9999',
2 DOWN_TIME_LIFE PIC'9999',
2 NUM_REPAIRS_YR PIC'99',
2 NUM_REPAIRS_LIFE PIC'999',
2 RATED_HP CHAR(3),
2 BLANKS CHAR(24);

```

STAT LEVEL NEST

```
3 1 DCL INDCST FILE RECORD SEQUENTIAL,  
1 IND,  
2 COUNTY_NUM PIC'999',  
2 SUP_SALARIES PIC'99999V99',  
2 CLER_SALARIES PIC'99999V99',  
2 UTILITIES PIC'99999V99',  
2 BLDG_COSTS PIC'99999V99',  
2 EQ_DEPR PIC'99999V99',  
2 EXP_TCCLS PIC'99999V99',  
2 SUPPLIES PIC'99999V99',  
2 MOVING_COSTS PIC'99999V99',  
2 EQ_INS PIC'99999V99';
```

SYMT LEVEL NEST

```
4 1 DCL CIRCOST FILE RECORD SEQUENTIAL,  
1 DIR,  
2 COUNTY_NUM PIC'999',  
2 EQ_NUM CHAR(8),  
2 DATE CHAR(4),  
2 FUEL_COST_YR PIC'9999V99',  
2 LUBRICANTS_YR PIC'999V99',  
2 ANTIFREEZE_YR PIC'99V99',  
2 TILES_TUBES_YR PIC'9999V99',  
2 EXPEN_PARTS_YR PIC'9999V99',  
2 PARTS_COST_YR PIC'9999V99',  
2 LABOR_COST_YR PIC'9999V99',  
2 MILES_HRS_YR PIC'99999',  
2 DOWN_TIME_YR PIC'9999',  
2 NUM_REPAIRS_YR PIC'99';
```

STMT LEVEL NEST

```

5 1 DCL BKVALU FILE RECORD SEQUENTIAL,
      1 BKVAL,
      2 CNTY_NAM PIC'999',
      2 VALUE PIC'99999999V99';

6 1 DCL CNTY_NAMES(99) CHAR(13) VARYING
      INIT('ADAIR','ADAMS','ALLAMAKEE','APPANOOSE','AUDURON','BENJON',
          'BLACK HAWK','BOONE','BREMER','BUCHANAN','BUENA VISTA','BUTLER',
          'CALHOUN','CARROLL','CASS','CEDAR','CEPRO GORDO','CHEPCKEE',
          'CHICKASAW','CLARKE','CLAY','CLAYTON','CLINTON','CRAWFORD','DALLAS',
          'DAVIS','DECATUR','DELAWARE','DES MOINES','DICKINSON','DUBUQUE',
          'EMMET','FAYETTE','FLOYD','FRANKLIN','FREMONT','GREENE','GRUNDY',
          'GUTHRIE','HAMILTON','HANCOCK','HARDIN','HARRISON','HENRY',
          'HOWARD','HUMBOLDT','IDA','IOWA','JACKSON','JASPER','JEFFERSON',
          'JOHNSON','JONES','KEOKUK','KOSSUTH','LEE','LINN','LOUISA','LUCAS',
          'LYON','MADISON','MAHASKA','MARION','MARSHALL','MILLS','MITCHELL',
          'MONONA','MONROE','MONTGOMERY','MUSCATINE','O'BRIEN','OSCEOLA',
          'PAGE','PALO ALTO','PLYMOUTH','POCAHONTAS','POLK','POTTAWATTAMIE',
          'POWESHIEK','RINGGOLD','SAC','SCOTT','SHELBY','SIOUX','STORY',
          'TAMA','TAYLOR','UNION','VAN BUREN','WAPELLO','WARREN',
          'WASHINGTON','WAYNE','WEBSTER','WINNEBAGO','WINNESHIEK','WOODBURY',
          'WORTH','WRIGHT');
    
```


STMT LEVEL NEST

```

7      1      DCL TEMP_BK_VAL      FIXED DEC(10,2);
8      1      DCL TOT_IND_CGS(99)    PIC'999999999V99' INIT((99)(10)'0');
9      1      DCL TOT_BK_VAL(99)     PIC'999999999V99' INIT((99)(10)'0');
10     1      DCL PATIC(99) PIC'999V999' INIT((99)(6)'0');
11     1      DCL INDEX      FIXED DEC(3);
12     1      DCL DEPRECIATION_YRS(0:54) FIXED DEC(6)
          INIT(10,5,5,5,5,6,7,7,6,7,6,8,5,6,8,9,10,10,10,10,12,13,13,14,
          13,14,15,10,10,10,13,13,8,8,10,10,10,10,10,10,8,10,10,10,12,
          13,13,8,11,11,11,12,12,12);
13     1      DCL IND_OK      BIT(1) INIT('1'B);
14     1      DCL BKVAL_OK   BIT(1) INIT('1'B);
15     1      DCL LAST_ERR_CNTY  FIXED DEC(3) INIT(0);
16     1      DCL EMDONE     BIT(1) INIT('0'B);
17     1      DCL DCCDNE     BIT(1) INIT('0'B);

18     1      ON ENDFILE(DIRCOST) BEGIN;
20     2      DCCDNE = '1'B;
21     2      IF EMDONE = '1'B
22     2      THEN GO TO DCNE;
23     2      ELSE DO;
24     2      1      NODIR:
          IF MSTR.COUNTY_NUM = LAST_ERR_CNTY
25     2      1      THEN DO;
26     2      2      PUT PAGE EDIT ('ERROR LISTING FOR ',
          CNTY_NAMES(MSTR.COUNTY_NUM),' COUNTY')(COL(30),A,A,A);
27     2      2      LAST_ERR_CNTY = MSTR.COUNTY_NUM;
28     2      2      END;
29     2      1      PUT SKIP(2) EDIT ('NO DIRECT COST CARD WAS FOUND FOR THE ',
          'FOLLOWING PIECE OF EQUIPMENT:')(A,A);
30     2      1      PUT SKIP EDIT ('EQUIPMENT NO.: ',MSTR.EQ_NUM,' MAKE & MODEL: ',
          MSTR.MAKE_MODEL,' SERIAL NO.: ',MSTR.SERIAL_NUM)
          (A,A,X(4),A,A,X(4),A,A);
31     2      1      WRITE FILE(TECMAST)FROM (MSTR);
32     2      1      READ FILE(SECMAST) INTO (MSTR);
33     2      1      GO TO NODIR;
34     2      1      END;
35     2      2      END;

36     1      ON ENDFILE(INDCOST)
37     1      IND_OK = '0'B;

38     1      ON ENDFILE(BKVALU)
39     1      BKVAL_OK = '0'B;

```

SYMT LEVEL NEST

```

40      1      ON ENDFILE(SECM*ST) BEGIN;
42      2      EMDCKE = '1'B;
43      2      IF DCDCNE = '1'B
44      2      THEN GO TO DCNE;
45      2      ELSE DO;
46      2      1      NOEQM:
47      2      1      IF DIR.COUNTY_NUM ^= LAST_EPR_CNTY
48      2      2      THEN DO:
49      2      2      PUT PAGE EDIT ('ERROR LISTING FOR ',
50      2      2      CNTY_NAMES(DIR.COUNTY_NUM),' COUNTY')(COL(30),A,A,A);
51      2      1      LAST_ERR_CNTY = DIR.COUNTY_NUM;
52      2      1      END;
53      2      1      PUT SKIP(2) EDIT ('THE EQUIPMENT NUMBER IN THE FOLLOWING DIRECT ',
54      2      1      'COST RECORD WAS NOT FOUND IN THE MASTER FILE')(A,A);
55      2      1      PUT SKIP EDIT (DIR)(A);
56      2      1      READ FILE(DIRCOST) INTO (DIR);
57      1      GO TO NOEQM;
58      2      1      END;
59      2      END;
60      1      DCL TEQMAST FILE RECORD SEQUENTIAL;
61      1      OPEN FILE(SEQMAST) INPUT;
62      1      OPEN FILE(SYSPRINT) LINESIZE(132) PAGESIZE(60);
63      1      OPEN FILE(BKVALU) INPUT;
64      1      OPEN FILE(INDCOST) INPUT;
65      1      OPEN FILE(DIRCOST) INPUT;
66      1      OPEN FILE(TEQMAST) OUTPUT;

```

STMT LEVEL NEST

```

64 1 READ FILE(INDCOST) INTO (IND);
65 1 DO WHILE (IND_OK);
66 1 1 /* FIND TOTAL INDIRECT COST COST FOR COUNTY */
    TOT_IND_COST(IND.COUNTY_NUM) =
        IND.SUP_SALARIES + IND.CLER_SALARIES
        + IND.UTILITIES + IND.BLDG_COSTS + IND.EQ_DEPR
        + IND.EXP_TOOLS + IND.SUPPLIES + IND.MOVING_COSTS
        + IND.EC_INS;
67 1 1 PEAD FILE(INDCOST) INTC (IND);
68 1 1 END;
69 1 READ FILE(BKVALU) INTO (BKVAL);
70 1 DO WHILE (BKVAL_OK);
71 1 1 TOT_BK_VAL(BKVAL.CNTY_NUM) = BKVAL.VALUE;
72 1 1 PEAD FILE(BKVALU) INTO (BKVAL);
73 1 1 END;
74 1 INDEX = 1;

/* THE MAX VALUE OF INDEX MUST BE CHANGED AS CITIES ARE ADDED */
75 1 DO WHILE (INDEX <= 99);
76 1 1 IF TOT_BK_VAL(INDEX) <= (10)'0'
77 1 1 THEN RATIO(INDEX) = TOT_IND_COST(INDEX)/TOT_BK_VAL(INDEX);
78 1 1 ELSE RATIO(INDEX) = (10)'0';
79 1 1 INDEX = INDEX + 1;
80 1 1 END;

```

STAT LEVEL NEST

```

81 1 NEXT:
    READ FILE(SFQMAST) INTO (MSTR);
82 1 READ FILE(DIRCCSY) INTO (DIR);

83 1 NEXT2:
    IF MSTR.COUNTY_NUM = DIR.COUNTY_NUM & MSTR.EC_NUM = DIR.EC_NUM
84 1 THEN DO; /* UPDATE LIFE FIGURES FOR ALL EQUIPMENT */
85 1 1 MSTR.IND_COST_YR = MSTR.BOOK_VALUE * RATIO(MSTR.COUNTY_NUM);
86 1 1 MSTR = DIR, BY NAME;
87 1 1 MSTR.MILES_HRS_LIFE = MSTR.MILES_HRS_LIFE + MSTR.MILES_HRS_YR;
88 1 1 MSTR.FUEL_COST_LIFE = MSTR.FUEL_COST_LIFE + MSTR.FUEL_COST_YR;
89 1 1 MSTR.LUBRICANTS_LIFE = MSTR.LUBRICANTS_LIFE + MSTR.LUBRICANTS_YR;
90 1 1 MSTR.TIRES_TUBES_LIFE = MSTR.TIRES_TUBES_LIFE + MSTR.TIRES_TUBES_YR;
91 1 1 MSTR.EXPEN_PARTS_LIFE = MSTR.EXPEN_PARTS_LIFE + MSTR.EXPEN_PARTS_YR;
92 1 1 MSTR.ANTIFREEZE_LIFE = MSTR.ANTIFREEZE_LIFE + MSTR.ANTIFREEZE_YR;
93 1 1 MSTR.PARTS_COST_LIFE = MSTR.PARTS_COST_LIFE + MSTR.PARTS_COST_YR;
94 1 1 MSTR.LABOR_COST_LIFE = MSTR.LABOR_COST_LIFE + MSTR.LABOR_COST_YR;
95 1 1 MSTR.DOWN_TIME_LIFE = MSTR.DOWN_TIME_LIFE + MSTR.DOWN_TIME_YR;
96 1 1 MSTR.NUM_REPAIRS_LIFE = MSTR.NUM_REPAIRS_LIFE + MSTR.NUM_REPAIRS_YR;
97 1 1 MSTR.IND_COST_LIFE = MSTR.IND_COST_LIFE + MSTR.IND_COST_YR;
98 1 1 MSTR.DEPRECIATION = (MSTR.ORIG_COST - MSTR.SALVAGE_VALUE)
    / DEPRECIATION_YRS(MSTR.CLASS_CODE);
99 1 1 TEMP_BK_VAL = MSTR.BOOK_VALUE - MSTR.DEPRECIATION;
100 1 1 IF TEMP_BK_VAL > MSTR.SALVAGE_VALUE
101 1 1 THEN MSTR.BOOK_VALUE = TEMP_BK_VAL;
102 1 1 ELSE MSTR.BOOK_VALUE = MSTR.SALVAGE_VALUE;
103 1 1 WRITE FILE(TEQMAST) FROM (MSTR);
104 1 1 GO TO NEXT;
105 1 1 END;

```

STATE LEVEL NEST

```

106 1      IF MSTR.COUNTY_NUM < DIR.COUNTY_NUM |
      (MSTR.COUNTY_NUM = DIR.COUNTY_NUM & MSTR.EQ_NUM < DIR.EQ_NUM)
107 1      THEN DO;
108 1 1      IF MSTR.COUNTY_NUM = LAST_ERR_CNTY
109 1 1      THEN DO;
110 1 2      PUT PAGE EDIT ('ERROR LISTING FOR ',
      CNTY_NAMES(MSTR.COUNTY_NUM), ' COUNTY')(COL(30),A,A,A);
111 1 2      LAST_ERR_CNTY = MSTR.COUNTY_NUM;
112 1 2      END;
113 1 1      PUT SKIP(2) EDIT ('NO DIRECT COST CARD WAS FOUND FOR THE ',
      'FOLLOWING PIECE OF EQUIPMENT:')(A,A);
114 1 1      PUT SKIP EDIT ('EQUIPMENT NO.: ',MSTR.EQ_NUM,'MAKE & MODEL: ',
      MSTR.MAKE_MODEL,'SERIAL NO.: ',MSTR.SERIAL_NUM)
      (A,A,X(4),A,A,X(4),A,A);
115 1 1      WRITE FILE(TECMAST) FROM (MSTR);
116 1 1      READ FILE(SECMAST) INTO (MSTR);
117 1 1      GO TO NEXT2;
118 1 1      END;

      /* IF MSTR.COUNTY_NUM > DIR.COUNTY_NUM | MSTR.EQ_NUM > DIR.EQ_NUM */
119 1      ELSE DO;
120 1 1      IF DIR.COUNTY_NUM = LAST_ERR_CNTY
121 1 1      THEN DO;
122 1 2      PUT PAGE EDIT ('ERROR LISTING FOR ',
      CNTY_NAMES(DIR.COUNTY_NUM), ' COUNTY')(COL(30),A,A,A);
123 1 2      LAST_ERR_CNTY = DIR.COUNTY_NUM;
124 1 2      END;
125 1 1      PUT SKIP(2) EDIT ('THE EQUIPMENT NUMBER IN THE FOLLOWING DIRECT ',
      'COST RECORD WAS NOT FOUND IN THE MASTER FILE')(A,A);
126 1 1      PUT SKIP EDIT (DIR)(A);
127 1 1      READ FILE(DIRCOST) INTO (DIR);
128 1 1      GO TO NEXT2;
129 1 1      END;
130 1      DONE;
      END FMMAIN;

```

Listing of Program MAIN

Step 3

STMT LEVEL NEST

```

1      SMAIN: PROC OPTICNS(MAIN):
2      1      DCL 1 MSTP,
          2 COUNTY_NUM          PIC'999',
          2 DISTRICT            CHAR(2),
          2 EQ_NUM              CHAR(8),
          2 CLASS_CODE          PIC'99',
          2 YEAR_MFD            CHAR(2),
          2 MFR_CODE             CHAR(3),
          2 ENGINE_MFR_CODE     CHAR(2),
          2 NUM_CYLS            CHAR(2),
          2 ENGINE_TYPE         CHAR(1),
          2 XMISSION_TYPE       CHAR(1),
          2 ORIG_COST           PIC'999999V99',
          2 SALVAGE_VALUE       PIC'999999V99',
          2 BOOK_VALUE          PIC'999999V99',
          2 DEPRECIATION        PIC'999999V99',
          2 MILES_HRS_YR        PIC'999999',
          2 FUEL_COST_YR        PIC'999999V99',
          2 LUBRICANTS_YR       PIC'9999V99',
          2 TIRES_TUBES_YR      PIC'9999V99',
          2 EXPEN_PARTS_YR      PIC'9999V99',
          2 ANTIFREEZE_YR       PIC'999V99',
          2 PARTS_COST_YR       PIC'999999V99',
          2 LABDR_COST_YR       PIC'999999V99',
          2 IND_COST_YR         PIC'999999V99',
          2 MILES_HRS_LIFE      PIC'999999',
          2 FUEL_COST_LIFE      PIC'999999V99',
          2 LUBRICANTS_LIFE     PIC'9999V99',
          2 TIRES_TUBES_LIFE    PIC'9999V99',
          2 EXPEN_PARTS_LIFE    PIC'9999V99',
          2 ANTIFREEZE_LIFE     PIC'9999V99',
          2 PARTS_COST_LIFE     PIC'999999V99',
          2 LABUR_COST_LIFE     PIC'999999V99',
          2 IND_COST_LIFE       PIC'999999V99',
          2 CLASS_DESC          CHAR(20),
          2 MAKE_MODEL          CHAR(14),
          2 SERIAL_NUM          CHAR(14),
          2 ENGINE_DESC         CHAR(14),
          2 CC_PURCHASED_FPCM   CHAR(12),
          2 DATE_PURCHASED,
            3 PDCN              CHAR(2),
            3 PCAY              CHAR(2),
            3 PYR               CHAR(2),
          2 DATE_DISPOSED,
            3 DMCN              CHAR(2),
            3 DDAY              CHAR(2),
            3 CYR               CHAR(2),
          2 DISPOSITION        CHAR(1),
          2 WHEELBASE          CHAR(3),
          2 DOWN_TIME_YR        PIC'9999',
          2 DOWN_TIME_LIFE     PIC'9999',
          2 NUM_REPAIRS_YR      PIC'99',
          2 NUM_REPAIRS_LIFE    PIC'999',
          2 RATED_HP            CHAR(3),
          2 BLANKS              CHAR(24):

```

STMT LEVEL NEST

```

3      1      DCL  CNTY_NAMES(99) CHAR(13) VARYING
          INIT('ADAIR','ADAMS','ALIAMAKEE','APPANOCSE','AUDUBON','BENTON',
          'BLACK HAWK','ECLINE','BREMER','RUFAMAN','BUENA VISTA','BUTLER',
          'CALHOUN','CARROLL','CASS','CEDAR','CERRO GORDO','CHERCKEE',
          'CHICKASAW','CLARKE','CLAY','CLAYTON','CLINTON','CRAWFORD','DALLAS',
          'DAVIS','DECATUR','DELAWARE','DES MOINES','DICKINSON','DUBUQUE',
          'EMMET','FAYETTE','FLOYD','FRANKLIN','FREMONT','GREENE','GRUNDY',
          'GUTHRIE','HAMILTON','HANCOCK','HARDIN','HARRISON','HENRY',
          'HOWARD','HUMBOLDT','IDA','IOWA','JACKSON','JASPER','JEFFERSON',
          'JOHNSON','JONES','KEOKUK','KOSSUTH','LEE','LINN','LOUISA','LUCAS',
          'LYON','MADISON','MAHASKA','MARION','MARSHALL','MILLS','MITCHELL',
          'MONONA','MONROE','MONTGOMERY','MUSCATINE','O'BRIEN','OSCEOLA',
          'PAGE','PALM ALTO','PLYMOUTH','POCAHONTAS','POLK','POTTAWATTAMIE',
          'POWESHIEK','RINGGOLD','SAC','SCOTT','SHELBY','SIOUX','STORY',
          'TAMA','TAYLOR','UNION','VAN BUREN','WAPELLO','WARREN',
          'WASHINGTON','WAYNE','WEBSTER','WINNEBAGO','WINNESHIEK','WOODBURY',
          'WORTH','WRIGHT');

4      1      DCL  DEPRECN_LIFE          PIC'999999V99';
5      1      DCL  DIF_CPMH             PIC'99V999';
6      1      DCL  DIR_CPMH_L           PIC'99V999';
7      1      DCL  DIR_IND_CPMH         PIC'99V999';
8      1      DCL  DIR_IND_CPMH_L       PIC'99V999';
9      1      DCL  DEPR_D_I_CPMH        PIC'99V999';
10     1      DCL  DEPR_D_I_CPMH_L      PIC'99V999';
11     1      DCL  TOT_DIR               PIC'99999999V999';
12     1      DCL  TOT_DIR_L            PIC'99999999V999';
13     1      DCL  TOT_COST              PIC'99999999V999';
14     1      DCL  TOT_COST_L           PIC'99999999V999';
15     1      DCL  LAST_CNTY_NUM        PIC'999';
16     1      DCL  LAST_CLASS            PIC'99';
17     1      DCL  LAST_EQ_NUM          CHAR(8);
    
```


SYMT LEVEL NEST

18	1	DCL CLASS_MI_HRS	PIC'9999999'	INIT((7)'0');
19	1	DCL CLASS_FUEL	PIC'999999V99'	INIT((8)'0');
20	1	DCL CLASS_LUB	PIC'9999V99'	INIT((6)'0');
21	1	DCL CLASS_ANTIFZ	PIC'9999V99'	INIT((6)'0');
22	1	DCL CLASS_TIRES	PIC'999999V99'	INIT((8)'0');
23	1	DCL CLASS_EXPEN_PARTS	PIC'999999V99'	INIT((7)'0');
24	1	DCL CLASS_REPAIR_PARTS	PIC'999999V99'	INIT((7)'0');
25	1	DCL CLASS_REPAIR_LABOR	PIC'999999V99'	INIT((7)'0');
26	1	DCL CLASS_TOT_DIR	PIC'999999V99'	INIT((8)'0');
27	1	DCL CLASS_DOWN_TIME	PIC'99999'	INIT((5)'0');
28	1	DCL CLASS_NUM_REPAIRS	PIC'99999'	INIT((4)'0');
29	1	DCL CLASS_TOT_IND	PIC'999999V99'	INIT((8)'0');
30	1	DCL CLASS_DEPRECN	PIC'999999V99'	INIT((8)'0');
31	1	DCL CLASS_COST	PIC'9999999V99'	INIT((9)'0');
32	1	DCL CLASS_DIR_CPMH	PIC'999V999'	INIT((5)'0');
33	1	DCL CLASS_D_I_CPMH	PIC'999V999'	INIT((5)'0');
34	1	DCL CLASS_D_D_I_CPMH	PIC'999V999'	INIT((5)'0');
35	1	DCL CLASS_MI_HRS_L	PIC'9999999'	INIT((7)'0');
36	1	DCL CLASS_FUEL_L	PIC'999999V99'	INIT((8)'0');
37	1	DCL CLASS_LUB_L	PIC'9999V99'	INIT((6)'0');
38	1	DCL CLASS_ANTIFZ_L	PIC'9999V99'	INIT((6)'0');
39	1	DCL CLASS_TIRES_L	PIC'999999V99'	INIT((8)'0');
40	1	DCL CLASS_EXPEN_PARTS_L	PIC'999999V99'	INIT((7)'0');
41	1	DCL CLASS_REPAIR_PARTS_L	PIC'999999V99'	INIT((7)'0');
42	1	DCL CLASS_REPAIR_LABOR_L	PIC'999999V99'	INIT((7)'0');
43	1	DCL CLASS_TOT_DIR_L	PIC'999999V99'	INIT((8)'0');
44	1	DCL CLASS_DOWN_TIME_L	PIC'99999'	INIT((5)'0');
45	1	DCL CLASS_NUM_REPAIRS_L	PIC'99999'	INIT((4)'0');
46	1	DCL CLASS_TOT_IND_L	PIC'999999V99'	INIT((8)'0');
47	1	DCL CLASS_DEPRECN_L	PIC'999999V99'	INIT((8)'0');
48	1	DCL CLASS_COST_L	PIC'9999999V99'	INIT((9)'0');
49	1	DCL CLASS_DIR_CPMH_L	PIC'999V999'	INIT((5)'0');
50	1	DCL CLASS_D_I_CPMH_L	PIC'999V999'	INIT((5)'0');
51	1	DCL CLASS_D_D_I_CPMH_L	PIC'999V999'	INIT((5)'0');
52	1	DCL CNTY_FUEL	PIC'9999999V99'	INIT((9)'0');
53	1	DCL CNTY_LUB	PIC'999999V99'	INIT((7)'0');
54	1	DCL CNTY_ANTIFZ	PIC'999999V99'	INIT((6)'0');
55	1	DCL CNTY_TIRES	PIC'999999V99'	INIT((8)'0');
56	1	DCL CNTY_EXPEN_PARTS	PIC'999999V99'	INIT((8)'0');
57	1	DCL CNTY_REPAIR_PARTS	PIC'999999V99'	INIT((8)'0');
58	1	DCL CNTY_REPAIR_LABOR	PIC'999999V99'	INIT((8)'0');
59	1	DCL CNTY_TOT_DIR	PIC'9999999V99'	INIT((9)'0');
60	1	DCL CNTY_TOT_IND	PIC'9999999V99'	INIT((9)'0');
61	1	DCL CNTY_DEPRECN	PIC'9999999V99'	INIT((9)'0');
62	1	DCL CNTY_COST	PIC'9999999V99'	INIT((9)'0');

STMT LEVEL NEST

```

63 1 DIR_FORM: FORMAT(CCL(3),A,CCL(15),A,CCL(31),P'ZZZZZ9',COL(39),
P'ZZZZ9V.99',COL(49),P'ZZZ9V.99',COL(59),P'ZZ9V.99',
CCL(67),P'ZZZZ9V.99',CCL(77),P'ZZZZ9V.99',COL(87),
P'ZZZZ9V.99',CCL(97),P'ZZZZ9V.99',CCL(108),P'ZZZZ9V.99',
COL(123),P'Z9V.999');

64 1 TOT_FORM: FORMAT(CCL(3),A,CCL(12),A,CCL(27),P'ZZZZZ9',COL(36),
P'ZZZZ9',COL(43),P'ZZZ9',COL(52),P'ZZZZ9V.99',COL(64),
P'ZZZZ9V.99',COL(76),P'ZZZZ9V.99',COL(88),P'ZZZZ9V.99',
COL(101),P'Z9V.999',COL(112),P'Z9V.999',COL(123),
P'Z9V.999');

65 1 GRAND_DIR: FORMAT(CCL(38),P'ZZZZ9V.99',COL(48),P'ZZZZ9V.99',COL(58),
P'ZZZ9V.99',COL(66),P'ZZZZ9V.99',CCL(76),P'ZZZZ9V.99',
CCL(86),P'ZZZZ9V.99',CCL(96),P'ZZZZ9V.99',CCL(107),
P'ZZZZ9V.99');

66 1 GRAND_TOT: FORMAT(CCL(51),P'ZZZZ9V.99',COL(63),P'ZZZZ9V.99',
COL(75),P'ZZZZ9V.99',COL(88),P'ZZZZ9V.99');

67 1 DCL LAST_YEAR FIXED DEC(4);
68 1 DCL 1 TODAYS_DATE,
2 C_YEAR CHAR(2),
2 C_MON_DAY CHAR(4);

69 1 TODAYS_DATE = DATE;
70 1 LAST_YEAR = 1899 + C_YEAR;
    
```

STMT LEVEL NEST

```

71 1 DCL YRDIR PRINT FILE;
72 1 DCL YFTOT PRINT FILE;
73 1 DCL LIFEDIR PRINT FILE;
74 1 DCL LIFETOT PRINT FILE;
75 1 DCL FEQMAST FILE RECCRD SEQUENTIAL;

76 1 ON ENDFILE(FEQMAST) GO TO DCNE;

78 1 ON ENDPAGE(YRDIR) BEGIN;
80 2 CALL HEADINGS;
81 2 END;

82 1 OPEN FILE(FEQMAST) INPUT;
83 1 OPEN FILE(YRDIR) LINESIZE(132) PAGESIZE(60);
84 1 OPEN FILE(YRTOY) LINESIZE(132) PAGESIZE(60);
85 1 OPEN FILE(LIFEDIR) LINESIZE(132) PAGESIZE(60);
86 1 OPEN FILE(LIFETOT) LINESIZE(132) PAGESIZE(60);
    
```

STMT LEVEL NEST

```

87 1 READ FILE(EQMAST) INTO (MSTR);
88 1 LAST_CNTY_NUM = COUNTY_NUM;

89 1 NEXTCOUNTY:
CALL HEADINGS;

90 1 NEXTCLASS:
PUT SKIP(2) FILE(YRDIR) EDIT('EQUIPMENT CLASS - ',CLASS_DESC)
(X(3),A,A);
91 1 PUT SKIP(2) FILE(YRTOT) EDIT('EQUIPMENT CLASS - ',CLASS_DESC)
(X(3),A,A);
92 1 PUT SKIP(2) FILE(LIFEDIR) EDIT('EQUIPMENT CLASS - ',CLASS_DESC)
(X(3),A,A);
93 1 PUT SKIP(2) FILE(LIFETOT) EDIT('EQUIPMENT CLASS - ',CLASS_DESC)
(X(3),A,A);

94 1 NEXTEQ:
TOT_DIR = FUEL_COST_YR + LUBRICANTS_YR + TIRES_TUBES_YR +
EXPEN_PARTS_YR + ANTIFREEZE_YR + PARTS_COST_YR +
LABCR_COST_YR;
95 1 TOT_DIR_L = FUEL_COST_LIFE + LUBRICANTS_LIFE + TIRES_TUBES_LIFE +
EXPEN_PARTS_LIFE + ANTIFREEZE_LIFE + PARTS_COST_LIFE +
LABCR_COST_LIFE;
96 1 DEPRECN_LIFE = ORIG_COST - BOOK_VALUE;
97 1 TOT_COST = TOT_DIR + IND_COST_YR + DEPRECIATION;
98 1 TOT_COST_L = TOT_DIR_L + IND_COST_LIFE + DEPRECN_LIFE;

99 1 IF MILES_HRS_YR ^= (6)'0'
100 1 THEN DO;
101 1 1 DIR_CPMH = TOT_DIR / MILES_HRS_YR;
102 1 1 DIR_IND_CPMH = (TOT_DIR + IND_COST_YR) / MILES_HRS_YR;
103 1 1 DEPR_D_I_CPMH = TOT_COST / MILES_HRS_YR;
104 1 1 END;
105 1 ELSE DO;
106 1 1 DIR_CPMH = (5)'0';
107 1 1 DIR_IND_CPMH = (5)'0';
108 1 1 DEPR_D_I_CPMH = (5)'0';
109 1 1 END;

110 1 IF MILES_HRS_LIFE ^= (6)'0'
111 1 THEN DO;
112 1 1 DIR_CPMH_L = TOT_DIR_L / MILES_HRS_LIFE;
113 1 1 DIR_IND_CPMH_L = (TOT_DIR_L + IND_COST_LIFE) / MILES_HRS_LIFE;
114 1 1 DEPR_D_I_CPMH_L = TOT_COST_L / MILES_HRS_LIFE;
115 1 1 END;
116 1 ELSE DO;
117 1 1 DEPR_D_I_CPMH_L = (5)'0';
118 1 1 DIR_IND_CPMH_L = (5)'0';
119 1 1 DIR_CPMH_L = (5)'0';
120 1 1 END;

121 1 PUT FILE(YRDIR) EDIT(EQ_NUM,MAKE_MODEL,MILES_HRS_YR,FUEL_COST_YR,
LUBRICANTS_YR,ANTIFREEZE_YR,TIRES_TUBES_YR,EXPEN_PARTS_YR,
PARTS_COST_YR,LABCR_COST_YR,TOT_DIR,DIR_CPMH)(R(DIR_FORM));

```

STMT LEVEL NEST

```

122 1 PUT FILE(LIFEDI?) EDIT(EQ_NUM,MAKE_MODEL,MILES_HRS_LIFE,
      FUEL_COST_LIFE,LUBRICANTS_LIFE,ANTIFREEZE_LIFE,
      TIRES_TUBES_LIFE,EXPEN_PARTS_LIFE,PARTS_COST_LIFE,
      LABOR_COST_LIFE,TOT_DIR_L,DIR_CPMH_L)(R(OIR_FORM));

123 1 PUT FILE(YRTOT) EDIT(EQ_NUM,MAKE_MODEL,MILES_HRS_YR,DOWN_TIME_YR,
      NUM_REPAIRS_YR,TOT_DIR,IND_COST_YR,DEPRECIATION,TOT_COST,
      DIR_CPMH,DIR_IND_CPMH,DEPR_D_I_CPMH)(P(TOT_FORM));

124 1 PUT FILE(LIFETOT) EDIT(EQ_NUM,MAKE_MODEL,MILES_HRS_LIFE,
      DOWN_TIME_LIFE,NUM_REPAIRS_LIFE,TOT_DIR_L,IND_COST_LIFE,
      DEPRECN_LIFE,TOT_COST_L,DIR_CPMH_L,DIR_IND_CPMH_L,
      DEPR_D_I_CPMH_L)(P(TOT_FORM));

125 1 CLASS_MI_HRS = CLASS_MI_HRS + MILES_HRS_YR;
126 1 CLASS_MI_HRS_L = CLASS_MI_HRS_L + MILES_HRS_LIFE;
127 1 CLASS_FUEL = CLASS_FUEL + FUEL_COST_YR;
128 1 CLASS_FUEL_L = CLASS_FUEL_L + FUEL_COST_LIFE;
129 1 CLASS_LUB = CLASS_LUB + LUBRICANTS_YR;
130 1 CLASS_LUB_L = CLASS_LUB_L + LUBRICANTS_LIFE;
131 1 CLASS_ANTIFZ = CLASS_ANTIFZ + ANTIFREEZE_YR;
132 1 CLASS_ANTIFZ_L = CLASS_ANTIFZ_L + ANTIFREEZE_LIFE;
133 1 CLASS_TIRES = CLASS_TIRES + TIRES_TUBES_YR;
134 1 CLASS_TIRES_L = CLASS_TIRES_L + TIRES_TUBES_LIFE;
135 1 CLASS_EXPEN_PARTS = CLASS_EXPEN_PARTS + EXPEN_PARTS_YR;
136 1 CLASS_EXPEN_PARTS_L = CLASS_EXPEN_PARTS_L + EXPEN_PARTS_LIFE;
137 1 CLASS_REPAIR_PARTS = CLASS_REPAIR_PARTS + PARTS_COST_YR;
138 1 CLASS_REPAIR_PARTS_L = CLASS_REPAIR_PARTS_L + PARTS_COST_LIFE;
139 1 CLASS_REPAIR_LABOR = CLASS_REPAIR_LABOR + LABOR_COST_YR;
140 1 CLASS_REPAIR_LABOR_L = CLASS_REPAIR_LABOR_L + LABOR_COST_LIFE;
141 1 CLASS_TOT_DIR = CLASS_TOT_DIR + TOT_DIR;
142 1 CLASS_TOT_DIR_L = CLASS_TOT_DIR_L + TOT_DIR_L;
143 1 CLASS_DOWN_TIME = CLASS_DOWN_TIME + DOWN_TIME_YR;
144 1 CLASS_DOWN_TIME_L = CLASS_DOWN_TIME_L + DOWN_TIME_LIFE;
145 1 CLASS_NUM_REPAIRS = CLASS_NUM_REPAIRS + NUM_REPAIRS_YR;
146 1 CLASS_NUM_REPAIRS_L = CLASS_NUM_REPAIRS_L + NUM_REPAIRS_LIFE;
147 1 CLASS_TOT_IND = CLASS_TOT_IND + IND_COST_YR;
148 1 CLASS_TOT_IND_L = CLASS_TOT_IND_L + IND_COST_LIFE;
149 1 CLASS_DEPRECN = CLASS_DEPRECN + DEPRECIATION;
150 1 CLASS_DEPRECN_L = CLASS_DEPRECN_L + DEPRECN_LIFE;
151 1 CLASS_COST = CLASS_COST + TOT_COST;
152 1 CLASS_COST_L = CLASS_COST_L + TOT_COST_L;
153 1 LAST_EQ_NUM = EQ_NUM;
154 1 LAST_CLASS = CLASS_CODE;
155 1 LAST_CNTY_NUM = COUNTY_NUM;

```

STMT LEVEL NEST

```

156 1 1 NEXTRECORD:
157 1 READ FILE(FECMAST) INTO (MSTR);
158 1 IF COUNTY_NUM = LAST_CNTY_NUM
159 1 1 THEN DO:
160 1 1 CALL UPCNTY;
161 1 1 CALL PRCLASSY;
162 1 1 PUT SKIP(2) FILE(YRDIR) EDIT ('GRAND TOTALS',CNTY_FUEL,CNTY_LUB,
CNTY_ANTIFZ,CNTY_TIRES,CNTY_EXPEN_PARTS,CNTY_REPAIR_PARTS,
CNTY_REPAIR_LABOR,CNTY_TOT_DIR)(X(6),A,R(GRAND_DIR));
163 1 1 PUT SKIP(2) FILE(YRTOT) EDIT ('GRAND TOTALS',CNTY_TOT_DIR,
CNTY_TOT_INC,CNTY_DEPRECN,CNTY_COST)(X(6),A,R(GRAND_TOT));
164 1 1 CLASS_MH_HRS = (7)'0';
165 1 1 CLASS_FUEL = (8)'0';
166 1 1 CLASS_LUB = (6)'0';
167 1 1 CLASS_ANTIFZ = (6)'0';
168 1 1 CLASS_TIRES = (8)'0';
169 1 1 CLASS_EXPEN_PARTS = (7)'0';
170 1 1 CLASS_REPAIR_PARTS = (7)'0';
171 1 1 CLASS_REPAIR_LABOR = (7)'0';
172 1 1 CLASS_TOT_DIR = (8)'0';
173 1 1 CLASS_DOWN_TIME = (5)'0';
174 1 1 CLASS_NUM_REPAIRS = (4)'0';
175 1 1 CLASS_TOT_INC = (8)'0';
176 1 1 CLASS_DEPRECN = (8)'0';
177 1 1 CLASS_COST = (9)'0';
178 1 1 CLASS_DIR_CPMH = (5)'0';
179 1 1 CLASS_D_I_CPMH = (5)'0';
180 1 1 CLASS_D_D_I_CPMH = (5)'0';
181 1 1 CLASS_MH_HRS_L = (7)'0';
182 1 1 CLASS_FUEL_L = (8)'0';
183 1 1 CLASS_LUB_L = (6)'0';
184 1 1 CLASS_ANTIFZ_L = (6)'0';
185 1 1 CLASS_TIRES_L = (8)'0';
186 1 1 CLASS_EXPEN_PARTS_L = (7)'0';
187 1 1 CLASS_REPAIR_PARTS_L = (7)'0';
188 1 1 CLASS_REPAIR_LABOR_L = (7)'0';
189 1 1 CLASS_TOT_DIR_L = (8)'0';
190 1 1 CLASS_DOWN_TIME_L = (5)'0';
191 1 1 CLASS_NUM_REPAIRS_L = (4)'0';
192 1 1 CLASS_TOT_INC_L = (8)'0';
193 1 1 CLASS_DEPRECN_L = (8)'0';
194 1 1 CLASS_COST_L = (9)'0';
195 1 1 CLASS_DIR_CPMH_L = (5)'0';
196 1 1 CLASS_D_I_CPMH_L = (5)'0';
197 1 1 CLASS_D_D_I_CPMH_L = (5)'0';
198 1 1 CNTY_FUEL = (9)'0';
199 1 1 CNTY_LUB = (7)'0';
200 1 1 CNTY_ANTIFZ = (6)'0';
201 1 1 CNTY_TIRES = (8)'0';
202 1 1 CNTY_EXPEN_PARTS = (8)'0';
203 1 1 CNTY_REPAIR_PARTS = (8)'0';
204 1 1 CNTY_REPAIR_LABOR = (8)'0';
205 1 1 CNTY_TOT_DIR = (9)'0';
206 1 1 CNTY_TOT_INC = (9)'0';
207 1 1 CNTY_DEPRECN = (9)'0';

```

STMT LEVEL NEST

207	1	1	CNTY_CCST	= (9)0*
208	1	1	GO TO NEXTCOUNTY;	
209	1	1	END;	

STMT LEVEL NEST

```

210 1      IF  LAST_CLASS = CLASS_CODE
211 1      THEN IF  EQ_NUM = LAST_EQ_NUM
212 1          THEN GO TO NEXTRECORD;
213 1          ELSE GO TO NEXTEQ;
214 1      ELSE;
215 1      CALL PPCLASST;
216 1      CALL UPONTY;
217 1          CLASS_MI_HRS           = (7)'0';
218 1          CLASS_FUEL             = (8)'0';
219 1          CLASS_LUB               = (6)'0';
220 1          CLASS_ANTIFZ           = (6)'0';
221 1          CLASS_TIRES            = (8)'0';
222 1          CLASS_EXPEN_PARTS       = (7)'0';
223 1          CLASS_REPAIR_PARTS      = (7)'0';
224 1          CLASS_REPAIR_LABOR      = (7)'0';
225 1          CLASS_TOT_DIR           = (8)'0';
226 1          CLASS_DOWN_TIME         = (5)'0';
227 1          CLASS_NUM_REPAIRS       = (4)'0';
228 1          CLASS_TOT_IND           = (8)'0';
229 1          CLASS_DEPRECN           = (8)'0';
230 1          CLASS_COST              = (9)'0';
231 1          CLASS_DIR_CPMH          = (5)'0';
232 1          CLASS_D_I_CPMH         = (5)'0';
233 1          CLASS_D_D_I_CPMH       = (5)'0';
234 1          CLASS_MI_HRS_L         = (7)'0';
235 1          CLASS_FUEL_L           = (8)'0';
236 1          CLASS_LUB_L             = (6)'0';
237 1          CLASS_ANTIFZ_L         = (6)'0';
238 1          CLASS_TIRES_L          = (8)'0';
239 1          CLASS_EXPEN_PARTS_L     = (7)'0';
240 1          CLASS_REPAIR_PARTS_L   = (7)'0';
241 1          CLASS_REPAIR_LABOR_L   = (7)'0';
242 1          CLASS_TOT_DIR_L        = (8)'0';
243 1          CLASS_DOWN_TIME_L      = (5)'0';
244 1          CLASS_NUM_REPAIRS_L    = (4)'0';
245 1          CLASS_TOT_IND_L        = (8)'0';
246 1          CLASS_DEPRECN_L       = (8)'0';
247 1          CLASS_COST_L           = (9)'0';
248 1          CLASS_DIR_CPMH_L       = (5)'0';
249 1          CLASS_D_I_CPMH_L      = (5)'0';
250 1          CLASS_D_D_I_CPMH_L    = (5)'0';
251 1      GO TO NEXTCLASS;

```



```

STMT LEVEL NEST
252 1 PRCLASST: PRCC;
253 2 CLASS_DIR: FORMAT (COL(30),P'ZZZZZ9',COL(38),P'ZZZZ9V.99',COL(49),
P'ZZZ9V.99',COL(58),P'ZZZ9V.99',COL(67),P'ZZZ9V.99',
COL(77),P'ZZZ9V.99',COL(87),P'ZZZ9V.99',COL(97),
P'ZZZ9V.99',COL(108),P'ZZZZ9V.99',COL(123),P'Z9V.999');
254 2 CLASS_TOT: FORMAT (COL(26),P'ZZZZZ9',COL(36),P'ZZZZ9',COL(43),P'ZZZ9',
COL(52),P'ZZZZ9V.99',COL(64),P'ZZZZ9V.99',COL(76),
P'ZZZZ9V.99',COL(88),P'ZZZZ9V.99',COL(101),P'Z9V.999',
COL(112),P'Z9V.999',COL(123),P'Z9V.999');
255 2 IF CLASS_MI_HRS = (7)'0'
256 2 THEN DO;
257 2 1 CLASS_DIR_CPMH = CLASS_TOT_DIR / CLASS_MI_HRS;
258 2 1 CLASS_D_I_CPMH = (CLASS_TOT_DIR+CLASS_TOT_IND)/CLASS_MI_HRS;
259 2 1 CLASS_D_D_I_CPMH = CLASS_COST / CLASS_MI_HRS;
260 2 1 END;
261 2 ELSE DO;
262 2 1 CLASS_D_I_CPMH = (5)'0';
263 2 1 CLASS_D_D_I_CPMH = (5)'0';
264 2 1 CLASS_DIR_CPMH = (5)'0';
265 2 1 END;
266 2 IF CLASS_MI_HRS_L = (7)'0'
267 2 THEN DO;
268 2 1 CLASS_DIR_CPMH_L = CLASS_TOT_DIR_L / CLASS_MI_HRS_L;
269 2 1 CLASS_D_I_CPMH_L = (CLASS_TOT_DIR_L + CLASS_TOT_IND_L) /
CLASS_MI_HRS_L;
270 2 1 CLASS_D_D_I_CPMH_L = CLASS_COST_L / CLASS_MI_HRS_L;
271 2 1 END;
272 2 ELSE DO;
273 2 1 CLASS_D_I_CPMH_L = (5)'0';
274 2 1 CLASS_D_D_I_CPMH_L = (5)'0';
275 2 1 CLASS_DIR_CPMH_L = (5)'0';
276 2 1 END;
277 2 PUT SKIP(2) FILE(YRDIR) EDIT ('CLASS TOTALS',CLASS_MI_HRS,CLASS_FUEL,
CLASS_LUB,CLASS_ANTIFZ,CLASS_TIRES,CLASS_EXPEN_PARTS,
CLASS_REPAIR_PARTS,CLASS_REPAIR_LABOR,CLASS_TOT_DIR,
CLASS_DIR_CPMH)
(X(6),A,R(CLASS_DIR));
278 2 PUT SKIP(2) FILE(YR TOT) EDIT ('CLASS TOTALS',CLASS_MI_HRS,
CLASS_DWN_TIME,CLASS_NUM_REPAIRS,CLASS_TOT_DIR,
CLASS_TOT_IND,CLASS_DEPRECN,CLASS_COST,CLASS_DIR_CPMH,
CLASS_D_I_CPMH,CLASS_D_D_I_CPMH)
(X(6),A,R(CLASS_TOT));
279 2 PUT SKIP(2) FILE(LIFEDIR) EDIT ('CLASS TOTALS',CLASS_MI_HRS_L,
CLASS_FUEL_L,CLASS_LUB_L,CLASS_ANTIFZ_L,CLASS_TIRES_L,
CLASS_EXPEN_PARTS_L,CLASS_REPAIR_PARTS_L,
CLASS_REPAIR_LABOR_L,CLASS_TOT_DIR_L,CLASS_DIR_CPMH_L)
(X(6),A,R(CLASS_DIR));

```

STMT LEVEL NEST

```
280 2 PUT SKIP(2) FILE(LIFETOT) EDIT ('CLASS TOTALS',CLASS_MI_HRS_L,  
CLASS_CCAN_TIME_L,CLASS_NUM_REPAIRS_L,CLASS_TOT_DIR_L,  
CLASS_TOT_IND_L,CLASS_DEPRECN_L,CLASS_COST_L,  
CLASS_DIR_CPMH_L,CLASS_D_I_CPMH_L,CLASS_D_D_I_CPMH_L)  
281 2 (X(6),A,R(CLASS_TCT));  
END PRCLASST;
```

STMT LEVEL NEST

```

282 1 HEADINGS: PROC;
283 2 COLUMN_START = (125 - LENGTH(CNTY_NAMES(COUNTY_NUM))) / 2;
284 2 PUT PAGE FILE(YRDIR) EDIT (CNTY_NAMES(COUNTY_NUM), ' COUNTY')
      (COL(COLUMN_START),A,A);
285 2 PUT PAGE FILE(YRDT) EDIT (CNTY_NAMES(COUNTY_NUM), ' COUNTY')
      (COL(COLUMN_START),A,A);
286 2 PUT PAGE FILE(LIFEDIR) EDIT (CNTY_NAMES(COUNTY_NUM), ' COUNTY')
      (COL(COLUMN_START),A,A);
287 2 PUT PAGE FILE(LIFEDT) EDIT (CNTY_NAMES(COUNTY_NUM), ' COUNTY')
      (COL(COLUMN_START),A,A);
288 2 PUT SKIP(2) FILE(YRDIR) EDIT (LAST_YEAR,
      ' EQUIPMENT DIRECT OPERATING COSTS')(COL(48),F(4),A);
289 2 PUT SKIP(2) FILE(LIFEDIR) EDIT
      ('LIFETIME EQUIPMENT DIRECT OPERATING COSTS')(COL(46),A);
290 2 PUT SKIP FILE(YRDT) EDIT(LAST_YEAR,
      ' EQUIPMENT TOTAL OPERATING COSTS')(COL(49),F(4),A);
291 2 PUT SKIP FILE(LIFEDT) EDIT
      ('LIFETIME EQUIPMENT TOTAL OPERATING COSTS')(COL(47),A);
292 2 PUT SKIP FILE(YRDIR) EDIT('TOTAL DIRECT')(COL(110),A);
293 2 PUT SKIP FILE(YRDIR) EDIT('EQUIPMENT', 'MILES/', 'ANTI',
      'EXPENDABLE REPAIR REPAIR', 'DIRECT CPM/')(
      COL(4),A,COL(31),A,COL(60),A,COL(76),A,COL(110),A);
294 2 PUT SKIP FILE(YRDIR) EDIT('NUMBER MAKE & MODEL HOURS FUEL',
      'LUBRICANTS FREEZE TIRES PARTS PARTS LABOR',
      'COSTS CPH')(COL(15),A,COL(48),A,COL(110),A);
295 2 PUT SKIP FILE(LIFEDIR) EDIT('TOTAL DIRECT')(COL(110),A);
296 2 PUT SKIP FILE(LIFEDIR) EDIT('EQUIPMENT', 'MILES/', 'ANTI',
      'EXPENDABLE REPAIR REPAIR', 'DIRECT CPM/')(
      COL(4),A,COL(31),A,COL(60),A,COL(76),A,COL(110),A);
297 2 PUT SKIP FILE(LIFEDIR) EDIT('NUMBER MAKE & MODEL HOURS FUEL',
      'LUBRICANTS FREEZE TIRES PARTS PARTS LABOR',
      'COSTS CPH')(COL(15),A,COL(48),A,COL(110),A);
298 2 PUT SKIP(2) FILE(YRDT) EDIT('TOTAL TOTAL',
      'DIRECT + DEPREC +')(COL(54),A,COL(111),A);
299 2 PUT SKIP FILE(YRDT) EDIT ('EQUIPMENT', ' MILES/ DOWN TIMES',
      'DIRECT INDIRECT', 'TOTAL DIRECT INDIRECT DIR + INDIR')(
      COL(2),A,COL(27),A,COL(54),A,COL(91),A);
300 2 PUT SKIP FILE(YRDT) EDIT ('NUMBER MAKE & MODEL HOURS',
      'TIME REPAIRED', 'COSTS COSTS DEPRECIATION',
      'COST CPM/CPH', 'CPM/CPH CPM/CPH')(
      COL(3),A,COL(37),A,COL(54),A,COL(92),A,COL(111),A);
301 2 PUT SKIP(2) FILE(LIFEDT) EDIT('TOTAL TOTAL',
      'DIRECT + DEPREC +')(COL(54),A,COL(111),A);
302 2 PUT SKIP FILE(LIFEDT) EDIT ('EQUIPMENT', ' MILES/ DOWN TIMES',
      'DIRECT INDIRECT', 'TOTAL DIRECT INDIRECT DIR + INDIR')(
      COL(2),A,COL(27),A,COL(54),A,COL(91),A);
303 2 PUT SKIP FILE(LIFEDT) EDIT ('NUMBER MAKE & MODEL HOURS',
      'TIME REPAIRED', 'COSTS COSTS DEPRECIATION',
      'COST CPM/CPH', 'CPM/CPH CPM/CPH')(
      COL(3),A,COL(37),A,COL(54),A,COL(92),A,COL(111),A);
304 2 END HEADINGS;

```

STPT LEVEL NEST

```
305 1      UPCNTYT:  PROC;
306 2      CNTY_FUEL = CNTY_FUEL + CLASS_FUEL;
307 2      CNTY_LUB = CNTY_LUB + CLASS_LUB;
308 2      CNTY_ANTIFZ = CNTY_ANTIFZ + CLASS_ANTIFZ;
309 2      CNTY_TIRES = CNTY_TIRES + CLASS_TIRES;
310 2      CNTY_EXPEN_PARTS = CNTY_EXPEN_PARTS + CLASS_EXPEN_PARTS;
311 2      CNTY_REPAIR_PARTS = CNTY_REPAIR_PARTS + CLASS_REPAIR_PARTS;
312 2      CNTY_REPAIR_LABOR = CNTY_REPAIR_LABOR + CLASS_REPAIR_LABOR;
313 2      CNTY_TOT_DIR = CNTY_TOT_DIR + CLASS_TOT_DIR;
314 2      CNTY_TOT_IND = CNTY_TOT_IND + CLASS_TOT_IND;
315 2      CNTY_DEPRECN = CNTY_DEPRECN + CLASS_DEPRECN;
316 2      CNTY_COST = CNTY_COST + CLASS_COST;
317 2      END UPCNTYT;
```

STMT LEVEL NEST

```

318 1      DONE:
319 1      CALL UPCNTYT;
320 1      CALL PRCLASST;
320 1      PUT SKIP(2) FILE(YRDIR) EDIT ('GRAND TOTALS',CNTY_FUEL,CNTY_LUB,
      CNTY_ANTIFZ,CNTY_TIRES,CNTY_EXPEN_PARTS,CNTY_REPAIR_PARTS,
      CNTY_REPAIR_LABOR,CNTY_TOT_DIR)(X(6),A,R(GRANC_DIR));
321 1      PUT SKIP(2) FILE(YRTOT) EDIT ('GRAND TOTALS',CNTY_TOT_DIR,
      CNTY_TOT_IND,CNTY_DEPRECN,CNTY_COST)(X(6),A,R(GRANC_TOT));
322 1      END SMAIN;
    
```

Listing of Program CTYSUMRY

STAT LEVEL NEST

```

1      EQCLSUM: PROC OPTIONS(MAIN);

/*      PROGRAM CTSUMRY                                     */
/*
/*      THIS PROGRAM LISTS PAST YEAR AND LIFETIME CPM/CPH   */
/*      AVERAGES FOR EACH COUNTY, DISTRICT, AND FOR THE WHOLE */
/*      STATE, WITHIN EACH EQUIPMENT CLASS.                 */
/*
/*      THIS PROGRAM IS ONE OF EIGHT COMPUTER PROGRAMS WHICH FORM */
/*      THE "COMPUTER BASED INFORMATION SYSTEM FOR COUNTY EQUIPMENT */
/*      COST RECORDS".                                       */
/*
/*      WRITTEN BY                                           */
/*      SYSTEMS DIVISION                                     */
/*      COLLEGE OF ENGINEERING                             */
/*      THE UNIVERSITY OF IOWA                             */
/*      IOWA CITY, IOWA                                   */
/*      JULY 1975                                          */
/*

2      1      DCL UEMF FILE RECORD SEQUENTIAL;
3      1      DCL OUT FILE PRINT;

/*      DECLARE THE STRUCTURE FOR THE EQUIPMENT COST RECORDS. */
/*

4      1      DCL 1 EQUIP_MSTR_FILE,
          2 COUNTY_NO          PIC'999',
          2 DISTRICT_NO       PIC'99',
          2 EQ_NUM            CHAR(8),
          2 CLASS_CODE        PIC'99',
          2 UNUSED_DATA_1     CHAR(42),
          2 MILES_HRS_YEAR    PIC'9999999',
          2 FUEL_COST_YEAR    PIC'99999999',
          2 LUBRICANTS_YEAR   PIC'999999',
          2 TIRES_TUBES_YEAR  PIC'9999999',
          2 EXPEN_PARTS_YEAR  PIC'9999999',
          2 ANTIFREEZE_YEAR   PIC'999999',
          2 PARTS_COST_YEAR   PIC'99999999',
          2 LABOR_COST_YEAR   PIC'99999999',
          2 INDIRECT_COST_YEAR PIC'99999999',
          2 MILES_HRS_LIFE    PIC'9999999',
          2 FUEL_COST_LIFE    PIC'99999999',
          2 LUBRICANTS_LIFE   PIC'9999999',
          2 TIRES_TUBES_LIFE  PIC'9999999',
          2 EXPEN_PARTS_LIFE  PIC'9999999',
          2 ANTIFREEZE_LIFE   PIC'999999',
          2 PARTS_COST_LIFE   PIC'99999999',
          2 LABOR_COST_LIFE   PIC'99999999',
          2 INDIRECT_COST_LIFE PIC'99999999',
          2 CLASS_DESC        CHAR(20),
          2 UNUSED_DATA_2     CHAR(84),
          2 FILLER            CHAR(26);

5      1      DCL CNTY_NAMES(99) CHAR(13) VARYING
          INIT('ADAIR','ADAMS','ALLAMAKEE','APPANOOSE','AUDUBON','BENTON',
          'BLACK HAWK','BOONE','BREMER','BUCHANAN','BUENA VISTA','BUTLER',
          'CALHOUN','CARROLL','CASS','CEDAR','CERRO GORDO','CHEROKEE',

```

STMT LEVEL NEST

'CHICKASAW', 'CLARKE', 'CLAY', 'CLAYTON', 'CLINTON', 'CRAWFORD', 'DALLAS',
 'DAVIS', 'DECATUR', 'DELAWARE', 'DES MOINES', 'DICKINSON', 'DUBUQUE',
 'EMMETT', 'FAYETTE', 'FLOYD', 'FRANKLIN', 'FREMONT', 'GREENE', 'GRUNDY',
 'GUTHRIE', 'HAMILTON', 'HANCOCK', 'HARDIN', 'HARRISON', 'HENRY',
 'HOWARD', 'HUMBOLDT', 'IDA', 'IOWA', 'JACKSON', 'JASPER', 'JEFFERSON',
 'JOHNSON', 'JONES', 'KEOKUK', 'KOSSUTH', 'LEE', 'LINN', 'LOUISA', 'MADISON',
 'LYON', 'MADISON', 'MAHASKA', 'MARION', 'MARSHALL', 'MILLS', 'MITCHELL',
 'MONROE', 'MONTGOMERY', 'MUSCATINE', 'O'BRIEN', 'OSCEOLA',
 'PAGE', 'PALO ALTO', 'PLYMOUTH', 'POCAHONTAS', 'POLK', 'POTTAWATTAMIE',
 'POWESHIEK', 'RINGGOLD', 'SAC', 'SCOTT', 'SHELBY', 'SIOUX', 'STORY',
 'TAMA', 'TAYLOR', 'UNION', 'VAN BUREN', 'WAPELLO', 'WARREN',
 'WASHINGTON', 'WAYNE', 'WEBSTER', 'WINNEBAGO', 'WINNESHIEK', 'WOODBURY',
 'WORTH', 'WRIGHT');

```

6      1      DCL (DIP_COST_YR,DIR_COST_LIFE)FIXED DEC (9) INIT (0);
7      1      DCL (TOT_DIR_COST_CTY_YR,TOT_DIR_COST_CTY_LIFE)FIXED DEC (9)
          INIT (0);
8      1      DCL (DIST_COST_YR,DIST_COST_LIFE)FIXED DEC (9) INIT(0);
9      1      DCL (STATE_COST_YR,STATE_COST_LIFE)FIXED DEC (10) INIT(0);
10     1      DCL (M_H_CTY_YR,M_H_CTY_LIFE)FIXED DEC (9) INIT (0);
11     1      DCL (M_H_DIST_YR,M_H_DIST_LIFE)FIXED DEC (9) INIT (0);
12     1      DCL (M_H_ST_YR,M_H_ST_LIFE)FIXED DEC (9) INIT (0);
13     1      DCL (VEH_CTY,VEH_DIST,VEH_STATE)FIXED DEC (10) INIT (0);
14     1      DCL (CTY_CPM_YR,CTY_CPM_LIFE,DIST_CPM_YR,DIST_CPM_LIFE,ST_CPM_YR,
          ST_CPM_LIFE)FIXED DEC (6,3) INIT (0);
15     1      DCL OLD_COUNTY FIXED DEC (3);
16     1      DCL OLD_DISTRICT FIXED DEC (2);
17     1      DCL OLD_EQUIP_CLASS FIXED DEC (2);

18     1      DCL 1 TODAYS_DATE,
          2 C_YEAR                                CHAR(2),
          2 C_MON_DAY                             CHAR(4);
19     1      DCL C_Y FIXED DEC (2);
20     1      DCL PAST_YEAR FIXED DEC (4);
21     1      TODAYS_DATE = DATE;
22     1      C_Y = C_YEAR;
23     1      PAST_YEAR = 1900 + C_Y - 1;

24     1      ON ENDFILE(UEMP) GO TO DONE;

26     1      ON ENDPAGE(OUT) BEGIN;
28     2      PLT FILE(OUT) PAGE;
29     2      PUT FILE(OUT) SKIP(2) EDIT('DIRECT OPERATING COSTS')(COL(56),A);
30     2      PUT FILE(OUT) SKIP(2) EDIT('COST PER MILE OR HOUR BY EQUIPMENT',
          ' CLASS, DISTRICT, AND COUNTY')(COL(35),A,A);
31     2      PUT FILE(OUT) SKIP(2) EDIT('COUNTY',PAST_YEAR,'LIFE','PIECES OF')
          (COL(23),A,COL(43),F(4),COL(53),A,COL(60),A);
32     2      PUT FILE(OUT) SKIP EDIT('CODE','NAME','CPM/CPH','CPM/CPH',
          'EQUIPMENT')
          (COL(20),A,COL(28),A,COL(41),A,COL(51),A,COL(60),A);
33     2      END;

34     1      OPEN FILE(OUT) PAGESIZE(60) LINESIZE(132) OUTPUT;
35     1      OPEN FILE(UEMP) INPUT;

```


STMT LEVEL NEST

```

36 1      SIGNAL ENCPAGE(OUT);
37 1      REPEAT: READ FILE(UENF) INTO(EQUIP_MSTR_FILE);
38 1      IF EQUIP_MSTR_FILE.CLASS_CCDE = 0
39 1      THEN GO TO REPEAT;
40 1      PUT FILE(OUT) SKIP(2) EDIT('EQUIPMENT CLASS - ',CLASS_DESC)
      (COL(11),A,F(20));
41 1      PUT FILE(OUT) SKIP(2) EDIT('DISTRICT ',DISTRICT_NO)
      (COL(14),A,F(2));
42 1      LOOP: M_H_CTY_YR = M_H_CTY_YR + MILES_HRS_YEAR;
43 1      M_H_CTY_LIFE = M_H_CTY_LIFE + MILES_HRS_LIFE;
44 1      DIR_COST_YR = FUEL_COST_YEAR + LUBRICANTS_YEAR + TIRES_TUBES_YEAR +
      EXPEN_PARTS_YEAR + ANTIFREEZE_YEAR + PARTS_COST_YEAR +
      LABOR_COST_YEAR;
45 1      DIR_COST_LIFE = FUEL_COST_LIFE + LUBRICANTS_LIFE + TIRES_TUBES_LIFE +
      EXPEN_PARTS_LIFE + ANTIFREEZE_LIFE + PARTS_COST_LIFE +
      LABOR_COST_LIFE;
46 1      TOT_DIR_COST_CTY_YR = TOT_DIR_COST_CTY_YR + DIR_COST_YR;
47 1      TOT_DIR_COST_CTY_LIFE = TOT_DIR_COST_CTY_LIFE + DIR_COST_LIFE;
48 1      VEH_CTY = VEH_CTY + 1;
49 1      OLD_COUNTY = COUNTY_NO;
50 1      OLD_DISTRICT = DISTRICT_NO;
51 1      OLD_EQUIP_CLASS = CLASS_CODE;
52 1      READ FILE(UENF) INTO (EQUIP_MSTR_FILE);
53 1      IF (OLD_COUNTY /= COUNTY_NO) |
      (OLD_EQUIP_CLASS < CLASS_CODE)
54 1      THEN DO:
55 1 1      IF M_H_CTY_YR /= 0
56 1 1      THEN CTY_CPM_YR = TOT_DIR_COST_CTY_YR/M_H_CTY_YR;
57 1 1      ELSE CTY_CPM_YR = 0;
58 1 1      IF M_H_CTY_LIFE /= 0
59 1 1      THEN CTY_CPM_LIFE = TOT_DIR_COST_CTY_LIFE/M_H_CTY_LIFE;
60 1 1      ELSE CTY_CPM_LIFE = 0;
61 1 1      DIST_COST_YR = DIST_COST_YR + TOT_DIR_COST_CTY_YR;
62 1 1      DIST_COST_LIFE = DIST_COST_LIFE + TOT_DIR_COST_CTY_LIFE;
63 1 1      TOT_DIR_COST_CTY_YR,TOT_DIR_COST_CTY_LIFE = 0;
64 1 1      M_H_DIST_YR = M_H_DIST_YR + M_H_CTY_YR;
65 1 1      M_H_DIST_LIFE = M_H_DIST_LIFE + M_H_CTY_LIFE;
66 1 1      M_H_CTY_YR,M_H_CTY_LIFE = 0;
67 1 1      VEH_DIST = VEH_DIST + VEH_CTY;
68 1 1      PUT FILE(OUT) SKIP EDIT(OLD_COUNTY,CNTY_NAMES(OLD_COUNTY),
      CTY_CPM_YR,CTY_CPM_LIFE,VEH_CTY)
      (COL(20),F(3),COL(25),A,COL(42),F(6,3,-2),COL(52),F(6,3,-2),
      COL(63),F(2));
69 1 1      VEH_CTY = 0;
70 1 1      END;
71 1      IF (OLD_DISTRICT /= DISTRICT_NO) | (OLD_EQUIP_CLASS < CLASS_CODE)
72 1      THEN DO:
73 1 1      IF M_H_DIST_YR /= 0
74 1 1      THEN DIST_CPM_YR = DIST_COST_YR/M_H_DIST_YR;

```

STMT LEVEL NEST

```

75     1     1     ELSE DIST_CPM_YR = 0;
76     1     1     IF M_H_DIST_LIFE /= 0
77     1     1     THEN DIST_CPM_LIFE = DIST_COST_LIFE/M_H_DIST_LIFE;
78     1     1     ELSE DIST_CPM_LIFE = 0;
79     1     1     STATE_COST_YR = STATE_COST_YR + DIST_COST_YR;
80     1     1     STATE_COST_LIFE = STATE_COST_LIFE + DIST_COST_LIFE;
81     1     1     DIST_COST_YR, DIST_COST_LIFE = 0;
82     1     1     M_H_ST_YR = M_H_ST_YR + M_H_DIST_YR;
83     1     1     M_H_ST_LIFE = M_H_ST_LIFE + M_H_DIST_LIFE;
84     1     1     M_H_DIST_YR, M_H_DIST_LIFE = 0;
85     1     1     VEH_STATE = VEH_STATE + VEH_DIST;
86     1     1     PUT FILE(OUT) SKIP EDIT('AVERAGE FOR DISTRICT ', OLD_DISTRICT,
        DIST_CPM_YR, DIST_CPM_LIFE, VEH_DIST)
        (COL(17), A, F(2), COL(42), F(6, 3, -2), COL(52), F(6, 3, -2), COL(62),
        F(3));
87     1     1     VEH_DIST = 0;
88     1     1     IF OLD_EQUIP_CLASS = CLASS_CODE
89     1     1     THEN PUT FILE(OUT) SKIP(2) EDIT('DISTRICT ', DISTRICT_NO)
        (COL(14), A, F(2));
90     1     1     END;

91     1     IF OLD_EQUIP_CLASS < CLASS_CODE
92     1     THEN DO;
93     1     1     IF M_H_ST_YR /= 0
94     1     1     THEN ST_CPM_YR = STATE_COST_YR/M_H_ST_YR;
95     1     1     ELSE ST_CPM_YR = 0;
96     1     1     IF M_H_ST_LIFE /= 0
97     1     1     THEN ST_CPM_LIFE = STATE_COST_LIFE/M_H_ST_LIFE;
98     1     1     ELSE ST_CPM_LIFE = 0;
99     1     1     STATE_COST_YR, STATE_COST_LIFE = 0;
100    1     1     M_H_ST_YR, M_H_ST_LIFE = 0;
101    1     1     PUT FILE(OUT) SKIP(2) EDIT('AVERAGE COST FOR STATE', ST_CPM_YR,
        ST_CPM_LIFE, VEH_STATE) (COL(14), A, COL(42), F(6, 3, -2), COL(52),
        F(6, 3, -2), COL(61), F(4));
102    1     1     VEH_STATE = 0;
103    1     1     PUT FILE(OUT) SKIP(2) EDIT('EQUIPMENT CLASS - ', CLASS_CESC)
        (COL(11), A, F(20));
104    1     1     PUT FILE(OUT) SKIP(2) EDIT('DISTRICT ', DISTRICT_NO)
        (COL(14), A, F(2));
105    1     1     END;

106    1     GO TO LOOP;

107    1     DONE: IF M_H_CTY_YR /= 0
108    1     THEN CTY_CPM_YR = TOT_DIR_COST_CTY_YR/M_H_CTY_YR;
109    1     ELSE CTY_CPM_YR = 0;
110    1     IF M_H_CTY_LIFE /= 0
111    1     THEN CTY_CPM_LIFE = TOT_DIR_COST_CTY_LIFE/M_H_CTY_LIFE;
112    1     ELSE CTY_CPM_LIFE = 0;
113    1     DIST_COST_YR = DIST_COST_YR + TOT_DIR_COST_CTY_YR;
114    1     DIST_COST_LIFE = DIST_COST_LIFE + TOT_DIR_COST_CTY_LIFE;
115    1     M_H_DIST_YR = M_H_DIST_YR + M_H_CTY_YR;
116    1     M_H_DIST_LIFE = M_H_DIST_LIFE + M_H_CTY_LIFE;
117    1     VEH_CIST = VEH_DIST + VEH_CTY;
118    1     IF M_H_DIST_YR /= 0

```

STMT LEVEL NEST

```

119      1      THEN DIST_CPM_YR = DIST_COST_YR/M_H_DIST_YR;
120      1      ELSE DIST_CPM_YR = 0;
121      1      IF M_H_DIST_LIFE != 0
122      1      THEN DISY_CPM_LIFE = DIST_COST_LIFE/M_H_DIST_LIFE;
123      1      ELSE DISY_CPM_LIFE = 0;
124      1      STATE_COST_YR = STATE_COST_YR + DIST_COST_YR;
125      1      STATE_COST_LIFE = STATE_COST_LIFE + DIST_COST_LIFE;
126      1      M_H_ST_YR = M_H_ST_YR + M_H_DIST_YR;
127      1      M_H_ST_LIFE = M_H_ST_LIFE + M_H_DIST_LIFE;
128      1      VEH_STATE = VEH_STATE + VEH_DIST;
129      1      IF M_H_ST_YR != 0
130      1      THEN ST_CPM_YR = STATE_COST_YR/M_H_ST_YR;
131      1      ELSE ST_CPM_YR = 0;
132      1      IF M_H_ST_LIFE != 0
133      1      THEN ST_CPM_LIFE = STATE_COST_LIFE/M_H_ST_LIFE;
134      1      ELSE ST_CPM_LIFE = 0;
135      1      PUT FILE(OUT) SKIP EDIT(OLD_COUNTY, CNTY_NAMES(OLD_COUNTY),
      CTY_CPM_YR, CTY_CPM_LIFE, VEH_CTY) (COL(20), F(3), COL(25), A,
      COL(42), F(6,3,-2), COL(52), F(6,3,-2), COL(63), F(2));
136      1      PUT FILE(OUT) SKIP EDIT('AVERAGE FOR DISTRICT ', OLD_DISTRICT,
      DIST_CPM_YR, DISY_CPM_LIFE, VEH_DIST) (COL(17), A, F(2), COL(42),
      F(6,3,-2), COL(52), F(6,3,-2), COL(62), F(3));
137      1      PUT FILE(OUT) SKIP(2) EDIT('AVERAGE COST FOR STATE', ST_CPM_YR,
      ST_CPM_LIFE, VEH_STATE) (COL(14), A, COL(42), F(6,3,-2), COL(52),
      F(6,3,-2), COL(61), F(4));
138      1      END EQCLSUM;

```

Listing of Program MFGAGE

STAY LEVEL NEST

```

1      MFRAGE: PROC OPTIONS(MAIN);

/*      PROGRAM MFRAGE                                */

/*      THIS PROGRAM LISTS PAST YEAR AND LIFETIME CPM/CPH      */
/*      AVERAGES FOR VARIOUS EQUIPMENT AGE GROUPINGS BY EQUIPMENT  */
/*      MANUFACTURERS WITHIN EACH EQUIPMENT CLASS.              */

/*      THIS PROGRAM IS ONE OF EIGHT COMPUTER PROGRAMS WHICH FORM  */
/*      THE "COMPUTER BASED INFORMATION SYSTEM FOR COUNTY EQUIPMENT  */
/*      CCST RECORDS".                                           */

/*      WRITTEN BY                                             */
/*      SYSTEMS DIVISION                                       */
/*      COLLEGE OF ENGINEERING                                 */
/*      THE UNIVERSITY OF IOWA                                 */
/*      IOWA CITY, IOWA                                       */
/*      JULY 1975                                             */

2      1      DCL UEMP FILE RECORD SEQUENTIAL;
3      1      DCL MFR FILE;
4      1      DCL OUT FILE PRINT;

/*      DECLARE THE STRUCTURE FOR THE EQUIPMENT COST RECORDS.  */

5      1      DCL 1 EQUIP_MSTR_FILE,
              2 COUNTY_NO                PIC'999',
              2 DISTRICT_NO              PIC'99',
              2 EQ_NUM                   CHAR(8),
              2 CLASS_CODE               PIC'99',
              2 YR_MFR                   PIC'99',
              2 MFR_CODE                 PIC'999',
              2 UNUSED_DATA_1           CHAR(37),
              2 MILES_HRS_YEAR           PIC'999999',
              2 FUEL_COST_YEAR           PIC'999999V99',
              2 LUBRICANTS_YEAR          PIC'999999',
              2 TIRES_TUBES_YEAR         PIC'9999V99',
              2 EXPEN_PARTS_YEAR         PIC'9999V99',
              2 ANTIFREEZE_YEAR          PIC'999999',
              2 PARTS_COST_YEAR          PIC'999999V99',
              2 LABOR_COST_YEAR          PIC'999999V99',
              2 INDIRECT_COST_YEAR       PIC'999999V99',
              2 MILES_HRS_LIFE           PIC'999999',
              2 FUEL_COST_LIFE           PIC'999999V99',
              2 LUBRICANTS_LIFE          PIC'9999V99',
              2 TIRES_TUBES_LIFE         PIC'9999V99',
              2 EXPEN_PARTS_LIFE         PIC'9999V99',
              2 ANTIFREEZE_LIFE          PIC'999999',
              2 PARTS_COST_LIFE          PIC'999999V99',
              2 LABOR_COST_LIFE          PIC'999999V99',
              2 INDIRECT_COST_LIFE       PIC'999999V99',
              2 CLASS_DESC               CHAR(20),
              2 UNUSED_DATA_2           CHAR(84),
              2 FILLER                   CHAR(26);

6      1      DCL ARRAY_ELMT FIXED DEC(3);

```

STMT LEVEL NEST

```

 7      1      DCL NAMES CHAR(23);
 8      1      DCL NO_MFR FIXED DEC(3) INIT(200);
 9      1      DCL MFR_NAMES(*) CHAR(23) VARYING CONTROLLED;
10     1      DCL MI_HR_VEH(5,3) FIXED DEC(10) INIT((15) 0);
11     1      DCL COSTS_YR_LIFE(5,2) FIXED DEC(10) INIT((10) 0);
12     1      DCL TOTALS(5,2) FIXED DEC(5,3) INIT((10) 0);
13     1      DCL OLD_EQUIP_CLASS FIXED DEC(2);
14     1      DCL OLD_MFR_CODE FIXED DEC(3);
15     1      DCL AGE(5) CHAR(7) VARYING
          INIT('0-3', '4-6', '7-9', '10-12', 'OVER 12');
16     1      DCL TODAYS_DATE CHAR(6);
17     1      DCL C_YEAR CHAR(2) DEFINED TODAYS_DATE POS(1);
18     1      DCL PAST_YEAR FIXED DEC(4);

19     1      TODAYS_DATE = DATE;
20     1      PAST_YEAR = 1900 + C_YEAR - 1;

21     1      OPEN FILE(UEMF) INPUT;
22     1      OPEN FILE(MFR) INPUT;
23     1      OPEN FILE(OUT) PAGESIZE(60) LINESIZE(132) OUTPUT;

24     1      ON ENDFILE(MFR) GO TO NEXT;
26     1      ON ENDFILE(UEMF) GO TO DCNE;

28     1      GET FILE(MFR) EDIT(NC_MFR)(COL(1), F(3));
29     1      ALLOCATE MFR_NAMES(0:NO_MFR);

30     1      ON ENDPAGE(OUT) BEGIN;
32     2      PUT FILE(OUT) PAGE;
33     2      PUT FILE(OUT) SKIP(2) EDIT('DIRECT OPERATING COSTS'
          (COL(56), A));
34     2      PUT FILE(OUT) SKIP(2) EDIT('COST PER MILE OR HOUR BY',
          'EQUIPMENT CLASS, MANUFACTURER, AND, AGE'
          (COL(34), A, A));
35     2      PUT FILE(OUT) SKIP(2) EDIT('MANUFACTURER', PAST_YEAR, 'LIFE'
          (COL(14), A, COL(35), F(4), COL(45), A));
36     2      PUT FILE(OUT) SKIP EDIT('AGE', 'CPM/CPH', 'CPM/CMH', 'NUMBER'
          (COL(23), A, COL(33), A, COL(43), A, COL(53), A));
37     2      END;
38     1      SIGNAL ENDPAGE(OUT);

          /*      READ IN A LIST OF THE MANUFACTURER NAMES.      */

39     1      HERE: GET FILE(MFR) EDIT(ARRAY_ELMT, NAMES)
          (COL(1), F(3), A(23));
40     1      MFR_NAMES(ARRAY_ELMT) = NAMES;
41     1      GO TO HERE;

          /*      READ A RECORD FROM THE MASTER FILE.      */

42     1      NEXT: READ FILE(UEMF) INTO (EQUIP_MSTR_FILE);
43     1      IF CLASS_CODE = 0
44     1      THEN GO TO NEXT;
45     1      PUT FILE(OUT) SKIP(2) EDIT('EQUIPMENT CLASS - ', CLASS_DESC)

```

START LEVEL NEST

```

      (COL(11),A,A(20));

46   1      LOOP: IF C_YEAR - YR_MFR <= 3
47   1          THEN I = 1;
48   1          ELSE IF C_YEAR - YR_MFR <= 6
49   1              THEN I = 2;
50   1          ELSE IF C_YEAR - YR_MFR <= 9
51   1              THEN I = 3;
52   1          ELSE IF C_YEAR - YR_MFR <= 12
53   1              THEN I = 4;
54   1          ELSE I = 5;

55   1      MI_HR_VEH(I,3) = MI_HR_VEH(I,3) + 1;
56   1      MI_HR_VEH(I,1) = MI_HR_VEH(I,1) + MILES_HRS_YEAR;
57   1      MI_HR_VEH(I,2) = MI_HR_VEH(I,2) + MILES_HRS_LIFE;

58   1      COSTS_YR_LIFE(I,1) = COSTS_YR_LIFE(I,1) + FUEL_COST_YEAR +
          LUBRICANTS_YEAR + TIRES_TUBES_YEAR + EXPEN_PARTS_YEAR +
          ANTIFREEZE_YEAR + PARTS_COST_YEAR + LABOR_COST_YEAR;
59   1      COSTS_YR_LIFE(I,2) = COSTS_YR_LIFE(I,2) + FUEL_COST_LIFE +
          LUBRICANTS_LIFE + TIRES_TUBES_LIFE + EXPEN_PARTS_LIFE +
          ANTIFREEZE_LIFE + PARTS_COST_LIFE + LABOR_COST_LIFE;

60   1      OLD_EQUIP_CLASS = CLASS_CODE;
61   1      OLD_MFR_CODE = MFR_CODE;

      /*      READ A RECORD FROM THE MASTER FILE.      */

62   1      READ FILE(UEMF) INTO (EQUIP_MSTR_FILE);

63   1      IF (OLD_MFR_CODE /= MFR_CODE) |
          (OLD_EQUIP_CLASS < CLASS_CODE)
64   1          THEN DO;
65   1              1      DO I = 1 TO 5;
66   1                  2      DO J = 1 TO 2;
67   1                      3      IF MI_HR_VEH(I,J) /= 0
68   1                          3      THEN TOTALS(I,J) = COSTS_YR_LIFE(I,J)/MI_HR_VEH(I,J);
69   1                          3      ELSE TOTALS(I,J) = 0;
70   1                  3      END;
71   1                  2      END;
72   1                  1      PUT FILE(OUT) SKIP(2) EDIT(MFR_NAMES(OLD_MFR_CODE))
          (COL(14),A);
          DO I = 1 TO 5;
73   1              1      DO I = 1 TO 5;
74   1                  2      IF MI_HR_VEH(I,3) > 0
75   1                      2      THEN PUT FILE(CUT) EDIT(AGE(I),
          (TOTALS(I,J) DC J = 1 TO 2), MI_HR_VEH(I,3))
          (COL(23),A,COL(34),F(6,3),COL(44),F(6,3),COL(52),F(4));
          ELSE;
76   1              2      END;
77   1              2      END;
78   1              1      MI_HR_VEH = 0;
79   1              1      COSTS_YR_LIFE = 0;
80   1              1      END;

81   1      IF OLD_EQUIP_CLASS < CLASS_CODE
82   1      THEN DO;

```

STMT LEVEL NEST

```
83      1      1      PUT FILE(OUT) SKIP(2) EDIT('EQUIPMENT CLASS - ',
          CLASS_DESC1(COL(11),A,A(20)));
84      1      1      END;
85      1      1      GO TO LOOP;

86      1      DONE: DO I = 1 TO 5;
87      1      1      DO J = 1 TO 2;
88      1      2      IF MI_HR_VEH(I,J) = 0
89      1      2      THEN TOTALS(I,J) = COSTS_YR_LIFE(I,J)/MI_HR_VEH(I,J);
90      1      2      ELSE TOTALS(I,J) = 0;
91      1      2      END;
92      1      1      END;
93      1      PUT FILE(OUT) SKIP(2) EDIT(MFR_NAMES(OLD_MFR_CCDE))
          (COL(14),A);
94      1      DO I = 1 TO 5;
95      1      1      IF MI_HR_VEH(I,3) > 0
96      1      1      THEN PUT FILE(OUT) SKIP EDIT(AGE(I),
          (TOTALS(I,J) DC J = 1 TO 2),MI_HR_VEH(I,3))
          (COL(23),A,COL(34),F(6,3),COL(44),F(6,3),COL(53),F(4));
97      1      1      ELSE:
98      1      1      END;
99      1      1      END MFRAGE;
```


Listing of Program UPDATE

STMT LEVEL NEST

1

UMF: PROC OPTIONS(MAIN);

```

/*      PROGRAM UPDATE                                */
/*      THIS PROGRAM UPDATES THE EQUIPMENT MASTER FILE BY REMOVING */
/*      THE RECORDS OF EQUIPMENT DISPOSED OF IN THE PAST YEAR.    */
/*      THIS PROGRAM IS ONE OF EIGHT COMPUTER PROGRAMS WHICH FORM */
/*      THE "COMPUTER-BASED INFORMATION SYSTEM FOR COUNTY EQUIPMENT */
/*      COST RECORDS".                                           */
/*      WRITTEN BY                                             */
/*          SYSTEMS DIVISION                                    */
/*          COLLEGE OF ENGINEERING                             */
/*          THE UNIVERSITY OF IOWA                             */
/*          IOWA CITY, IOWA                                   */
/*          JULY 1975                                          */
/*      DECLARE THE STRUCTURE FOR THE EQUIPMENT COST RECORDS.  */

```

2

1

DECLARE OLDMST FILE RECORD SEQUENTIAL;

1 UPDATED_EQ_MSTR_FILE;

```

2 COUNTY_NUMBER          PIC'999',
2 DISTRICT_NUMBER       CHAR(2),
2 EQUIPMENT_NUMBER      CHAR(8),
2 CLASS_CODE            CHAR(2),
2 OTHER_INFO1           CHAR(98),
2 LIFE_M_H              PIC'999999',
2 FUEL_COST              PIC'9999999',
2 LUB_COST              PIC'999999',
2 TIRES_TUBES_COST      PIC'999999',
2 EXP_PARTS_COST        PIC'999999',
2 ANTIFREEZE_COST       PIC'99999',
2 PART_COST             PIC'9999999',
2 LABOR_COST            PIC'9999999',
2 INDIRECT_COST         PIC'9999999',
2 CLASS_DESCRIPTION     CHAR(20),
2 MAKE_MODEL_DESCRIPTION CHAR(14),
2 OTHER_INFO2           CHAR(46),
2 DATE_SOLD,
3 MONTH                 CHAR(2),
3 DAY                   CHAR(2),
3 YEAR                  CHAR(2),
2 DISPOSAL_METHOD       CHAR(1),
2 OTHER_INFO3           CHAR(43);

```

3

1

DECLARE LETTERS FIXED DEC(5);

4

1

DECLARE NFMST FILE RECORD SEQUENTIAL;

5

1

DECLARE TOYAL_DIRECT_COST PIC'999999999' INIT(0);

6

1

DECLARE CPM FIXED DEC(5,3);

7

1

DECLARE COUNTY_NAME(99) CHAR(13) VARYING

```

INIT('ADAIR','ADAMS','ALLAMAKEE','APPANOOSE','AUDUBON','BENTON',
'BLACK HAWK','BOONE','BREMER','BUCHANAN','BUENA VISTA','BUTLER',
'CALHOUN','CARROLL','CASS','CEDAR','CERRO GORDO','CHEROKEE',
'CHICKASAW','CLARKE','CLAY','CLAYTON','CLINTON','CRAWFORD','DALLAS',

```

STMT LEVEL NEST

```
'DAVIS', 'DECATUR', 'DELAWARE', 'DESMOINES', 'DICKINSON', 'DLRUQUE',
'EMMET', 'FAYETTE', 'FLOYD', 'FRANKLIN', 'FREMONT', 'GREENE', 'GRUNDY',
'GUTHRIE', 'HAMILTON', 'HANCOCK', 'HARDIN', 'HARRISON', 'HENRY',
'HOWARD', 'HUMBOLDT', 'IDA', 'IOWA', 'JACKSON', 'JASPER', 'JEFFERSON',
'JOHNSON', 'JONES', 'KEOKUK', 'KOSSUTH', 'LEE', 'LINN', 'LOUISA', 'LUCAS',
'LYON', 'MADISON', 'MAHASKA', 'MARION', 'MARSHALL', 'MILLS', 'MITCHELL',
'MONROE', 'MONROE', 'MONTGOMERY', 'MUSCATINE', 'O'BRIEN', 'OSCEOLA',
'PAGE', 'PALO ALTO', 'PLYMOUTH', 'POCAHONTAS', 'POLK', 'POTTAWATTAMIE',
'POWESHIEK', 'RINGGOLD', 'SAC', 'SCOTT', 'SHELBY', 'SIOUX', 'STORY',
'TAMA', 'TAYLOR', 'UNION', 'VAN BUREN', 'WAPELLA', 'WARREN',
'WASHINGTON', 'WAYNE', 'WEBSTER', 'WINNEBAGO', 'WINNESHIEK', 'WOODBURY',
'WORTH', 'WRIGHT');
```

```
8 1 DECLARE DISP_METH CHAR(6) VAR;
9 1 DECLARE NEW_CCUNTY PIC'999' INIT(0);

10 1 OPEN FILE(OLDMST) INPUT;
11 1 OPEN FILE(NEWMST) OUTPUT;

12 1 ON ENDFILE(OLDMST) GO TO DCNE;
```

STMT LEVEL NEST

```

/*      READ A RECORD FROM THE MASTER FILE, AND CHECK TO SEE IF IT  */
/*      IS FLAGGED FOR DELETION.                                     */
14      1      NEXT: READ FILE(OLD*ST) INTO (UPDATED_EQ_MSTR_FILE);

15      1      IF DISPOSAL_METHOD = ' '
16      1      THEN DO;
17      1      1      WRITE FILE(NEW*ST) FROM (UPDATED_EQ_MSTR_FILE);
18      1      1      GO TO NEXT;
19      1      1      END;

20      1      ELSE IF DISPOSAL_METHOD = 'T' THEN DISP METH = 'TRADED';
22      1      ELSE IF DISPOSAL_METHOD = 'J' THEN DISP METH = 'JUNKED';
24      1      ELSE DISP METH = 'SOLD';

25      1      IF NEW_COUNTY ^= COUNTY_NUMBER
26      1      THEN DO;
27      1      1      LETTERS = LENGTH(COUNTY_NAME(COUNTY_NUMBER));
28      1      1      PUT PAGE;
29      1      1      PUT SKIP EDIT(COUNTY_NAME(COUNTY_NUMBER), ' COUNTY'
30      1      1      (COL((125-LETTERS)/2),A,A);
31      1      1      PUT SKIP(2) EDIT('LIST OF EQUIPMENT DISPOSED OF IN 19',YEAR)
32      1      1      (COL(47),A,A);
33      1      1      PUT SKIP(2) EDIT('TOTAL TOTAL') (COL(30),A);
34      1      1      PUT SKIP EDIT(' EQUIPMENT MILES/ DIRECT ',
35      1      1      'DISPOSAL DISPOSAL', 'LIFE') (A,A,X(6),A);
36      1      1      PUT SKIP EDIT(' NUMBER MAKE & MODEL HOURS COSTS ',
37      1      1      ' DATE METHOD', 'CPM/CPH') (A,A,X(5),A);
38      1      1      PUT SKIP(1);
39      1      1      NEW_COUNTY = COUNTY_NUMBER;
40      1      1      END;

41      1      TOTAL_DIRECT_COST = FUEL_COST + LUB_COST + TIRES_TUBES_COST +
42      1      EXP_PARTS_COST + ANTIFREEZE_COST + PART_COST + LABOR_COST;
43      1      IF LIFE_M_H ^= 0
44      1      THEN CPM = TOTAL_DIRECT_COST/(100*LIFE_M_H);
45      1      ELSE CPM = 0;

46      1      PUT SKIP EDIT(EQUIPMENT_NUMBER,MAKE_MODEL_DESCRIPTION,LIFE_M_H,
47      1      TOTAL_DIRECT_COST,MONTH,'-',DAY,'-',YEAR,DISP_METH,CPM)
48      1      (COL(3),A,X(2),A,X(2),F(7,0),X(2),F(9,2,-2),X(3),A,A,A,A,A,
49      1      X(7),A,X(6),F(6,3));
50      1      GO TO NEXT;

51      1      DONE: END UMF;

```