MASTER

296
7/5/68 *per*
AEC RESEARCH AND DEVELOPMENT REPORT

DSED, A DATA SET EDIT PROGRAM
AND
THE ALL MIGHTY MOUSE SUBPROGRAM

AUTHOR:

Alfred A. Brooks

**UNION
CARBIDE**

UNION CARBIDE CORPORATION
NUCLEAR DIVISION

Operating the

OAK RIDGE GASEOUS DIFFUSION PLANT • OAK RIDGE NATIONAL LABORATORY
OAK RIDGE Y-12 PLANT • PADUCAH GASEOUS DIFFUSION PLANT

*For the Atomic Energy Commission
Under U.S. Government Contract W7405 eng 26*

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

Printed in the United States of America. Available from Clearinghouse for Federal
Scientific and Technical Information, National Bureau of Standards,
U.S. Department of Commerce, Springfield, Virginia 22151
Price: Printed Copy \$3.00; Microfiche \$0.65

— LEGAL NOTICE —

This report was prepared as an account of Government sponsored work. Neither the United States, nor the Commission, nor any person acting on behalf of the Commission:

- A. Makes any warranty or representation, expressed or implied, with respect to the accuracy, completeness, or usefulness of the information contained in this report, or that the use of any information, apparatus, method, or process disclosed in this report may not infringe privately owned rights; or
- B. Assumes any liabilities with respect to the use of, or for damages resulting from the use of any information, apparatus, method, or process disclosed in this report.

As used in the above, "person acting on behalf of the Commission" includes any employee or contractor of the Commission, or employee of such contractor, to the extent that such employee or contractor of the Commission, or employee of such contractor prepares, disseminates, or provides access to, any information pursuant to his employment or contract with the Commission, or his employment with such contractor.

Date of Issue: June 24, 1968

Report Number: K-1748

Subject Category: Mathematics

and Computers

DSED, A DATA SET EDIT PROGRAM
AND
THE ALL MIGHTY MOUSE SUBPROGRAM

Alfred A. Brooks

Computing Technology Center

Union Carbide Corporation,
Nuclear Division
Oak Ridge, Tennessee

Work performed at the Computing Technology Center and the Biology Division of the Oak Ridge National Laboratory operated by Union Carbide Corporation, Nuclear Division for the United States Atomic Energy Commission.

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

TABLE OF CONTENTS

	Page
Abstract	vii
Section I - DSEP, A Data Set Edit Program	1
A. General Description	1
B. Control and Request Cards	3
C. Input/Output Routine - ALLIO	9
D. Error Processing	10
E. Schematic Flow of Data Set Edit Program	15
F. Master Record Format	18
Section II - All Mighty Mouse Subprogram	21
A. Specifications for All Mighty Mouse Subprograms	21
B. Master Data Set Record Format	32
C. Error Screens and Error Processing	38
D. Pathology Dictionary and Dictionary Update Program	41
E. Schematic Flow of the All Mighty Mouse Subprograms	47
Section III - All Mighty Mouse Users Guide and Set-Up	
Instructions	49
A. General Description	49
B. Set-up Instructions	51
Section IV - Program Listings	58
A. Data Set Edit Program	59
B. All Mighty Mouse Subroutines	69
C. Dictionary Update Program	94
Section V - Sample Input Data	100

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

ACKNOWLEDGMENTS

The author wishes to acknowledge the support of this work by the Biology Division of the Oak Ridge National Laboratory, the helpful discussions of Mr. Alfred Hume and the assistance of Miss Jane Dunphey who assisted with the programming.

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

ABSTRACT

This report describes the Data Set Edit Program and the set of sub-routines used to process data for the All Mighty Mouse master tapes. The IBM 360 FORTRAN program is suitable for reducing and managing data which refers to an individual and which consists of several subtopics. The master data sets are constructed of self-describing records stored sequentially by the individual serial (identification) number. The records may be of variable content and variable structure as long as the basic underlying rational of the record format is retained.

The master data set is intended for use in a subsequent sequential, conditional search and analysis program*.

The program contains capabilities to update, correct, delete and erase records or subtopics as well as to provide lists of various types, pooled data sets and a transaction log of all activity.

The calling program is general and is made specific to any problem by the use of subroutines. The application to low level radiation data on mice is described. This report consists of the following sections:

- (1) a description of the Data Set Edit Program,
- (2) the specifications of the subprograms required to process data for the All Mighty Mouse master tapes, (including the pathology dictionary update program).
- (3) a User's Guide and Setup Instructions,

THIS PAGE
WAS INTENTIONALLY
LEFT BLANK

- (4) program lists of the main program and all subroutines including a skeletal input/output subprogram and,
- (5) sample input data.

*Hume, A and Brooks, A. A., SADS, Statistical Analysis of Data Subsets, KDP-3310, in publication.

SECTION I

DSEP, A DATA SET EDIT PROGRAM

A. General Description

The Data Set Edit Program is a nontopic-oriented Fortran program to be used in processing data for storage as data sets on magnetic tapes, disks, or data cells. The several subtopics to be processed and stored must be related solely by an identification number, and the records are stored in the increasing order of this identification number in the data set. The record format is binary, and the first two words are the record number and the number of words of information which follow and which consists of locator and data subfields. The first and last record of each data set are reserved for data set identification and a termination flag. The first record and last record are given some arbitrary low and high number, respectively, which cannot refer to actual data.

The only (except for deletion of record) unit of data which can be processed by the program is a subtopic and each record maintains a list of its subtopic locators which indicate the presence and location of the subtopic data in the record. The subtopics may consist of fixed-or variable-length strings of words. The value of the subtopic locators is determined by the program as each string of subtopic information is added to the record. The value of a locator is the location of the start of the field identified minus the location of the locator itself. Thus, the locator is relative to its own loca-

tion, and gives the location of the identified field. Any locators for subfields within the string of data for a subtopic must be supplied as part of the data string for that subtopic and be consistent with the foregoing description.

The program has the ability to perform the following services:

1. Update the master data set by subtopic from external data and the current data set and provide copy of the updated record.
2. Correct a subtopic in a record.
3. Delete a subtopic from a record.
4. Delete an entire topic or record.
5. Erase an entire topic or subtopic from all records between limits.
6. Pool records between numerical limits from two master data sets to a third data set.
7. Provide lists of records between limits by record, by subtopic or by locators, or provide a list of omitted records or subtopics.
8. Provide a transaction log of any data set editing or listing.
9. Start data sets upon request.

The program is made specific to any problem by supplying an input/output routine called ALLIØ which is capable of reducing the incoming raw data to the desired word string and writing out the data by subtopic in the format desired. Printed copy of update transactions is produced one data set record per page while the pagination of lists is controlled by ALLIØ. The main program calls ALLIØ for initialization and for cleanup if required. The skeletal form of ALLIØ and

its flow for output purposes must conform to a predetermined logic but the details may be variable. The control words are read into the main program as data on a control card, and the subtopic information to be processed is recognized by the presence of these words on request cards. The input routine needs to supply only the identification number, the length of the data string, and the data string itself.

The data set reference numbers used by the program are:

04	Transaction log
05	Input
06	Error messages - Sysout
11	Master data set
12	Master data set
13	Master data set
16	Output - Sysout

B. Control and Request Cards

Control Card

The control information is read into the Data Set Edit Program in the following format:

1.	Cols. 1-16	(4A4)	Program or topic title
2.	Cols. 17-22	(I6)	Lowest (first) record identification number
3.	Cols. 23-28	(I6)	Highest (flag) record identification number

4.	Cols. 29-34	(I6)	Control number passed to ALLIO on
initialization
5.	Cols. 35-36	(I2)	Number of subtopics (locators)
6.	Cols. 37-40	(A4)	Title of the collective topic (all
subtopics)
7.	Cols. 41-80	(8A4)	Titles of the subtopics comprising
total record.

A summary of the control information is printed in the transaction log and placed in the first record of the data set.

Request Cards

The cards which declare the type of request to be performed take the formats shown in the table on the following page.

	Request	Subject	Data Set Controls		Aux. Con.	Record Limits			
			(1)	(2)		Primary Lower	Primary Upper	Secondary Lower	Secondary Upper
Cols.	1-4	10-13	25-30	31-36	37-42	46-51	55-60	64-69	73-78
Format	A4	A4	I6	I6	I6	I6	I6	I6	I6
Request 1a)	STAR			d					
Request 1b)	STAR	TWØ	d ₁	d ₂					
Request 2a)	UPDA	SUBT	d ₁	d ₂	m				
Request 2b)	CØRR	SUBT							
Request 2c)	DELE	SUBJ							
Request 2d)	ERAS	SUBJ				l ₁	l ₂		
Request 3)	PØØL		d ₁	d ₂	d ₃	l ₁	l ₂	l ₃	l ₄
Request 4a)	LIST	SUBJ	r			l ₁	l ₂		
Request 4b)	LIST	LØCA	r			l ₁	l ₂		
Request 4c)	LIST	ØMIS	r		n	l ₁	l ₂		
Request 5)	END	REQU							

where

- (1) Upper case characters are reserved words (left adjusted)
 - (a) SUBT may take on any of the names of the subtopics
 - (b) SUBJ may take on any of the names of the subtopics or the entire topic.
- (2) Lower case letters are numbers (right adjusted)
 - (a) d's are data set identification numbers
 - (b) l's are record identification numbers and zeros are equivalent to the lowest or highest record numbers.
 - (c) m is 0 or nonzero.
 - (d) n is 0 or a subtopic index.
 - (e) r's are data set reference numbers.

RETURN TO: _____ DATE: _____

BLDG: RM: CHARGE :

[illegible]

Execution of the requests are described below:

1. The execution of a STAR request results in:
 - (a) A data set being initialized as data set reference number 12 and given the identification number, d.
 - (b) Data sets being initialized as data set reference numbers 11 and 12 and given the identification numbers d_1 and d_2 respectively.
2. The execution of an UPDA, CORR, DELE or ERAS request results in:
 - (a) The data set (reference number 11) identified as d_1 being updated by the subject information specified by SUBT and which follows the control card. A single data card bearing the highest (flag) record identification number in the columns appropriate to the subject data will terminate the request. The updated information will be placed in the data set (reference number 12) identified by d_2 . The reference numbers will be permuted $11 \rightarrow 13$, $12 \rightarrow 11$, $13 \rightarrow 12$, but the identification numbers remain associated with the data set. Failure to correctly specify a consistent set of controls or updating an existing subject will result in an error message. A copy of the information for the updated record will be transmitted to data set reference number 16 if m is nonzero.
 - (b) The information supplied will be used to correct an existing record and its specified subtopic material in a manner identical to an updating request. A nonexistent subtopic of a record cannot be corrected and an error message will result

for that record if this is attempted.

- (c) The existing subtopic information in the data set will be deleted for each record specified. Deletion of all subtopics will leave the record identification number and the null locator words. Deletion by title of the entire topic will eliminate the entire record. Record identification numbers for deletion are in an I6 format, one per card. An attempt to delete a nonexistent topic or subtopic will produce an error message.
 - (d) ERAS is the same as DELE except that all records from an ℓ_1 to ℓ_2 inclusive are processed and no record cards are needed. If ℓ_1 and ℓ_2 are zero all records are erased.
3. Execution of the POOL request will result in all records from data set reference number 11 between the limits on the identification number of ℓ_1 and ℓ_2 and all the records from data set reference number 13 between the limits ℓ_3 and ℓ_4 to be pooled and overwritten into data set reference number 12 with preference given to data set reference number 11 as a source. Omission of the limits will make them all-inclusive. The reference numbers of the data sets are not permuted. If the identification number, d_3 , is zero a copy of data set 11 is made as data set 12.
 4. (a) The execution of the LIST request will produce a list of the specified information, SUBJ, between the limits $[\ell_1, \ell_2]$ specified from the data set specified by reference number $[r]$. If no limits are given, the list is all-inclusive. If no reference number is given, reference number 11 is assumed.

(After an update, correct, delete or erase this is the most recent data set. After pooling the pooled data set is mounted as reference number 12.) The list is transmitted to data set reference number 16, and the data sets are not permuted.

- (b) The request to list LOCA will list only the record identification numbers, number of words and the locator words.
 - (c) The request to list OMIS will produce a list of records from which the subtopic n is omitted, where n is the index of the subtopic, n equal to zero gives omitted records.
5. The execution of the END request will cause a branch to ALLIO for cleanup and termination of the job. A log of the request transactions which were stored as a data set reference number 4 is transmitted to data set reference number 16.

An improperly stated request will result in the END request with end file, rewinding of the data sets, and recovery of data set reference number 4 (transaction log) if possible.

C. Input/Output Routine - ALLIO

The input/output routine must be written to conform with the needs of the specific data supplied for each subject, the required data reduction, and the output format required by the user for his subject material. In addition, it must conform to the requirements of the main program. A skeletal form of ALLIO which meets the logical requirements of the calling program is shown in Appendix 1. Simple input/output and data conversion statements are shown as an example. The record identification is transmitted as IDN and the number of

words in the data string for the subject is transmitted as NSTR(1) followed by the data string in the array NSTR (or STR). ALLIØ must be capable of receiving an initialization call and termination call even though these may not be required.

D. Error Processing

The philosophy of handling errors in the Data Set Edit Program has been to prevent all detectable errors from reaching the master data sets and responses to errors are generally conservative. Errors of several types can occur at several locations. The processing of errors in the ALLIØ and related routines is not described here since it is peculiar to the specific applications. The errors may classified by the results they produce as follows:

1. loss of control to the system
2. loss of access to specific I/Ø devices
3. non-recognition of request
4. errors in data set identification
5. errors in record sequencing

The following is a discussion of the above errors occurring in the data logger itself.

1. The loss of control to the system will be recognized by a system message (IHCXXI) or a dump depending upon the job control cards used. The IBM manual (C28-6031) on completion codes should be consulted for a complete discussion. The following is a list of the most frequent errors:

(a) IHC212I

Explanation: IBC -- An attempt has been made to read or write a record, under FORMAT control, that exceeds the buffer length.

(b) IHC213I

Explanation: IBC -- The input list in an I/Ø statement without the FORMAT specification is larger than the logical record.

(c) IHC214I

Explanation: IBC -- An attempt has been made to write more than 255 FORTRAN records (without a FORMAT specification) within one FORTRAN logical RECORD.

(d) IHC215I

Explanation: IBC -- An invalid character exists for the decimal input corresponding to an I, E, F, or D format code.

(e) IHC217I

Explanation: IBC -- An end of data set was sensed during a READ operation; that is, a program attempted to read beyond the data.

(f) IHC218I

Explanation: IBC -- A permanent input/output error has been encountered, or an attempt has been made to read or write with magnetic tape a record that is less than 18 bytes long.

(g) IHC219I

Explanation: FIOCS -- A data set is referred to in the load module, but no DD statement is supplied for it or a DD statement has an erroneous dd name.

The data set is given on end file mark and rewind. The update log is available as data set number four but must be recovered by a separate job. A program, called RECLOG, is supplied for this purpose. The subsequent use (except as follows) of a master data set terminated in this manner may result in an error (end file) due to the absence of the normal last (flag) record. The data can be recovered by a POOL request using this data set mounted as data set reference number 11.

2. The loss of access to a specific I/O device (i.e. the ERR=ddd or END=ddd during a READ statement) still permits the program to issue a message, properly close out data sets even if incomplete and spill the transaction log into the output stream prior to termination. These messages are given below:

(a) PARAMETER I/O ERROR (FATAL)

Explanation: Error while reading first data card.

Action: No transaction log entries have been made, no initialization of ALLIO occurred. All data sets left in their original status before mounting.

(b) REQUEST I/O ERROR (FATAL)

Explanation: Error or end of file while reading a REQUEST card.

Action: This acts as a normal END card and all data sets

are closed normally, job ends normally leaving this and subsequent requests unfilled.

(c) DELETE I/Ø ERROR IDNOS IDNH IDN

Explanation: Error while reading delete data card. IDNH and IDN were the previous and current (?) record serial numbers.

Action: CALL CLSØUT (IDNH), STØP 00 is issued after closing open files (see action of CLØSØUT velow).

(d) DATA SET REFERENCE NØ. ddd CLØSED ØUT AT ERRØR ØN SERNØ iii

Explanation: Program has experienced an error and transferred to CLSØUT.

Action: Data set indicated closed out at record number indicated. The transaction log will have been recovered.

3. The non-recognition of request words results in the following message:

(a) REQUEST WØRD ERRØR (FATAL)

Explanation: The first four characters including blanks of the first request word are not correct.

Action: Treated as an END request resulting in a normal termination leaving this and subsequent requests unfilled.

(b) ABØVE REQUEST NØT LEGITIMATE. ERRØR IS FATAL

Explanation: The second request word (i.e. four characters) are not a proper subject or subtopic title (see control card)

Action: Treated as END request same as above.

4. The errors in data set recognition usually result from improper data set mounting or request cards and produce the following messages.

(a) DATA SET DISCREPANCY (FATAL) ERROR CODE (etc.)

Explanation: The mounted data sets do not agree with those specified on the request card.

The remainder of the message is self-explanatory

Action: Treated as END request.

(b) IMPROPER DATA SET MOUNTED (etc.)

Explanation: Data set mounting error for POOL request.

This could be any one of three data sets and only the first error is detected.

Action: This request is disregarded and job is terminated by the normal route.

5. Errors in record sequencing or action give rise to the following messages:

(a) ERROR IN SEQUENCE OF RECORDS SERNO=ddd BEFORE SERNO=eee

Explanation: Error in sequence of record numbers for this request. If ddd = eee a duplicate record exists.

Action: No transaction for this record (i.e. second record if duplicate)

(b) PRIOR OR NO ENTRY FOR RECORD ddd

Explanation: Prior entry in master data set for an update request or no entry for a correct or delete request.

Action: Request ignored for this record.

E. Schematic Flow of Data Set Edit Program

The schematic flow diagram of the Data Set Edit Program is shown in Figure 1. Underlined, upper case words indicate the names of subprograms and non-underlined upper case words are entry points. Lower case words are names of functions performed in various portions of the programs. Arrows connecting subroutines represent calls by one subprogram to an other while arrows within blocks are loops within the subprograms.

The complete program consists of four subprograms described below:

- (1) MAIN - The calling program when entered performs certain initialization functions; such as, reading the semi-permanent control card and providing an initialization call to ALLIØ to which two numbers may be transmitted (i.e. the number of locators and an arbitrary control number from the control card). Following this a request card is read and the routine STCLØT is initialized to enable abnormal termination procedures to be followed. A branch then occurs to the appropriate region of MAIN to fulfill the request. An update, delete, correct or erase request is filled in a loop which may require calls to RDALLI (for update and correction information) and to WRALLØ (for listing updated records). The listing functions are carried out in a loop which may require access to WRALLØ. The starting and pooling requests do not require access to any routine except ITIME. As indicated in the figure the correct data set mounting is verified before all requests except lists and starts (which do not transmit

topic information to the master data sets). The error and end sections provide for abnormal and normal termination calling CLSØUT and RECLØG respectively.

- (2) STCLØT - This subroutine provides for the proper nonstandard closing of any open files (i.e. master data sets) in the advent that an error in machine operation or data requires and permits it. The entry, CLSØUT, provides for the non-standard closing when STCLØT has been initialized which is done with each request where required. The second entry, RECLOG, provides for a call to ALLIØ for any required termination procedures and for the recovery of the transaction log from FT04F001. RECLØG is also used for the same purpose after CLSØUT has been called.
- (3) ALLIØ - The subprogram, ALLIØ, is determined by the format of the user's data and by the format of listings desired by the user. The entry ALLIØ is called by MAIN for any desired initialization. The entry RDALLI is called to input raw data, reduce the data and to prepare the word strings to be returned to the MAIN program. The entry WRALLØ is called to effect the desired listing of the data supplied to or residing in a master data set. A call from RECLØG provides for any termination procedures required. The basic logic and flow of control in ALLIØ is fixed but the details are a function of the user's requirements.
- (4) ITIME - This subroutine is called frequently by MAIN to supply the data and time to the program.

SCHEMATIC FLOW FOR DATA SET EDIT PROGRAM

F. Master Record Format

The rationale of the format used for the records in the master data sets is that it should (1) permit variable length subtopics; (2) permit the addition of information to the end of a fixed length subfield without disturbing the access to the earlier portion of the field; (3) permit a self-description of the location of various subfields for the purpose of data management and reading; and, (4) permit the description of the location of data items or data lists to be nonvarying with changing subtopic content.

This was accomplished by providing number fields referred to as locator fields preceding each data subfield. The value of a locator is chosen such that when added to the address of the locator, the sum is the location of the first word of a succeeding subfield. Thus a series of locators chosen from successive locator subfields will eventually identify the location of a data subfield containing a variable number of data items the first of which should be the number of items in the list including itself. Since the first locator of a locator subfield "points" to the first word beyond the locator subfield, it is the number of locators and hence this field is indeed a variable subfield. In actual practice, if a data subfield is a fixed length, then the length item is not necessary but still may serve a useful purpose in subfield identification.

The evaluation of the location of a data subfield is given by the following iterative expression:

$$a_{j,i+1} = (a_{j,i} + d_{j,i}) + n(a_{j,i} + d_{j,i})$$

where $a_{j,i}$ = address of the start of the i^{th} locator field for the j^{th} data field

$d_{j,i}$ = displacement of the i^{th} locator for the j^{th} data subfield from $a_{j,i}$

$n(a_{j,i} + d_{j,i})$ = value of the i^{th} locator of the j^{th} subfield

$a_{j,0}$ is the origin of the primary or first locator subfield and is the same for all j

$a_{j,i+1}$ is the address of the start of the j^{th} data field to which must be added the displacement of the k^{th} data item, $d_{j,k}=k$

The calculation of a address becomes a succession of rapid fixed point additions and is quite economical of machine time.

Since the n 's are contained in the locator subfields of the record, the only description of the location of the data item required is the values of the $d_{j,i}$'s. These are invariant with the varying record content and with additions to the fixed-length subfields. They may be concisely expressed by an expression such as:

$$(d_0 A + d_1 A + d_3 A + d_4)$$

where d_1 is one of the required displacements

A signifies 'add the contents of the location indicated to the address of the location and proceed.'

The first subfield of any record, referred to as the identification field, contains the record identification number as the first item, the number of words remaining in the record as the second item,

followed by the primary locator subfield. This field is followed by appropriate fixed-length or varying-length locator and data subfields. The existence or nonexistence of data subfields can be indicated by nonzero or zero locators in a fixed-length locator subfield.

The primary locators are constructed by the DSEP main program and are stored in a fixed-length locator field and if zero indicate the absence of a subtopic (i.e. subfield). The remaining locator subfields must be constructed in ALLIØ in the data reduction process. The total length of the data subfield for a subtopic must be the first item of the subfield transmitted by ALLIØ to DSEP in order to properly construct the primary locator subfield. The relative origin of the primary locator field is arbitrary but is usually taken as the address of the record identification number (i.e., the first primary locator has a displacement of 2) if the record items reside in contiguous locations.

SECTION II

ALL MIGHTY MOUSE SUBPROGRAM

A. Specifications for All Mighty Mouse Subprograms

The following specifications are a supplement to the description for the Data Set Edit Program. They contain information specific to the ALLIO subprogram for the All Mighty Mouse data reduction and to subprograms called by it. Included are the following:

- (1) Primary control card format.
- (2) Input specifications for each subtopic.
- (3) Master tape record format for each subtopic.
- (4) Description of dictionary used by pathology.
- (5) Description of other auxiliary subprograms.

1. Primary Control Card Format

The primary control card which assigns the Data Set Edit Program parameters has the following format for the All Mighty Mouse Program:

<u>Col.</u>	<u>Contents</u>
1-16	ALL MIGHTY MØUSE
17-22	000000
23-28	999999
29-34	000015
35-36	08
37-40	MØUS
41-44	ANAM

<u>Col.</u>	<u>Contents</u>
45-48	PATH
49-52	HEMA

2. Data Input Specifications

i. Specifications for Anamnesis Information

The input for the Anamnesis data is two cards per mouse from the Deadi program. They must occur in increasing order of mouse serial number. All dates are numeric by month, day, and year. Missing data is blank or zero. The card format is described below:

Card 1

<u>Col.</u>	
1-6	Mouse serial number
7	Card number = 0
8-13	Death date
14	Death code, alphanumeric
15-33	Mouse name, alphanumeric
34-39	Birth date
40-42	First weight (in tenths of a gram), numeric
43-48	Date of first weight
49-51	Second weight
53-57	Date of second weight
58-60	Third weight
61-66	Date of third weight

Col.

67	Radiation rate, numeric
68-70	Radiation amount, numeric
71-76	Date of radiation

Card 2Col.

1-6	Mouse serial number, numeric
7	Card number = 1, numeric
8-9	First room number, numeric
10-15	Date of first room number
16-17	Second room number
18-23	Date of second room number
24-25	Third room number
26-31	Date of third room number
32-37	Cage number, numeric
38-43	Date of cage number
44-48	Expanded death code, alphanumeric
49-59	Not used, alphanumeric
60-62	Experiment number

The data on these cards is subject to an error screen described later and must conform to same. The experiment number should agree with those on other subtopics input cards for the same mouse.

ii. Specifications of Pathology Information

Information to be processed efficiently by a high-speed computer must be stated in a precise and arbitrary manner unless complex, time-consuming, free-form interpretations are made. If the information is to be interrogated, then some standard format needs to be used or the interrogation will be fruitless. The mouse autopsy subroutine requires that:

- (1) the words used to describe diagnoses be limited to those words defined in the dictionary,
- (2) the words be correctly spelled,
- (3) the words be correctly punched in the cards, and
- (4) the cards be placed in the correct sequence.

Data forms are provided to facilitate the proper transcription of information to punched cards and the program will reject many erroneous data entries; but, in the final analysis, the accuracy of the data stored on the master tape is the result of a conscientious effort on the part of all persons responsible for its transcription.

Format of Autopsy Cards

The cards used to describe the diagnoses are identified as follows:

1. Process Card

- a. Col. 1 Importance Code
- b. Col. 2-25 Process word (24 characters)

- c. Col. 26-49 First Modifier word (24 characters)
- d. Col. 50-73 Second Modifier word (24 characters)
- e. Col. 74-80 Card identification

Card identification is the last five digits of the mouse serial number followed by two digits in sequence from 00 to 99.

2. Site Card

- a. Col. 1-2 Must be blank
- b. Col. 3-26 Site word (24 characters)
- c. Col. 27-50 Site modifier word (24 characters)
- d. Col. 51-69 Cause word (19 characters)
- e. Col. 70-72 Numeric data (3 digits)
- f. Col. 73 Units
- g. Col. 74-80 Card identification

3. Process Modifier Card

- a. Col. 1-25 Must be blank
- b. Col. 26-80 Same use as process card

4. Site Modifier Card

- a. Col. 1-26 Must be blank
- b. Col. 27-50 Same use as site card
- c. Col. 51-73 Should be blank
- d. Col. 74-80 Card identification

The above cards are identified solely by the contents of Cols. 2, 3, 26, and 27; therefore, all words must start in

these columns. All alphanumeric fields are left-adjusted and all numeric fields (data) are right-adjusted.

The only words permitted are those defined in the dictionary for the use intended. As many as ten modifier words may be used to modify either processes or sites. Blanks are included in this count. The term 'word' as used above implies a word, words, or phrase up to 25 (19 for cause) alphanumeric characters in length, including blanks.

Order of Autopsy Cards

There are two lead cards which identify the mouse and provide other pertinent information. Their format is shown on the data sheet. These are followed by the actual autopsy information.

A single diagnosis (involving multiple sites) must consist of the following:

- a. A process card followed by zero or more process-modifier cards.
- b. One (or more) site cards followed by zero or more site-modifier cards.

The process, process-modifier, site, and site-modifier cards should be placed in that order. When several site and site-modifier cards follow one set of process and process-modifier cards, the diagnosis will be expanded into several diagnoses

See next page for continuation of Card 1 and Card 2

Card 3 etc.

Card 4 etc.

Last diagnosis followed by END AUTOPSY in Cols. 2-12

Cols. 74-78 = Ser. No. mod 100000 79-80 sequenced 00-99

*No. 1 - Underlying Cause of Death

No. 2 - Immediate Cause of Death

Card 1
(cont)

Card 2

		GROSS DIAGNOSIS:		HIST. NO.
GROSS No	MICRO Yes	GROSS	MICRO	
		General Nutrition		
50	51	Skin		
52	53	Mouth-Lips		
54	55	Teeth-Gums		
56	57	Tongue		
58	59	Breast		
60	61	Ly. Nd. - Superficial		
62	63	Ly. Nd. - Deep		
64	65	Salivary Glands		
66	67	Pericardium/Pleura		
68	69	Peritoneum		
70	71	Bone Marrow - Femur		
72	73	Bone Marrow - Sternum		
1	2	Thymus		
3	4	Thyroid		
5	6	Lung		
7	8	Heart		
9	10	Vessels		
11	12	Spleen		
13	14	Pancreas		
15	16	Liver		
17	18	Gall Bladder		
19	20	Esophagus		
21	22	Stomach		
23	24	Small Intestine		
25	26	Colon		
27	28	Cecum		
29	30	Anus		
31	32	Testis/Ovary		
33	34	Sem. Ves/Uterus		
35	36	Prostate/Vagina		
37	38	Preputial/Cervix		
39	40	External Genitalia		
41	42	Urinary Bladder		
43	44	Kidney		
45	46	Adrenal		
47	48	Bone		
49	50	Joints		
51	52	Cranium		
53	54	Brain-Mngs		
55	56	Pituitary		
57	58	Ear		
59	60	Eye		

having the same process and different sites. As many as fifty diagnoses may be designated for a complete autopsy. The last diagnosis for a mouse is followed by the word END AUTØPSY starting in column 2.

These data cards are subject to an error screen which is described later and they must conform to same. The experiment number should be the same as that on other subtopic input cards for the same mouse.

iii. Specifications for Hematology Information

The input data for hematology is keypunched one card per mouse. They must occur in increasing order of mouse serial number. The fields are all numeric except for the code for filed samples or missing data. The format is described below:

<u>Cols.</u>	
1-6	Mouse serial number
7	Not used
8-19	Red cell count -- four 3-digit numbers.
20-35	White cell count -- four sets of one-space and three-digit numbers.
36-50	Platelet counts -- five sets of one-space and two-digit numbers.
51-62	Hematocrit counts -- four sets of one-space and two-digit numbers.

63-77 Differential counts -- five sets of
one-space and two-digit numbers.

- (a) Lymphocytes
- (b) Monocytes
- (c) Mature neutrophiles
- (d) Immature neutrophiles
- (e) Eosinophiles

78-80 Experiment number

Missing data may be indicated by a blank or a -1 and a filed
sample may be indicated by a -2, a ++, or ampersands, &&.

All numbers, ++'s and &&'s are to be right-adjusted. The
experiment number should be the same as that on other sub-
topic input cards for the same mouse.

HEMATOLOGY DATA SHEET

RETURN TO: _____ DATE: _____

BLDG: RM: CHARGE:

RIGHT ADJUST ALL NUMBERS; USE 360 CHARACTERS

(Missing Data = -1, Sample Filed = -2)

DO NOT PUNCH DECIMAL POINTS

[illegible]

B. Master Data Set Record Format

The following description of the Master Data Set Record format serves not only to describe the words in the record by location, name and type but also to describe the locator sequence used to address the item. Each field is given a qualified numeric classification such as $d_0.d_1.d_2$. These numbers (d_i) except for 1. (i.e., the identification field) are the sequence of displacements that must be specified in order to locate the origin of the field. The item number is also the displacement required to locate the item indicated (or the start of a list). The addressing code for the origin of a field is given to the right of its name in parentheses, i.e., $(d_1A+d_2A+d_3A)$. The complete address of an item (or start of a list) is of the form $(d_1A+d_2A+d_3A + \text{item no.})$. In RDALLI, the subscript which refers to a specified item in the NSTR (or STR) arrays is one more than the item number.

It should be noted that the two successive lists of modifier codes in the diagnosis fields can be located by the following addressing codes:

- (1) process modifiers $(3A+2A+kA+7)$
- (2) site modifiers $(3A+2A+kA+7A)$

where k is the index of the diagnosis desired. This is due to the fact that the first word in a list gives its length including itself. It should also be noted that in at least one subsequent program using this record, that the first two items of the identification field are reversed in order but that this does not affect the remaining items or their addressing.

The description of the record follows.

Key for mode in which data is stored:

A = Alphanumeric; I = Fixed Point; F = Floating Point; B = Bit

1. Identification Field - Fixed (Address of mouse identification number)

<u>Item No.</u>	<u>Contents</u>	<u>Mode</u>
0	Mouse Serial Number	I
1	Number of words remaining in record	I
2	Anamnesis	I
3	Pathology Field Locator	I
4	Hematology Field Locator	I
5-9	Unassigned Field Locators	I

2. Anamnesis Field - Fixed (2A)

<u>Item No.</u>	<u>Contents</u>	<u>Mode</u>
0	Number of words in Field	I
1	Death Date	I
2	Death Code	A1
3	Room Digit, Mouse Strain	A2
4	Foundation Stock Generation Number	A3
5	Foundation Stock Pair Number	A2
6	Grandmother's Pair Number	A2
7	Mother's Pair Number	A2
8	Animal's Pair Number	A2
9	Offspring Pair Number	A2

Anamnesis Field - Fixed (Continued)

<u>Item No.</u>	<u>Contents</u>	<u>Mode</u>
10	Litter Order Number	A1
11	Sex	A1
12	Individual's Litter Number	I
13	Birth Date	I
14	Weight 1 (tenths of grams)	I
15	Date of Item 14 (yr, mo, da)	I
16	Weight 2	I
17	Date of Item 16	I
18	Weight 3	I
19	Date of Item 18	I
20	Radiation Rate	I
21	Radiation Amount	I
22	Radiation Date	I
23	Room 1	I
24	Date of Item 23	I
25	Room 2	I
26	Date of Item 25	I
27	Room 3	I
28	Date of Item 27	I
29	Cage Number	I
30	Date Placed in Cage	I
31	Expanded Death Code	A4
32	Expanded Death Code Continued	A1

Anamnesis Field - Fixed (Continued)

<u>Item No.</u>	<u>Contents</u>	<u>Mode</u>
33	Not Used	A
34	Not Used	A
35	Age at Death	I
36	Age at Radiation Date	I
37	Age at Weight 1	I
38	Age at Weight 2	I
39	Age at Weight 3	I
40	Age at Cage Date	I
41	Age at Room 1 Date	I
42	Age at Room 2 Date	I
43	Age at Room 3 Date	I
44	Experiment Number	I

3. Pathology Field - Variable (3A)Locator Field (3A)

<u>Item No.</u>	<u>Contents</u>	<u>Mode</u>
0	Number of Words in Pathology Field	I
1	Auxiliary Data Field Locator	I
2	Autopsy Field Locator	I

3.1 Auxiliary Data Field - Fixed (3A + 1A)

<u>Item No.</u>	<u>Contents</u>	<u>Mode</u>
0	Experiment Number	I
1	Prosector	A3
2	Date of Autopsy	I

Auxiliary Data Field - Fixed (Continued)

<u>Item No.</u>	<u>Contents</u>	<u>Mode</u>
3	Pathologist	A3
4	(Fate of Carcass (4 bits) 1 = Yes) (Gross examination (42 bits) 1 = No) (Micro examination (42 bits) 1 = Yes)	B32-B3
5		B32-B3
6		B32-B5

3.2 Autopsy Field - Variable (3A + 2A)Locator Field (3A + 2A)

<u>Item No.</u>	<u>Contents</u>	<u>Mode</u>
0	Number of Words in Autopsy Field	I
1 to k	Diagnosis Locations	I

3.2k Diagnosis Fields - Variable (3A + 2A + kA)

<u>Item No.</u>	<u>Contents</u>	<u>Mode</u>
0	Number of Words in Diagnosis	I
1	Process Code	I
2	Site Code	I
3	Cause Code	I
4	Importance Code	I
5	Data	I
6	Unit Code	A1
7	Number of Process Modifiers + 1	I
8 to m	Process Modifier Codes	I
m+1	Number of Site Modifiers + 1	I
m+2 to n	Site Modifier Codes	I

4. Hematology Field - Fixed (4A)

<u>Item No.</u>	<u>Contents</u>	<u>Mode</u>
0	Number of Words in Hematology Field	I
1	Mean Red Count	F
2	Mean White Count	F
3	Mean Platelet Count	F
4	Mean Hematocrit Count	F
5	Not Used	
6-9	Number of Data Points for 2-5	F
10	Not Used	
11-14	Standard Deviations for 2-5	F
15	Not Used	
16-19	Red Count Raw Data	F
20-23	White Count Raw Data	F
24-28	Platelet Count Raw Data	F
29-32	Hemocrit Raw Data	F
33	Lymphocytes	F
34	Monocytes Percent	F
35	Mature Neutrophiles Percent	F
36	Immature Neutrophiles Percent	F
37	Eosinophiles Percent	F
38	Experiment Number	I

C. Error Screens and Error Processing

Error Screens

- a. Dictionary discrepancies
- b. Anamnesis Data

Rejection conditions

- 1. Birth date or death date equal zero
- 2. Birth date prior to 1-1-64
 Death date prior to birth date
 Death date after present date
- 3. Birth date < nonzero dates < death date
- 4. 20 < nonzero weight < 999
- 5. Weight = 0 and weight date $\neq$ 0
 or Weight $\neq$ 0 and weight date = 0
- 6. Radiation rate = 0 ($\neq$ 0) and radiation date $\neq$ 0 (=0)
- 7. First room number = 0 or first room date = 0
- 8. Radiation date $\neq$ 0 and second room number = 0
- 9. Second (third) room number = 0 ($\neq$ 0) and second (third)
 room date $\neq$ 0 (=0)
- 10. Cage number = 0 or cage date = 0
- 11. 0 > age at death > 1200
- c. Pathology
 - 1. Dictionary omissions
 - 2. Card sequence
 - 3. Process-site sequence
 - 4. Unrecognizable data cards

d. Hematology

1. None

Error Processing

The philosophy of error handling for the All Mighty Mouse program is to prevent all detected errors from reaching the master data sets.

The Data Set Edit Program section should be read for errors detected in that portion of the program or for errors which cause a loss of control to the system, and for the function of CLSOUT. The discussion below refers only to errors detected in the subprograms specific to the All Mighty Mouse data reduction and which issue messages from these subprograms:

1. The messages from errors which result in lack of access to specific I/O devices are given below:

(a) ANAMNESIS I/Ø ERRØR FØRMAT ddd SERNØ eee fff

Explanation: Error in reading an Anamnesis data card. The format number ddd identifies the READ statement and the SERNØ eee fff indicate the record (or adjacent record) involved.

Action: CALL CLSØUT (MØSD(1)); STØP 00

(b) PATHØLØGY I/Ø ERRØR FØRMAT ddd SERNØ eee fff

See (a).

(c) HEMATØLØGY I/Ø ERRØR FØRMAT ddd SERNØ eee

See (a).

2. The messages from errors which are detected in error screens are given below:

- (a) DICTIONARY ERROR THE FORWARD DICTIONARY IS TOO LONG OR
TOO SHORT MRLIS = ddd

Explanation: The pathology forward dictionary was too long
or short.

Action: CALL RECLDG. No changes have been made in mounted
data sets.

- (b) DICTIONARY ERROR THE REVERSE DICTIONARY IS TOO LONG OR TOO
SHORT MRLIS = ddd

Same as (a) for reverse dictionary.

- (c) DICTIONARY ERROR DUAL OR DISORDERED ENTRY ENTRY (CODE)
I = ddd

Explanation: Dual or disordered forward (or reverse)
dictionary, ddd gives index of word in error

Action: See (a).

- (d) ANAMNESIS CARD ERRORS SERNOS ddd eee

Explanation: Different mouse numbers on the two cards (see
ddd eee).

Action: No transaction for mouse.

- (e) ANAMNESIS DATA ERROR ddd

Explanation: The error screen has detected an error on
mouse ddd.

Action: No transaction for mouse ddd.

- (f) PATHOLOGY DATA ERROR ddd

Explanation: Card sequence error or word not located in

(f) Explanation (Continued)

dictionary (see dictionary omission list at end of transaction log).

Action: No transaction for mouse ddd.

(g) UNRECOGNIZABLE CARD REJECTED FOR MOUSE dddd

Explanation: Pathology data card contained blanks in columns 1, 2, 26 and 27.

Action: Card ignored, no transaction.

D. Pathology Dictionary and Dictionary Update Program

1. Pathology Dictionary

The pathology dictionary contains about 2000 entries of pathological terms and their synonyms (or equivalents) arranged in alphanumeric order and defined for one or more of the four uses; process, cause, site and modifier. They provide (in the forward dictionary) the means of translating pathology terms into numeric codes for the sake of economy. In its reverse form of about 2000 words, the numeric codes can be translated into pathology terms using the preferred form for the synonyms.

The dictionary is created and maintained by an updating program which maintains the codes of an original dictionary inviolate on updating. The dictionary exists as a non-formatted data set on magnetic tape, disk or data cell.

2. Dictionary Update Program

The program is intended to service lexicons whose purpose is to

2. Dictionary Update Program (Continued)

provide for the translation (encoding and decoding) of alphanumeric text to the single word numeric codes and vice versa. The entries in the lexicon can have from one to four different numeric codes for different uses. The entries can be classified as primary entries, synonym entries and compound entries. A primary entry is the preferred text associated with a numeric code and is the only decoding translation possible. A synonym entry is any non-preferred text (usually an abbreviation) to be associated with a numeric code (or rather the numeric code for a primary entry in its appropriate use is associated with a synonym entry on encoding). The four different uses for a synonym may refer to different primary entries. The compound entry is a special form of the synonym entry intended for use in expanding the reference to a single use into a multiple use encoding. A control digit in the forward dictionary with non-zero element indicates that an entry is a compound entry.

The forward dictionary contains all three types of entries sorted to alphanumeric order with up to four numeric codes for each. These numeric codes are positive for primary entries and must be set to non-overlapping ranges for each use. The synonym entries and compound entries contain the negative numeric code for the primary entries to which they are synonymous. The encoding process uses the absolute value of the numeric code.

2. Dictionary Update Program (Continued)

The reverse dictionary for decoding contains only the ordered positive numeric codes and hence refers only to the preferred text.

The dictionary update program is written for the IBM 360 computer in Fortran. The program has the capability of:

1. Starting an original dictionary and setting the machine code ranges
2. Adding dictionary entries - two uses/card
3. Deleting dictionary entries - two uses/card (Deletion of all uses causes the deletion of the entry.)
4. Defining synonyms - two uses/card
5. Defining compound entries - two uses/card
6. Sorting the lexicon into alphanumeric order by entry and numeric order by code
7. Producing data set on a magnetic tape, disk or data cell containing the complete, updated, sorted lexicon and the inverse sorted lexicon which does not contain synonyms
8. Producing a lexicon master list sorted by entry showing all codes and compound words
9. Receiving a lexicon for updating from either cards, magnetic tape, disk or data cell.
10. Deleting blanks from the lexicon.

2. Dictionary Update Program (Continued)

The restrictions on the use of the program are:

1. The lexicon must be sorted to carry out the DElete, SYNonym, or CØM-pound instructions and to add a second reference to a prior entry. This may be carried out by the insertion of the word SØRT into the update stream in the instruction field.
2. The array dimensions are all 2100.
3. The last entry must be the flag, LASTWØRD, which is a reserved word.

The records containing the forward and reverse dictionaries are described below:

Forward Dictionary - one unformatted record :

<u>Word</u>	<u>Contents</u>
1	number of words in dictionary = n
2-3	date of last update R*8
4	time of last update
*5-10	alphanumeric text
11	type code
12-15	numeric codes for uses
16	repeat from * n times
Flag word	'LAST'
Flag word	'WØRD'

Reverse Dictionary - one unformatted record

<u>Word</u>	<u>Contents</u>
1	number of words in dictionary = m
*2	numeric code
3	pointer to entry in forward dictionary
4	repeat from * m times
Flag word	-99999999

The format of the update and control cards is shown on the attached data sheet.

Description of Data Set-up

A. Control Card (I6, 4(1X,A8))

- (a) Logical unit no. = 0 starting dictionary supplied by program

= n old dictionary taken from data set

reference number n

N.B. Job control cards must be changed

- (b) Column (4) headings for word use

- B. Word entry cards or control cards in any order are subject to the restriction that the dictionary must be sorted with respect to the entry word in order to process an entry which is a prior entry and/or sorted with respect to the main entry in order to process a synonym.

C. Flag Card - (2A4) - LASTWORD

A listing of an input data deck is attached.

PATHOLOGY DICTIONARY DATA SHEET

Request No. _____ Prog. _____

(029 PUNCH)

CON = ADD, DEL, SYN, SØR

USE = 1, 2, 3, 4 (Process, Site, Cause, Modifier)

SYNONYMOUS TO:

[illegible]

E. Schematic Flow of the All Mighty Mouse Subprograms

The flow of logic and control in MAIN and STCLØT is identical to that described in Section I. For convenience the ALLIØ subprogram (see Figure II) has been split into three subprograms having the same general logic. The initialization of RDALL and WRALL are provided by calls from ALLIØ and the entries RDALLI and WRALLØ serve the purposes described earlier. The ALLIØ subprogram also initializes DICT at the same time. The subprogram DICT reads in the pathology dictionary from some external source. In processing pathology data, ALLIØ must have access to the dictionary which it obtains thru the entries NCØDE and DECØDE in DICT. At termination ALLIØ calls DICT to write out any dictionary omissions. The subprograms ISØRTS, ISEARS and ISEARM are sorting and searching routines used by ALLIØ and DICT. The subprograms CVBCBI, NPRES, YRMØDA and NDAGE are subprograms for the conversion of numeric symbols to binary numbers, fetching the current date, converting the form of the data and computing ages in days respectively.

SCHEMATIC FLOW FOR ALL MIGHTY MOUSE

SECTION III

ALL MIGHTY MOUSE USERS GUIDE AND SET UP INSTRUCTIONS

A. General Description

The All Mighty Mouse Program is a master tape updating program for biological information. The program uses three master tapes and should always process information from the most recent master tape to the oldest master tape holding the middle-aged master tape in reserve (even though it may be mounted for a second update). In addition to updating, correcting and deleting information on the most recent master tape, the program can be used to provide lists of various types and to pool information from two tapes onto a third tape. The program will automatically advance the tapes after each operation except a list or pool so that the most recent is ready to be used in the next operation. Lists are taken from the most recent tape unless otherwise specified.

In order to prevent the accidental use of a tape the request card for each operation which could modify a tape contains the identification numbers of the tapes to be used and verifies that they are mounted correctly. This requires that each tape be identified in the first record preferably with its own reel number. This, in turn, makes it necessary to start brand new master tapes by a separate starting step. The pooling operation requires three tapes to be used and hence all three tapes must be identified on the request card.

In updating or correcting a tape, the smallest unit of information is a subtopic which is identified by name (i.e., Anamnesis, Pathology and Hematology). These units are stacked together in increasing order of mouse number and processed between a request and end of data flag card, a single card having an identification or mouse number of 999999 in the field appropriate to the type of data being processed. Data of different types cannot be intermixed but several requests to process different types of data may be stacked provided that the request cards indicate the appropriate new tape identification numbers.

The delete request can refer to an entire record or topic which is identified by the word MOUSE in the request card. Otherwise, it is identical to the deletion of a subtopic (i.e., the smallest unit). In any case the data for a delete consists solely of the mouse number in columns 1-6. The flag card is 999999 in columns 1-6.

The pooling request differs from the previous requests in that it requires no subject data from cards and usually it will be preceded only by a start request. The program does not place the new pooled master tape in position for further update but leaves it mounted as data set reference number 12.

Lists of any type may be interspersed between any requests and they do not reposition the master tapes.

A transaction log which records each transaction and relates all tapes to their predecessors is produced. If lost due to machine errors it may be recovered by a stand-alone program from the tape containing it.

B. Set Up Instructions

1. Input Data

The input data is currently of three types described below:

a. Anamnesis - received from P. Leslie.

The data consists of two cards per mouse having a six digit mouse serial number in columns 1-6 and a card number (0 or 1) in column 7. They should be in numerical order when received. If necessary they can be sorted into order on columns 1-7. The flag card is 999999 in columns 1-6.

b. Pathology - received from J. Conklin.

The input data consists of not less than three cards per mouse. The mouse serial number is in the first card, columns 6-11, and the last 5 digits of this number are in columns 74-78 of each card with a sequence number, 00-99, in columns 79-80. The last card for each mouse has END AUTØPSY in columns 2-12. The cards should be in order but can be sorted on columns 74-80 if necessary. Note that the flag card for this subject is 999999 in columns 6-11.

c. Hematology - received from T. Odell.

The data consists of one card per mouse with the mouse serial

number in columns 1-6 and can be sorted on these columns if required. The flag card is 999999 in columns 1-6.

2. Master Tapes

The current master tapes are 6755, 24441 and 3212 and they are rotated in the following manner, each tape taking its turn in the sequence of master tapes:

The tapes when mounted are associated with the following data set reference numbers:

FT11F001	most recent master
FT12F001	oldest master
FT13F001	middle-aged master

It is necessary to change the last DD continuation card of each of these data sets in order to insure that the system requests the operators to mount the tapes correctly.

If a job does not involve FT12F001 or FT13F001, the DD cards may be removed to prevent the system requiring a tape mount.

3. Request Sheet

Normally the person submitting data will send a request sheet entitled "All Mighty Mouse Request" and will indicate on this sheet the service desired and from this sheet the required request card can be punched after determining the most recent and oldest master tapes from the last transaction log.

The following are instructions for punching the necessary request cards from the request sheet:

(All words left adjusted; all numbers right adjusted.)

i. Request for

Update, correct, or delete of Anamnesis, Pathology, Hematology or Mouse

- (a) Cols. 1-9 Punch the request word checked (i.e., Update, correct, or delete)
- (b) Cols. 10-24 Punch the subject word checked (i.e., Mouse, Anamnesis, Pathology, Hematology)
- (c) Cols. 25-30 Punch the identification number of the most recent master tape
- (d) Cols. 31-36 Punch the identification number of the oldest master
- (e) Col. 42 Punch a "1" if copy is requested

ii. Request for a list of subjects

- (a) Cols. 1-9 Punch the word "list"
- (b) Cols. 10-24 Punch one of the subject words (i.e., Mouse, Anamnesis, Pathology, Hematology, Locator, Omission)
- (c) Cols. 25-30 Leave blank if the most recent master tape or the master tape mounted on FT11F001 is desired as a source for the list.

Punch either "12" or "13" to obtain list of tapes referred to by those reference numbers. Blank, 0, and 11 are synonymous.

(d) Col. 42

If an omission list is requested, it can refer to records or to subtopics.

Punches in column 42 control this as follows:

Blank or 0 = Total Record

1 = Anamnesis

2 = Pathology

3 = Hematology

iii. Request to pool master tapes

(a) Cols. 1-6 Punch "pool"

(b) Cols. 25-30 Punch identification number of primary source tape

Cols. 31-36 Punch identification number of resulting new master tape

Cols. 37-42 Punch identification number of secondary source tape.

(c) Cols. 46-51 Primary source lower record limit

Cols. 55-60 Primary source upper record limit

Cols. 64-69 Secondary source lower record limit

Cols. 73-78 Secondary source upper record limit

Blank or zero limit corresponds to 0 and 999999.

Note: The new master tape produced will probably have been started just prior to the pooling operation by a start request.

iv. Start Request

The start request is used to provide a new tape to be used as a master tape. The tape contains no information except its identification record and flag record. The request card is punched as follows:

(a)	Cols. 1-9	Punch "Start"
(b)	Cols. 10-24	Punch "Two" is two tapes are desired
(c)	Cols. 25-30	Punch identification of first new tape
(on ref. no. 11)
	Cols. 31-36	Punch identification of second new tape
(on ref. no. 12)

If a single tape is desired, use only columns 31-36, tape will be on unit 12. It may be necessary to start two tapes, process a request to add data, and start a third tape when no master tapes are available.

v. End Request

The word END in columns 1-3 causes a normal ending of a stack of requests. The word REQUEST should be placed in columns 10-16 to distinguish this card from other END cards.

4. Deck Setup

i. Job Control Cards

- (a) Job Card - Change estimates for lines and time

- (b) FT11F001 { change last continuation card
- FT12F001 { to specify the appropriate
- FT13F001 { master tape

ii. Following FT05F001 DD *

- (a) One semi-permanent data card
- (b) Stacked requests
 - (1) Request card
 - (2) Data cards if required
 - (3) Flag card if data cards present
- (c) End Request Card

iii. End of Job Step

- (a) /*

5. Disposition of Output

- (1) Save and file the HASP system log, the control card sheet, the last copy of transaction log and the HASP statistics on following pages.
- (2) Return cards, lists, and first copy of transaction log as requested.

6. Non-Standard Termination

- (1) Recover transaction log from pool tape on FT04F001 using the RECLØG program.

7. Sample Forms Attached

- (1) All Mighty Mouse Request

ALL MIGHTY MOUSE REQUEST

Submitted by _____ Bldg. _____ Rm. _____ Phone _____

Return to _____ Bldg. _____ Rm. _____ Phone _____

(One request per sheet except lists[#])REQUEST (check one)
1 9SUBJECT (check one[#])
10 24

Printed	No
Copy	Copy
<u>Col 42=1</u>	<u>Col 42=0</u>

UPDATE _____

LOCATORS _____

CORRECT _____

OMISSIONS _____

DELETE _____

MOUSE _____

ANAMNESIS _____

POOL^{*,**} _____

PATHIOLOGY _____

LIST^{*#} _____

HEMATOLOGY _____

*Primary Source Record Limits 46 51 To 55 60 from Tape No. 25 30**Secondary Source Record Limits 64 69 To 73 78 from Tape No. 37 42

[#]Lists of several subjects using the same tape with same limits to be processed as separate requests may be specified on one request sheet. Omission lists can be further qualified as to subject information (i.e., Mouse, Anamnesis, Hematology or Pathology). Lists will be taken from the most recent master tape unless specified (see primary record limits) and blank primary record limits will be all inclusive.

N.B. All numbers are right adjusted and all words are left adjusted in the columns designated.

SECTION IV
PROGRAM LISTINGS

A. Data Set Edit Program

```

C      DATA SET EDIT PROGRAM (UPDATE,CORRECT,DELETE,ERASE,POOL,LIST,ETC)
C
      REAL*8 DATE,TDAT,TDAT
      INTEGER LOC(30),ILO(30)
      DIMENSION XLAB(4),XTYPE(30),XLAB7(4),XLAB8(4),LTAP(3),NRECAD(1000)
      1,STR(1000),NSTR(1000),TSTR(1000),TSTR7(1000),TSTR9(1000)
      EQUIVALENCE (TSTR7,LOC,TSTR),(TSTR9,STR,NSTR,ILOC)
      DATA STAR/'STAR'/,UPDA/'UPDA'/,DELE/'DELE'/,CORR/'CORR'/,
      1 TEND/'END '/,ERAS/'ERAS'/,STWO/'TWO '/,TLIST/'LIST'/,
      2 XLOC/'LOCA'/,POOL/'POOL'/,TOMI/'OMIS'/
      COMMON /BLOCK1/ITRAN,NRECAD
      99 FORMAT(' ',I9,A4,11I9)
      CALL ITIME(NTIM,NGTM,DATE,NAYR)
      FTIM=NTIM/360000.
      READ(5,1,ERR=9850)XLAB,NSTART,NLAST,LUNIT,NOLO,TREC,(XTYPE(I),I=1,
      1 NOLO)
      1 FORMAT(4A4,3I6,I2,11A4/20A4)
      WRITE (4,10) XLAB,DATE,FTIM,NSTART,NLAST,LUNIT,NOLO,TREC,
      1(I,XTYPE(I),I=1,NOLO)
      10 FORMAT('1TRANSACTION LOG FOR ',4A4,' PROGRAM'
      1/5X,'PROGRAM PARAMETERS USED ON ',A8,' AT ',F9.3
      2/8X,'FIRST AND LAST RECORD NUMBERS ARE ',I6,' AND ',I6
      3/8X,'ALL IO CONTROL NUMBER IS ',I6/8X,'NUMBER OF LOCATORS IS ',I6
      4/8X,'SUBJECT CODE IS ',A4/8X,'SUBTOPICS ARE (NO/CODE)'
      5/(8X,10(I5,2X,A4)))
      N7H=11
      N7=N7H
      N8H=12
      N8=N8H
      N9H=13
      N9=N9H
      NWUSF=NOLO+13
      NSTR(1)=LUNIT
      NSTR(2)=NOLO
      CALL ALLIO(0,0,STR,NSTR)
      100 CALL STCLOT(N8)
      COMMENT *****
      COMMENT READ REQUEST CARD
      READ(5,2,ERR=9860,END=9860)XJOB,XJOB2,TYPE,TYPE2,NOLD,NEW,NTHIRD,
      1N71,N72,N91,N92
      2 FORMAT(2A4,1X,2A4,7X,3I6,4(3X,I6))
      IF(XJOB.EQ.UPDA.OR.XJOB.EQ.CORR.OR.XJOB.EQ.DELE.OR.XJOB.EQ.ERAS)
      1 GO TO 101
      IF(XJOB.EQ.TLIST) GO TO 900
      IF(XJOB.EQ.POOL) GO TO 1100
      IF(XJOB.EQ.STAR) GO TO 1000
      IF(XJOB.EQ.TEND) GO TO 9900
      GO TO 9800
      COMMENT *****
      COMMENT START PROCESSING UPDATE,CORRECTION,DELETE,OR ERASE PROCEDURES
      101 WRITE(4,20)XJOB,XJOB2,TYPE,TYPE2,NOLD,NEW
      20 FORMAT('OREQUEST ',2A4,2X,2A4,' FROM DATA SET IDNO ',I6,
      1' TO DATA SET IDNO ',I6)
      IF(NTHIRD.NE.0) WRITE (4,25)
      25 FORMAT(' UPDATED RECORD REQUESTED')
      IF(N71.EQ.0) N71=NSTART
      IF(N72.EQ.0) N72=NLAST
      IF(XJOB.EQ.ERAS) WRITE(4,27) N71,N72
      27 FORMAT(' RECORD NUMBERS ',I7,' TO ',I7,' INCLUSIVE')
      NSTOP=0
      ITRAN=0
      DO 102 I=1,NOLO

```



```

102 ILOC(I)=0
    DO 105 I=1,NOLO
    IF(TYPE.EQ.XTYPE(I)) GO TO 106
105 CONTINUE
    I=NOLO
    IF(TYPE.NE.TREC) GO TO 9100
106 ITYP=I
    ILOC(ITYP)=NOLO+1-ITYP
COMMENT READ AND CHECK FIRST RECORDS ON OLD AND NEW DATA SETS
    REWIND N7
    REWIND N8
    READ(N7)IDN7,NWD7,XLAB7,NOLD7,NEW7,TDAT,FTIMT
    READ(N8)IDN8,NWD8,XLAB8,NOLD8,NEW8,TDAT,FTIMD
    IF(NEW7.NE.NOLD.OR.NEW8.NE.NEW) GO TO 9170
    DO 120 I=1,4
    IF(XLAB7(I).NE.XLAB(I).OR.XLAB8(I).NE.XLAB(I)) GO TO 9170
120 CONTINUE
200 REWIND N8
COMMENT START NEW DATA SET N8
    WRITE(4,13) NEW7,TDAT,FTIMT
13 FORMAT(' THE PREVIOUS DATA SET WITH IDNO ',I6,' WAS DATED ',A8,
1 ' AT ',F6.3,' HOURS')
    CALL ITIME(NTIM,NOTM,DATE,NAYR)
    FTIM=FLOAT(NTIM)/360000.
    WRITE(4,12)NEW,DATE,FTIM
12 FORMAT(' THE UPDATED DATA SET WITH IDNO ',I6,' IS REDATED ',A8,
1 ' AT ',F6.3,' HOURS')
    WRITE(N8) NSTART,NWDSF,XLAB,NEW7,NEW,DATE,FTIM,NLAST,LUNIT,
    INOLO,TREC,(XTYPE(I),I=1,NOLO)
COMMENT PROCESSING UPDATE
    IDN=NSTART
    IDNH=IDN
201 IF(IDN7.EQ.NLAST) GO TO 205
    READ(N7) IDN7,NWRD,(TSTR(I),I=1,NWRD)
205 IF(IDN.EQ.NLAST) GO TO 240
207 IF(XJOB.EQ.DELE.OR.XJOB.EQ.ERAS) GO TO 230
    CALL RDALL(I,ITYP,IDN,STR(NOLO+1),NSTR(NOLO+1))
    ISTR=NSTR(NOLO+1)
    GO TO 235
COMMENT READ A DELETE CARD
230 IF(XJOB.EQ.ERAS) GO TO 231
    READ(5,5,ERR=9870) IDN
    GO TO 234
231 IDN=IDN7
234 ISTR=0
    5 FORMAT(I6)
235 IF(IDN.GT.IDNH) GO TO 238
    WRITE(4,54) IDNH,IDN
54 FORMAT(' ERROR IN SEQUENCE OF RECORDS SERNO ',I6,' BEFORE SERNO
1 ',I6)
    GO TO 205
238 IDNH=IDN
COMMENT CHECK POINT CHARLIE
240 CONTINUE
    IF(IDN7.LT.IDN) GO TO 350
    IF(IDN7.GT.IDN) GO TO 360
    IF(IDN7.EQ.NLAST) GO TO 301
    IF (XJOB.EQ.ERAS.AND.(IDN.LT.N71.OR.IDN.GT.N72)) GO TO 350
265 IF((XJOB.EQ.DELE.OR.XJOB.EQ.ERAS).AND.TYPE.EQ.TREC) GO TO 370
COMMENT DELETION OR ERASURE IN TOTO
    IF(XJOB.EQ.UPDA.AND.LOC(ITYP).NE.0) GO TO 359

```

```

C
COMMENT SET UP CONTROLS TO MANIPULATE ARRAYS FOR UPDA,DELE,CORR AND ERAS
  IL=ITYP+1
  DO 271 I=IL,NOL0
    IF(LOC(I).NE.0) GO TO 275
271 CONTINUE
    INEX=NOL0
    GO TO 277
275 INEX=I
277 LOT=LOC(ITYP)+ITYP
    LON=LOC(INEX)+INEX
    IF(INEX.EQ.NOL0) GO TO 280
    LOND=LON
    GO TO 281
280 LOND=NWRD+1
281 IF(XJOB.EQ.UPDA) GO TO 282
    LOTD=LOT
    IF(LOC(ITYP).NE.0) GO TO 284
    GO TO 359
282 LOTD=LOND
284 IC=ISTR-(LOND-LOTD)
286 IF(INEX.EQ.NOL0) GO TO 292
    IF(IC.LT.0) GO TO 288
COMMENT SHIFT ARRAY RIGHT TO MAKE SPACE
    ID=NWRD+1
    DO 287 I=LOND,NWRD
      ID=ID-1
      IDD=ID+IC
287 TSTR(IDD)=TSTR(ID)
    GO TO 290
COMMENT SHIFT ARRAY LEFT TO MAKE SPACE
288 DO 289 I=LOND,NWRD
289 TSTR(I+IC)=TSTR(I)
290 LON=LOND+IC
    GO TO 294
292 LON=0
294 IF(XJOB.EQ.DELE.OR.XJOB.EQ.ERAS) GO TO 296
COMMENT ENTER NEW DATA INTO ARRAY
    IR=LOTD-1
    DO 295 I=1,ISTR
295 TSTR(I+IR)=STR(I+NOL0)
COMMENT RESET LOCATORS AND NUMBER OF WORDS
296 NWRD=NWRD+IC
    IF(XJOB.EQ.DELE.OR.XJOB.EQ.ERAS) GO TO 298
    LOT=LOTD
    GO TO 300
298 LOT=ITYP
300 LOC(ITYP)=LOT-ITYP
    IF(INEX.EQ.NOL0) GO TO 302
    LOC(INEX)=LON-INEX
    IL=INEX+1
    DO 304 I=IL,NOL0
      IF(LOC(I).NE.0) LOC(I)=LOC(I)+IC
304 CONTINUE
    GO TO 302
301 NWRD=1
302 WRITE(N8) IDN7,NWRD, (TSTR(I),I=1,NWRD)
    IF(IDN7.LT.NLAST) GO TO 370
COMMENT SPILL OUT TRANSACTION LOG
    GO TO 400
COMMENT WRITE RECORD FROM OLD DATA SET INTO NEW DATA SET
350 IF(IDN7.EQ.NSTART) GO TO 355

```

```

WRITE(N8) IDN7,NWRD,      (TSTR(I),I=1,NWRD)
355 READ(N7) IDN7,NWRD,    (TSTR(I),I=1,NWRD)
    IF(XJOB.EQ.ERAS) IDN=IDN7
    GO TO 240
360 IF(XJOB.NE.DELE.AND.XJOB.NE.CORR) GO TO 361
359 CONTINUE
    WRITE(4,23) IDN
    23 FORMAT('      PRIOR OR NO ENTRY FOR RECORD ',I6,' NO TRANSACTION')
    GO TO 205
COMMENT WRITE NEW RECORD INTO NEW DATA SET
361 KSTR=ISTR+NLO
    INLO=NLO+1
    WRITE(N8) IDN,KSTR,(ILOC(I),I=1,NLO),(STR(I),I=INLO,KSTR)
    NBR=2
    GO TO 371
370 NBR=1
371 IF(ITRAN.EQ.1000) GO TO 401
    ITRAN=ITRAN+1
373 NRECAD(ITRAN)=IDN
    IF(NTHIRD.EQ.0) GO TO(201,205),NBR
    WRITE(16,24) IDN,DATE,FTIM
    24 FORMAT('1UPDATED RECORD SERNO ',I6,' DATE ',A8,' TIME ',F9.3/' ')
    IF(NBR.EQ.2) GO TO 372
    CALL WRALLO(NLO+1,IDN,TSTR,LOC)
    GO TO 201
372 CALL WRALLO(NLO+1,IDN,STR,ILOC)
    GO TO 205
400 END FILE N8
401 IF(ITRAN.EQ.0) NRECAD(1)=0
    WRITE(4,26) ITRAN,(NRECAD(I),I=1,ITRAN)
    26 FORMAT(' NUMBER OF PROCESSED RECORDS ',I6 /' RECORD NUMBERS',15I7
    1/((1X,17I7)))
    IF(IDN7.GE.NLAST) GO TO 405
    ITRAN=1
    GO TO 373
405 NH=N7
    N7=N8
    N8=NH
COMMENT FOR TWO DATA SET SYSTEM N8=NH FOR THREE DATE SETS N8=N9
    N8=N9
    N9=NH
    GO TO 100
COMMENT *****
COMMENT DATA SET LISTING SECTION
900 IF(NOLD.EQ.0) NOLD=N7
    NL=NOLD
    REWIND NL
    IF(N71.EQ.0) N71=NSTART
    IF(N72.EQ.0) N72=NLAST
    READ(NL) IDN8,NWD8,XLAB8,NOLD8,NEW8,DATE,FTIMT
    WRITE(4,6) TYPE,TYPE2,NEW8,N71,N72,NL
    6 FORMAT('OREQUEST' LIST ',2A4,' FROM DATA SET IDNO ',I6,' RECORDS
    1',I6,' TO ',I6,' INCLUSIVE',I6,' REFNO= ',I3)
    WRITE(16,8) XLAB8,NEW8,DATE,FTIMT,NOLD8
    8 FORMAT('1LIST OF DATA SET ENTITLED ',4A4,' AND IDENTIFIED AS ',I6,
    1' DATED ',A8,' AT ',F9.3,' HOURS. PRIOR DATA SET WAS ',I6)
    IF(TYPE.EQ.XLOC) WRITE(16,7)
    7 FORMAT('O RECORD WORDS   SUBJECT LOCATORS ' )
    SUBJ=TREC
    IF(NTHIRD.NE.0) SUBJ=XTYPE(NTHIRD)
    IF(TYPE.EQ.TOMI) WRITE(16,90) SUBJ

```

```

90 FORMAT('0 RECORDS OMITTED FOR SUBTOPIC ',A4/)
   IDLP=NSTART
   NFLG=1
   DO 920 J=1,10000
   READ(NL) IDN,NWRD,(TSTR(I),I=1,NWRD)
   IF(IDN.GT.N72) GO TO 925
   IF(IDN.LT.N71) GO TO 920
   IF(TYPE.NE.XLOC) GO TO 902
   WRITE(16,9) IDN,NWRD,(LOC(I),I=1,NOL)
9   FORMAT(' ',2I6,1X,8I4)
   IF(IDN.EQ.NLAST) GO TO 925
   GO TO 920
902 IF(TYPE.NE.TOMI) GO TO 905
   IF(NFLG.EQ.0) GO TO 955
   IF(IDN.EQ.IDLP+1.AND.(NTHIRD.EQ.0.OR.LOC(NTHIRD).NE.0)) GO TO 975
   IDFM=IDLP+1
   NFLG=0
955 IF(NTHIRD.EQ.0.OR.LOC(NTHIRD).NE.0) GO TO 970
   IF(IDN.EQ.NLAST) GO TO 970
   IDLM=IDN
   GO TO 980
970 IDLM=IDN-1
   WRITE(16,91) IDFM,IDLM
91  FORMAT(2X,I7,' TO 'I7,' INCLUSIVE')
   NFLG=1
975 IDLP=IDN
980 IF(IDN.EQ.NLAST) GO TO 925
   GO TO 920
905 IF(IDN.EQ.NLAST) GO TO 925
   DO 915 I=1,NOL
   IF(TYPE.EQ.TREC.AND.LOC(I).NE.0) GO TO 910
   IF(TYPE.NE.XTYPE(I).OR.LOC(I).EQ.0) GO TO 915
910 CALL WRALLO(I,IDN,TSTR,LOC)
915 CONTINUE
920 CONTINUE
925 GO TO 100
COMMENT *****
COMMENT DATA SET INITIALIZING SECTION
COMMENT START BRAND NEW DATA SETS OR SETS
1000 NWORD=1
   REWIND N7
   REWIND N8
   CALL ITIME(NTIM,NOTM,DATE,NAYR)
   FTIM=FLOAT(NTIM)/360000.
   WRITE(4,14)N8,NEW
14  FORMAT('0DATA SET REFERENCE NUMRFR ',I6,' IDENTIFIED AS ',I6,' STA
   IRTED')
   WRITE(N8) NSTART,NWDSF,XLAB,NOLD,NEW,DATE,FTIM,NLAST,LUNIT,
   INOLO,TREC,(XTYPE(I),I=1,NOL)
   IF(TYPE.EQ.STWO) WRITE(N7) NSTART,NWDSF,XLAB,NOLD,NOLD,DATE,FTIM,
   INLAST,LUNIT,NOLO,TREC,(XTYPE(I),I=1,NOL)
   WRITE(N8)NLAST,NWORD,NWORD
   END FILE N8
   IF(TYPE.NE.STWO) GO TO 1010
   WRITE(4,14)N7,NOLD
   WRITE(N7)NLAST,NWORD,NWORD
   END FILE N7
1010 WRITE(4,32)DATE,FTIM
32  FORMAT(' NEW DATA SET IS DATED ',A8,' AT ',F9.3)
   GO TO 100
COMMENT *****
COMMENT POOL DATA SET SECTION

```

```

COMMENT CHECK DATA SET MOUNTING FOR POOLING
1100 LTAP(1)=NOLD
    LTAP(2)=NEW
    LTAP(3)=NTHIRD
    IF(N71.EQ.0) N71=NSTART
    IF(N91.EQ.0) N91=NSTART
    IF(N72.EQ.0) N72=NLAST
    IF(N92.EQ.0) N92=NLAST
    WRITE(4,31)N71,N72,NOLD,N91,N92,NTHIRD,NEW
31  FORMAT('OREQUEST POOL RECORDS ',I6,' TO ',I6,' FROM DATA SET IDNO
1',I6,'/10X,'AND RECORDS ',I6,' TO ',I6,' FROM DATA SET IDNO ',I6,
2' INTO DATA SET IDNO ',I6)
1110 DO 1130 I=1,3
    NTAP=LTAP(I)
    NL=N7H-1+I
    IF(NTAP.EQ.0) GO TO 1130
    REWIND NL
    READ(NL)IDN,NWD,XLAB7,NULL,NEW7,TDATE,FTIMT
    IF(IDN.NE.NSTART.OR.NEW7.NE.NTAP) GO TO 9135
    DO 1120 K=1,4
    IF(XLAB7(K).NE.XLAB(K)) GO TO 9135
1120 CONTINUE
    IF(I.EQ.2) GO TO 1130
    WRITE(4,13) NEW7,TDATE,FTIMT
1130 CONTINUE
    REWIND N8H
COMMENT START NEW DATA SET 12
    CALL ITIME(NTIM,NOTM,DATE,NAYR)
    FTIM=FLOAT(NTIM)/360000.
    WRITE(4,12)NEW,DATE,FTIM
    WRITE(N8H) NSTART,NWD,XLAB,NEW7,NEW,DATE,FTIM
1140 MT7=NSTART
    MT9=NSTART
    IF(NOLD.NE.0) GO TO 1145
    MT7=NLAST
    N71=NLAST
    N72=NLAST
1145 IF(NTHIRD.NE.0) GO TO 1150
    MT9=NLAST
    N91=NLAST
    N92=NLAST
1150 IF(MT7.EQ.NLAST) GO TO 1160
    READ(N7H,END=1155) MT7,NW7,(TSTR7(I),I=1,NW7)
    GO TO 1160
1155 MT7=NLAST
1160 IF(MT9.EQ.NLAST) GO TO 1170
    READ(N9H,END=1191)MT9,NW9,(TSTR9(I),I=1,NW9)
1170 IF(MT7.LT.MT9) GO TO 1175
    IF(MT7.GT.MT9) GO TO 1190
    IF(MT7.EQ.NLAST.AND.MT9.EQ.NLAST) GO TO 1195
    IF(MT7.GT.N72) GO TO 1176
    IF(MT9.GT.N92) GO TO 1191
    IF(MT7.LT.N71.AND.MT9.LT.N91) GO TO 1150
    WRITE(N8H) MT7,NW7,(TSTR7(I),I=1,NW7)
    GO TO 1150
1175 IF(MT7.LE.N72) GO TO 1177
1176 MT7=NLAST
    GO TO 1170
1177 IF(MT7.LT.N71.OR.MT7.EQ.NSTART) GO TO 1180
    WRITE(N8H)MT7,NW7,(TSTR7(I),I=1,NW7)
1180 READ(N7H,END=1176)MT7,NW7,(TSTR7(I),I=1,NW7)

```

```

      GO TO 1170
1190 IF(MT9.LE.N92) GO TO 1192
1191 MT9=NLAST
      GO TO 1170
1192 IF(MT9.LT.N91.OR.MT9.EQ.NSTART) GO TO 1160
      WRITE(N8H) MT9,NW9,(TSTR9(I),I=1,NW9)
      GO TO 1160
1195 NW1=1
      WRITE(N8H)NLAST,NW1,NW1
      END FILE N8H
      GO TO 100
COMMENT *****
COMMENT ERROR HANDLING SECTION
COMMENT *****
COMMENT ILLEGITIMATE REQUEST
9100 WRITE(4,21)
      21 FORMAT('OABOVE TOPIC NOT LEGITIMATE. ERROR IS FATAL')
      GO TO 9900
COMMENT *****
COMMENT IMPROPER DATA SET MOUNTED FOR POOL
9135 WRITE(4,41) NTAPE,NEW7,XLAB7
      41 FORMAT(' IMPROPER DATA SET MOUNTED(FATAL)'/ ' REQUESTED DATA SET I
      IDNO ',I6,' MOUNTED DATA SET IDNO ',I6,' ENTITLED ',4A4)
      GO TO 9900
COMMENT *****
COMMENT MOUNTING ERROR REWIND DATA SETS AND EXIT
9170 WRITE(4,11) NOLD,XLAB,NEW7,XLAB7,NEW,XLAB,NEW8,XLAB8
      11 FORMAT('ODATA SET DISCREPANCY (FATAL) ' /
      1' OOLD DATA SET REQUESTED WAS ',I6,' LABELED ',4A4,
      2' , DATA SET FOUND WAS ',I6,' LABELED ',4A4/
      3' ONEW DATA SET REQUESTED WAS ',I6,' LABELED ',4A4,
      4' , DATA SET FOUND WAS ',I6,' LABELED ',4A4)
      GO TO 9900
COMMENT *****
9800 WRITE(4,95)
      95 FORMAT('OREQUEST WORD ERROR (FATAL)')
      GO TO 9900
9850 WRITE(16,98)
      98 FORMAT('OPARAMETER I/O ERROR (FATAL)')
      STOP 00
9860 WRITE(4,97)
      97 FORMAT(' REQUEST I/O ERROR (FATAL)')
      GO TO 9900
9870 WRITE(4,96)IDNH,IDN
      96 FORMAT('ODELETE I/O ERROR SERNOS ',2I7)
      CALL CLSOUT(IDNH)
COMMENT STOP 00
9900 WRITE(4,35)
      35 FORMAT('OEND REQUESTS')
      CALL RECLOG
COMMENT STOP 00
      RETURN
      END

```

```

SUBROUTINE STCLOT(N8D)
C SUBROUTINE TO CLOSE OUT MASTER DATA SET N8
REAL*8 DATE
C
  DIMENSION REWRIT(33)
  DIMENSION NRECAD(1000)
  DATA TEND/'END ',TLIST/'LIST'/
  DATA NLAST/999999/,NWD/1/
  COMMON /BLOCK1/ITRAN,NRECAD
  N8=N8D
  RETURN
COMMENT *****
  ENTRY CLSOUT(IDNO)
COMMENT CLOSE OUT PARTIAL MASTER DUE TO ERROR
C
  WRITE(N8) NLAST,NWD,NWD
  END FILE N8
  IF(ITRAN.EQ.0) NRECAD(1)=0
  WRITE(4,26) ITRAN,(NRECAD(I),I=1,ITRAN)
26 FORMAT(' NUMBER OF PROCESSED RECORDS ',I6,' RECORD NUMBERS',15I7
1/(1X,17I7))
  WRITE(4,10) N8,IDNO
10 FORMAT(' DATA SET REFERENCE NO. ',I4,'CLOSED OUT AT ERROR ON SERNO
1 ',I7)
COMMENT *****
  ENTRY RECLOG
COMMENT START TRANSACTION LOG RECOVERY PROCEDURES
C
  CALL ALLIO(1,0,DUM,DUM)
  WRITE(4,99) TEND,TLIST
C RECOVER LOG OF TRANSACTIONS
DO 2010 I=1,2
  REWIND 4
2000 READ(4,99,END=2010) REWRIT
99 FORMAT(33A4)
  IF(REWRIT(1).EQ.TEND.AND.REWRIT(2).EQ.TLIST) GO TO 2010
  WRITE(16,99) REWRIT
  GO TO 2000
2010 CONTINUE
  CALL ITIME(NTIM,NOTM,DATE,NAYR)
  FTIM=NTIM/360000.
  WRITE(16,98) DATE,FTIM
98 FORMAT('OJOB COMPLETED ON ',A8,' AT ',F9.3)
  STOP 00
  END

```

```

C      SKELETAL FORM OF ALLIO TO BE USED WITH DATA SET EDIT PROGRAM
C
C      SUBROUTINE ALLIO(NSUB,IDN,STR,NSTR)
C
C      DIMENSION STR(1),NSTR(1)
C      EQUIVALENCE (STR,NSTR) IN CALLING PROGRAM
C      DIMENSION HOLL(9)
C      DATA HOLL/'THIS',' IS ','SUBJ','ECT ',' FOR',' REC','ORD ',' EUR',
1'EKA '/'
C      IF(NSUB.NE.0) GO TO 50
C      STR AND NSTR ARE TWO EQUIVALENCED NAMES FOR THE SAME ARRAY
C      IN WHICH DATA ARE TRANSMITTED
C      PERFORM ANY INITIALIZATION PROCEDURES HERE
C      NOLO IS THE NUMBER OF SUBJECT LOCATORS
C      LUNIT=NSTR(1)
C      NOLO=NSTR(2)
C      RETURN
C
C      50 CONTINUE
C      PERFORM ANY TERMINATION PROCEDURES HERE
C      RETURN
C      COMMENT *****
C      THIS ENTRY CALLED FOR ALL INPUT OF SUBJECT DATA
C      ENTRY RDALLI(NSUB,IDN,STR,NSTR)
C
C      GO TO(100,200,300,400),NSUB
C      100 CONTINUE
C      READ AND DATA REDUCTION FOR SUBJECT 1
C      REDUCED DATA IS PLACED IN STR (AND NSTR) ARRAY AS FOLLOWS
C      NSTR(1)=NUMBER OF WORDS PLACED IN ARRAY INCLUDING ITSELF
C      (STR(1),I=2,NSTR(1))=REDUCED DATA
C      READ(5,1)IDN
C      1 FORMAT(I6)
C      NSTR(1)=12
C      DO 110 I=2,5
C      110 STR(I)=HOLL(I-1)
C      NSTR(6)=NSUB
C      DO 120 I=7,9
C      120 STR(I)=HOLL(I-2)
C      NSTR(10)=IDN
C      STR(11)=HOLL(8)
C      STR(12)=HOLL(9)
C      RETURN
C
C      200 CONTINUE
C      SEE COMMENTS AT STATEMENT 100
C      READ(5,1)IDN
C      NSTR(1)=10
C      DO 210 I=2,5
C      210 STR(I)=HOLL(I-1)
C      NSTR(6)=NSUB
C      DO 220 I=7,9
C      220 STR(I)=HOLL(I-2)
C      NSTR(10)=IDN
C      RETURN
C      300 CONTINUE
C      REPLACE STOP WITH APPROPRIATE STATEMENTS FOR SUBJECT 3
C      STOP 3

```



```

      RETURN
400  CONTINUE
C    REPLACE STOP WITH APPROPRIATE STATEMENTS FOR SUBJECT 4
      STOP 4
      RETURN
C    EXTEND PATTERN FOR MORE THAN FOUR SUBJECTS
COMMENT *****
C
C    THIS ENTRY CALLED FOR ALL OUTPUT OF SUBJECT DATA
C
      ENTRY WRALLO(NSUB, IDN, STR, NSTR)
C
C    IDN IS THE RECORD IDENTIFICATION NUMBER
C    NSUB IS THE SUBJECT CONTROL NUMBER
      DO 9999 I=1, NOLO
      IF(NSUB.EQ.NOLO+1) GO TO 5000
      IF(NSUB.NE.I) GO TO 9999
5000 IF(NSTR(I).EQ.0) GO TO 9999
C    DATA STRING IS RETURNED FROM MAIN PROGRAM IN THE FORM
C    NSTR(1) THRU NSTR(NOLO) ARE SUBJECT LOCATORS (ZERO IF MISSING)
C    LOCATION OF NTH SUBJECT=NSTR(NTH)+NTH
C    NSTR(NSTR(NTH)+NTH)=NUMBER OF WORDS IN NTH SUBJECT
C    REMAINDER OF SUBJECT FOLLOWS AS CREATED IN DATA REDUCTION STEP
      LGSBI=NSTR(I)+1
      NWDI=NSTR(LGSBI)
      LUPSB=LGSBI+NWDI-1
      GO TO (5100,5200,5300,5400),I
5100 CONTINUE
C    PLACE STATEMENTS FOR OUTPUT OF SUBJECT 1 HERE
      WRITE(16,10) IDN, (NSTR(I), I=LGSBI, LUPSB)
      10 FORMAT(8HORECORD ,I7,8H WORDS ,2X,I3,2X,4A4,I3,3A4,I7,6A4)
      GO TO 9999
5200 CONTINUE
C    PLACE STATEMENTS FOR OUTPUT OF SUBJECT 2 HERE
      WRITE(16,10) IDN, (NSTR(I), I=LGSBI, LUPSB)
      GO TO 9999
5300 CONTINUE
C    PROVIDE STATEMENTS FOR WRITING OUT SUBJECT 3 HERE
      GO TO 9999
5400 CONTINUE
C    PROVIDE STATEMENTS FOR WRITING OUT SUBJECT 4 HERE
      GO TO 9999
C    EXTEND PATTERN FOR MORE THAN FOUR SUBJECTS
9999 CONTINUE
      RETURN
      END

```

B. All Mighty Mouse Subroutines

```

      SUBROUTINE ALLIC(NSUB,MCS,STR,NSTR)
      COMMENT ALLIC HAS BEEN SPLIT INTO THREE SUBROUTINES ( RDALL,WRALL )
      C
        DIMENSION STR(1000),NSTR(1000)
        IF(NSUB.NE.0) GO TO 17
      COMMENT INITIALIZE ALL SECTIONS
        LUNIT=NSTR(1)
        NOLC=NSTR(2)
        CALL RDALL(NOLC,LUNIT)
        CALL WRALL(NOLC,LUNIT)
        CALL DICT(LUNIT)
        RETURN
      COMMENT WRITE OUT DICTIONARY OMISSIONS
        17 CALL DICT(0)
      COMMENT STOP,00
        RETURN
      END
      SUBROUTINE RDALL(NOLC,LUNIT)
      C
      C
      REAL*4 MEANS,NPTS
      INTEGER PROCC,SITEC,CAUSC,PMDC,SMDC,PRCC,SITE,CAUS,PMD,SMD
      DIMENSION X(22),RR(5),SIGMA(5),SUM1(5),NPTS(5),MEANS(5)
      1,PRCC(50),SITE(50),CAUS(50),IMP(50),NDATA(50),UNIT(50),PMD(50,10)
      2,IPMT(50),ISMT(50),NSTR(1000),LSTR(51),MAST(96),LPTR(51)
      3,STR(1000),MASK(32),SMD(50,10)
      6      , PROCC(6),FMD1F(6),FMD2R(6),SITER(6),SMDR(6),CAUSR(6)
      7,CD(4),PMDC(10),SMDC(10),JL(2),JU(2),MOSD(6),MPROC(2)
      DATA BLANK/' '/,CAUS(6)/' '/,END/'END '/,JL/16,23/
      1,JU/20,31/,PLPL/' ++ '/,AMP/' && '/,MPROC/1000328,1000568/
      2,TWOM/' -2 '/,RMOM/' -1 '/
      NYMD(MDY)=10000*MOD(MDY,100)+100*(MDY/10000)+MOD(MDY,10000)/100
      COMMENT INITIALIZE ALL SECTIONS
      COMMENT SET LIMITS ON DATES
        NSTR=640000
        NPR=NPRES(0)
      18 MASK(1)=1
        DO 19 I=1,30
      19 MASK(I+1)=2*MASK(I)
        RETURN
      C
      ENTRY RDALL1(NSUB,MOS,STR,NSTR)
      GO TO (20,100,370),NSUB
      C
      COMMENT READ ANAMNESIS
      C
      20 READ(5,1,ERR=1026) MOSD(1),(NSTR(1),I=2,23)
      1 FORMAT(I6,1X,I6,A1,A2,A3,5A2,A1,A1,A2,I6,3(I3,I6),I1,I3,I6)
        MOS=MOSD(1)
        IF(MOSD(1).EQ.999999) RETURN
        READ(5,3,ERR=1027) MOSD(2),(NSTR(1),I=24,35),NSTR(45)
      3 FORMAT(I6,1X,3(I2,I6),2I6,4A4,I3)
      CHECK MOUSE NUMBERS ON CARDS
        IF(MOSD(1).NE.MOSD(2)) GO TO 1040
        NSTR(2)=NYMD(NSTR(2))
        NSTR(14)=NYMD(NSTR(14))
      COMMENT ANAMNESIS INPUT ERROR SCREEN
        NDE=NSTR(2)
        NBR=NSTR(14)
      CHECK BIRTH DATE AND DEATH DATE FOR ZERO
        IF(NBR.EQ.0.OR.NDE.EQ.0) GO TO 50

```

```

CHECK BIRTH DATE AND DEATH DATE FOR RANGE AND ORDER
  IF(NBR.LT.NSTR(1).OR.NDE.GT.NFR.OR.NDE.LT.NBR) GO TO 50
  DO 22 I=1,2
  JLI=JL(I)
  JUI=JU(I)
  DO 21 J=JLI,JUI,2
  NSTR(J)=NYMD(NSTR(J))
CHECK NONZERO DATES FOR RANGE
  IF(NSTR(J).NE.0.AND.(NSTR(J).LT.NBR.CP.NSTR(J).GT.NDE)) GO TO 50
  21 CONTINUE
  22 CONTINUE
CHECK WEIGHTS
  DO 23 I=15,19,2
CHECK WEIGHT RANGES
  IF(NSTR(I).NE.0.AND.(NSTR(I).LT.20.OR.NSTR(I).GT.999)) GO TO 50
CHECK WEIGHTS AND WEIGHT DATES
  IF(NSTR(I).EQ.0.AND.NSTR(I+1).NE.0) GO TO 50
  IF(NSTR(I).NE.0.AND.NSTR(I+1).EQ.0) GO TO 50
  23 CONTINUE
CHECK RADIATION RATE AND DATE
  IF(NSTR(21).EQ.0.AND.NSTR(23).NE.0) GO TO 50
  IF(NSTR(21).NE.0.AND.NSTR(23).EQ.0) GO TO 50
CHECK FIRST ROOM NUMBER AND DATE
  IF(NSTR(24).EQ.0.OR.NSTR(25).EQ.0) GO TO 50
CHECK RADIATION DATE AND SECOND ROOM NUMBER
  IF(NSTR(23).NE.0.AND.NSTR(26).EQ.0) GO TO 50
CHECK ROOM NUMBERS AND ROOM DATES
  DO 24 I=26,28,2
  IF(NSTR(I).EQ.0.AND.NSTR(I+1).NE.0) GO TO 50
  IF(NSTR(I).NE.0.AND.NSTR(I+1).EQ.0) GO TO 50
  24 CONTINUE
CHECK CAGE DATE AND NUMBER
  IF(NSTR(30).EQ.0.OR.NSTR(31).EQ.0) GO TO 50
  NSTR(36)=NDAGE(NSTR(14),NSTR(2))
CHECK AGE RANGE
  IF(NSTR(36).LT.0.OR.NSTR(36).GT.1200) GO TO 50
  NSTR(37)=NDAGE(NBR,NSTR(23))
  NSTR(38)=NDAGE(NBR,NSTR(16))
  NSTR(39)=NDAGE(NBR,NSTR(18))
  NSTR(40)=NDAGE(NBR,NSTR(20))
  NSTR(41)=NDAGE(NBR,NSTR(31))
  NSTR(42)=NDAGE(NBR,NSTR(25))
  NSTR(43)=NDAGE(NBR,NSTR(27))
  NSTR(44)=NDAGE(NBR,NSTR(29))
  NSTR(1)=45
  RETURN
C
COMMENT ERROR MESSAGES
  50 WRITE(4,91) MOS
  91 FORMAT('          ANAMNESIS DATA ERROR ',I7)
  GO TO 20
COMMENT *****
  1026 ICER=1
  GO TO 1028
  1027 ICER=2
  1028 WRITE(4,92) ICER,MOSD(1),MOSD(2)
  92 FORMAT('OANAMNESIS I/O ERROR FORMAT ',I3,' SERNOS ',4I7)
  CALL CLSQUT(MOSD(1))
COMMENT STOP 00
COMMENT *****
  1040 WRITE(4,93) MOSD(1),MOSD(2)
  93 FORMAT('          ANAMNESIS CARD ERROR SERNOS ',2I7)

```

```

      GC TO 20
COMMENT *****
COMMENT READ PATHOLCGY
100 NSTR(2)=2
    NSTR(3)=8
    NFL=0
    READ(5,4,ERR=1361)NSTR(4),MOS,STR(5),NSTR(6),STR(7)
1, (MAST(I),I=1,4),(MAST(I+4),MAST(I+46),I=1,12),IDCD,NCRD
4  FORMAT(1X,I4,I6,A3,I6 ,A3,4(I1,1X),18X,24I1,I5,I2)
    MOSH=MOS
    IF(MOSH.EQ.999999) GO TO 360
    IDCDH=MCD(MOSH,100000)
    IF(IDCD.NE.IDCDH.OR.NCRD.NE.0) NFL=1
    READ(5,5,ERR=1362) (MAST(I+16),MAST(I+58),I=1,30),IDCD,NCRD
5  FORMAT(60I1,13X,I5,I2)
    IF(IDCD.NE.IDCDH.OR.NCRD.NE.1) NFL=1
    J=0
    DO 110 I=1,2
    MASS=0
    DC 105 K=1,30
    J=J+1
    IF(MAST(J).EQ.1) MASS=MASS+MASS(K)
105 CONTINUE
    NSTR(I+7)=MASS
110 CONTINUE
    NSTR(6)=NYMD(NSTR(6))
    IP=0
    ISM=0
    IPM=0
    ISPL=0
    ISITE=0
    PROCC=0
    SITEC=0
    CAUSC=0
    IMPH=0
    PMDC(1)=0
    SMDC(1)=0
    DATAH=BLANK
    UNITH=BLANK
COMMENT READ AUTOPSIES UNTIL 'END'
201 NCRDH=MCD(NCRD+1,100)
    READ(5,6,ERR=1363)IMPR,PROCR,PMCLR,PMD2R,
1  SITER,SMCR,(CAUSR(I),I=1,5),DATAR,UNITR,
2CD,IDCD,NCRD
6  FORMAT(T1,I1,18A4,I1,2X, 16A4,A3,A3,A1,T1,1X,A1,A1,22X,A1,A1,46X
1,I5,I2)
    IF(IDCD.NE.IDCDH.OR.NCRD.NE.NCRDH) NFL=1
    IF(PCR(1).NE.END)GO TO 205
    NRET=3
    GO TO 250
COMMENT TRANSFER TO SPILL ON END CARD
C
COMMENT DETERMINE TYPE OF CARD
205 IF(CC(1).NE.BLANK) GO TO 210
    IF(CC(2).NE.BLANK) GO TO 220
    IF(CC(3).NE.BLANK) GO TO 230
    IF(CC(4).NE.BLANK) GO TO 240
COMMENT UNRECOGNIZABLE CARD REJECTED
    NFL=1
    WRITE(4,50) MOSH
90 FORMAT(' UNRECOGNIZABLE CARD REJECTED FOR MOUSE ',I7)
    GO TO 201

```

COMMENT RETURN TO READ NEW CARD

C

COMMENT CHECK TO SEE IF SPILL IS NECESSARY

210 IF(ISPL.LT.2) GO TO 211

ISPL=0

NRET=1

IF(ISITE.NE.1) NFL=1

GO TO 250

COMMENT TRANSFER TO SPILL ON NEW PROCESS CARD

C

COMMENT PROCESS DATA FROM LAST PROCESS CARD READ

211 ISPL=ISPL+1

IMPH=IMPR

CALL NCCDE(PROCR,1,MOS,PROCC)

COMMENT PROCESS DATA FOR PROCESS MODIFIERS

230 IPM=IPM+1

CALL NCCDE(PMD1R,4,MOS,PMDC(IPM))

IPM=IPM+1

CALL NCCDE(PMD2R,4,MOS,PMDC(IPM))

IF(PROCC.EQ.1.OR.PMDC(IPM).EQ.1.OR.PMDC(IPM-1).EQ.1) NFL=1

GO TO 201

COMMENT RETURN TO READ NEW CARD

C

COMMENT CHECK TO SEE IF SPILL IS NECESSARY

220 IF(ISPL.LT.2) GO TO 221

ISPL=1

NRET=2

IF(ISITE.NE.1) NFL=1

GO TO 250

COMMENT TRANSFER TO SPILL ON NEW SITE CARD

C

COMMENT PROCESS DATA FROM LAST SITE CARD READ

221 ISPL=ISPL+1

ISITE=1

CALL NCCDE(SITER,2,MOS,SITEC)

CALL NCCDE(CAUSR,3,MOS,CAUSC)

DATAM=DATAR

UNITH=UNITR

COMMENT PROCESS DATA FOR SITE MODIFIERS

240 ISM=ISM+1

CALL NCCDE(SMCR,4,MOS,SMDC(ISM))

IF(SITEC.EQ.1.OR.CAUSC.EQ.1.OR.SMDC(ISM).EQ.1) NFL=1

GO TO 201

COMMENT RETURN TO READ NEW CARD

C

COMMENT SPILL ALL PROCESSED DATA INTO ARRAYS

250 IP=IP+1

IMP(IP)=IMPH

PROC(IP)=PROCC

ID=0

DO 252 I=1,IPM

IF(PMDC(I).EQ.0) GO TO 252

ID=ID+1

PMD(IP,ID)=PMDC(I)

252 CONTINUE

IPMT(IP)=ID

SITE(IP)=SITEC

ID=0

DO 254 I=1,ISM

IF(SMDC(I).EQ.0) GO TO 254

ID=ID+1

SMD(IP,ID)=SMDC(I)

```

254 CONTINUE
  ISMT(IP)=ID
  CAUS(IP)=CAUSC
  CALL CVBCBI(DATAH,NEIN)
  NDATA(IP)=NEIN/10
  UNIT(IP)=UNITH
  ISM=0
  ISITE=0
  IF(NRET.NE.2) IPM=0
  GO TO (211,221,290),NRET
COMMENT RETURN TO PROCESS LAST CARD READ
C
290 IF(NFL.EQ.1) GO TO 1366
  NSUM=IP
  IPU=IP
  LSTR(1)=1
  IF (IPU.NE.1) CALL PSCRTM(4,IMP,1,IPU,LSTR)
  DO 295 I=1,IPU
    IPD=IPU+1-I
295  LPTR(I)=LSTR(IPD)
    ID=11+IP
    IF (IPU.EQ.1) GO TO 300
    JLH=1
    DO 296 I=1,2
      JB=JLH
      IF(JB.GT. IPU) GO TO 300
      DO 298 J=JB,IPU
        JC=LPTR(J)
        IF(PROC(JD).NE.MPROC(I)) GO TO 298
        IF(J.EQ.JLH) GO TO 297
        KI=JLH+1
        DO 296 K=KL,J
          KD=J+KL-K
296  LPTR(KD)=LPTR(KD-1)
        LPTR(JLH)=JC
297  JLH=JLH+1
298 CONTINUE
299 CONTINUE
300 DO 350 IPD=1,IPU
    IP=LPTR(IPD)
COMMENT COMPUTE LCCATORS
    IPDD=LPTR(IPD-1)
    IF(IPD.NE.1) NSUM=NSUM+8+IPMT(IPDD)+ISMT(IPDD)
    NSTR(IPD+11)=NSUM
    ID=ID+1
    NSTR(ID)=9+IPMT(IP)+ISMT(IP)
    ID=ID+1
    NSTR(ID)=PROC(IP)
    NSTR(ID+1)=SITE(IP)
    NSTR(ID+2)=CAUS(IP)
    NSTR(ID+3)=IMP(IP)
    NSTR(ID+4)=NDATA(IP)
    NSTR(ID+5)=UNIT(IP)
    NSTR(ID+6)=IPMT(IP)+1
    ID=ID+6
    IUP=IPMT(IP)
    IF(IUP.EQ.0) GO TO 331
    DO 320 IPM=1,IUP
320  LSTR(IPM)=PMD(IP,IPM)
    IF(IUP.NE.1) CALL ISORTS(4,LSTR,IUP)
321 DO 330 IPM=1,IUP
    ID=ID+1

```

```

      NSTR(ID)=LSTR(IPM)
330 CONTINUE
331 ID=ID+1
      NSTR(ID)=ISMT(IP)+1
      IUP=ISMT(IP)
      IF(IUP.EQ.0) GO TO 350
      DO 335 ISM=1,IUP
335  LSTR(ISM)=SMD(IP,ISM)
      IF(IUP.NE.1) CALL ISORTS(4,LSTR,IUP)
336 DO 340 ISM=1,IUP
      IC=IC+1
      NSTR(ID)=LSTR(ISM)
340 CONTINUE
350 CONTINUE
      NSTR(1)=ID
      NSTR(11)=ID-10
360 RETURN
COMMENT *****
1361 IOER=4
      GO TO 1370
1362 IOER=5
      GO TO 1370
1363 IOER=6
1370 WRITE(4,94)IOER,MCSH,MOS
      94 FORMAT('OPATHOLOGY I/C ERROR FORMAT ',I3,' SERNOS ',2I7)
      CALL CLSOUT(MOSH)
COMMENT STOP 00
COMMENT *****
1366 WRITE(4,96) MOSH
      96 FORMAT('          PATHOLOGY DATA ERROR ',I7)
      GO TO 100
COMMENT *****
C
C **** INPUT TO HEMATOLOGY
C
370 READ(5,371,ERR=3770)MCS,NCD,X,NSTR(39)
371 FORMAT(16,I1,4A3,4(1X,A3),14A3,13)
      MCSH=MOS
      IF(MOS.EQ.999999) RETURN
      DO 373 I=1,4
      SUM1(I)=0.
      SIGMA(I)=-1.
      MEANS(I)=-1.
373  NPTS(I)=0.
      MEANS(5)=BLANK
      SIGMA(5)=BLANK
      K=0
      DO 3733 I=1,5
      NN=0
      N=4
      IF(I.EQ.3.OR.I.EQ.5)N=5
      DO 3734 J=1,N
      K=K+1
      IF(X(K).NE.BLANK.AND.X(K).NE.8MG) GO TO 3712
      X(K)=-1.
      GO TO 3734
3712 IF(X(K).NE.PLPL.AND.X(K).NE.TWOM.AND.X(K).NE.AMP) GO TO 3713
      X(K)=-2.
      GO TO 3734
3713 NPTS(I)=NPTS(I)+1.
      CALL CVBCBI(X(K),IX)
      XX=IX/10.

```

```

      IF(I.EQ.1.OR.I.EQ.2) XX=IX/100.
      X(K)=XX
      IF(I.EQ.5) GO TO 3734
      NN=NPTS(I)
      RR(KK)=X(K)
      SUM1(I)=SUM1(I)+X(K)
3734 CCNTINUE
      IF(I.EQ.5) GO TO 3733
      IF(NN.GT.0) MEANS(I)=SUM1(I)/NPTS(I)
      IF(NN.LE.1) GO TO 3733
      DIF=0.
      DO 3731 KK=1,NN
3731 DIF =DIF+(RR(KK)-MEANS(I))**2
      SIGMA(I)=SQRT(DIF/(NPTS(I)-1.))
3733 CUNTINUE
      IF(MEANS(3).GE.0.) MEANS(2)=.025*MEANS(3)
      IF(SIGMA(3).GE.0.) SIGMA(3)=.025*SIGMA(3)
      ID1=1
      J=ID1
      K=0
      NSTR(ID1)=39
      M=ID1+15
      DO 3760 I=1,5
      J=J+1
      STR(J) = MEANS(I)
      STR(J+5)=NPTS(I)
      STR(J+10)=SIGMA(I)
      N=4
      IF(I.EQ.3.OR.I.EQ.5)N=5
      DO 3760 L=1,N
      K=K+1
      M=M+1
3760 STR(M)=X(K)
      RETURN
COMMENT *****
3770 IGER=371
      WRITE(4,95)IGER,MOSH
95  FORMAT('OHEMATCLOGY I/O ERRORR FORMAT ',I4,' SERNOS ',I7)
      CALL CLSQUT(MOSH)
COMMENT STGP 00
      RETURN
      END
      SUBROUTINE WRALL(NULOP,LUNITP)
C      CONTINUATION OF ALLIC
C
      REAL*8 DATE,YRMODA,TIT
      DIMENSION STR(1000),NSTR(1000),TITLE(3),TIT(5),MASK(32),AYA(90)
      1,PROH(6),SITH(6),CAGH(6),PMOD(6,10),SMOD(6,10),WT(3),DATE(10)
      DATA BLANK/' ',CNE/' 1',TITLE/'XBAR','NPTS','STDV'/
      1 ,TIT/'LYMPH ',MONOCYTE',MAT.NEUT',IMM.NEUT',EOSING. '/'
      NYMD(MDY)=10000*MOD(MDY,100)+100*(MDY/10000)+MOD(MDY,10000)/100
COMMENT INITIALIZE ALL SECTIONS
      18 MASK(1)=1
      DO 19 I=1,30
      19 MASK(I+1)=2*MASK(I)
      RETURN
COMMENT *****
      ENTRY WRALLQ(NSUE,MOS,STR,NSTR)
      IF(NSUB.EQ.9) GO TO 30
      IF(NSUB.NE.1) GO TO 400
      30 IF(NSTR(1).EQ.0) GO TO 400
COMMENT *****

```


COMMENT WRITE ANAMNESIS

```

ID1=NSTR(1)+1
DATE(1)=YRMOCA(NSTR(ID1+1))
DC 31 I=1,4
ICW=1+I+ID1+11
IF(I.LT.4) WT(I)=FLCAT(NSTR(ICW+1))/10.
DATE(I+1)=YRMOCA(NSTR(ICW))
31 DATE(I+5)=YRMOCA(NSTR(ICW+9))
DATE(10)=YRMOCA(NSTR(IC1+30))
ID4=ID1+3
IC13=ID1+12
WRITE(16,2) MOS,(NSTR(1), I = ID4, ID13), DATE(2), DATE(1),
1 WT(1),DATE(3),WT(2),DATE(4),WT(3),DATE(5),
2 NSTR(ID1+20),NSTR(IC1+21),DATE(6),NSTR(ID1+35),NSTR(ID1+29),
3 DATE(10),NSTR(ID1+23), DATE(7),NSTR(IC1+25),DATE(8),NSTR(ID1+27),
4 DATE(9),NSTR(IC1+2) ,NSTR (ID1+31), NSTR (ID1+32)
2 FORMAT('O MSERN MOUSE NAME SEX BORN DIED ',
1' WT 1 DATE 1 WT 2 DATE 2 WT 3 DATE 3 RAD2RATE AMT DATE ',
2' AGE' /1X,I6,2X,A2,1X,A3,5(1X,A2),2(1X,A1),1X,A2,2X,
3 2(1X,A7 ),1X,3(F4.1,1X,A7,2X), 6X,I1,I5,1X,A7,1X,I5/
4'C',9X,'CAGE DATE' ,35X,'ROOM DATE ROOM DATE RCGM DATE'
5,18X,'CCDE' /
6 9X,I6,1X,A7,34X,3(I2,2X,A7,3X),12X,A1,1X,A4,A1 )
IF(NSUB.NE.9) RETURN
COMMENT *****
400 IF(NSUB.EQ.9) GO TO 401
IF(NSUB.NE.2) GO TO 700
401 IF(NSTR(2).EQ.0) GO TO 700
COMMENT WRITE PATHOLOGY
C

```

```

ID1=NSTR(2)+2
DATE(1)=YRMOCA(NSTR(ID1+5))
WRITE(16,11) MCS,NSTR(IC1+3),NSTR(ID1+4),DATE(1),NSTR(ID1+6)
11 FORMAT('O MSERN EXPNC PROSEC DATE PATH'
1/'O',I6,I8,4X,A4,3X,A7,3X,A4)
IC=0
DC 475 I=1,3
IM=IC1+6+I
DC 475 J=1,30
IC=IC+1
AYA(IC)=BLANK
IF(IAND(MASK(J),NSTR(IM)).NE.0) AYA(IC)=ONE
475 CONTINUE
WRITE(16,12) (AYA(I),I=5,46),(AYA(I),I=1,4),(AYA(I),I=47,88)
12 FORMAT('O C S D X',13X,' S M T T B L L S P P B B T T L H ',
1'V S P L G E S I C C A T S P P E U K A B J C B P E E'/
2' I E I R',14X,'K / / N S N N A R R M M H H N R ',
3'S P A I B S T N O E N / / / / X R I D O Q R R I A Y '/
4' D C S A',14X,'N L G G T S C L P T F S Y R G T ',
5'L L N V L C O T L C S C U V C G B D R N I N N T R E'/
6/16X,' GROSS(1=NU) ',42A2/8X, 4A2,' MICRO(1=YES)',42A2
7// 'O',6X,'IMP PROCESS',18X,'CAUSE',20X,'SITE',21X,' DATA U'//)
IAUTC=ID1+10
500 NAUTO=NSTR(IAUTO+1)
DC 600 I=1,NAUTO
IC=IAUTO+I
ILOCD=NSTR(IC)+ID
CALL DECODE(STR(ILCCD+1),1,PROH)
CALL DECODE(STR(ILOCD+2),2,SITH)
CALL DECODE(STR(ILOCD+3),3,CAUH)
WRITE(16,13)NSTR(ILCCD+4),PROH,CAUH,SITH,NSTR(ILOCD+5),STR(ILOCD+6
1)

```

```

13 FCRMAT('O      ',I2,2X,3(6A4,1X),I5,1X,A1)
   ILOCPM=ILOCD+7
   NPMD=NSTR(ILOCPM)-1
   ILOCSM=ILOCPM+NPMD+1
   NSMD=NSTR(ILOCSM)-1
   JUP=MAX0(NPMD,NSMD)
   IF(JUP.EQ.0) GO TO 600
   JUPB=JUP+1
   DC 550 J=1,JUPB
   IF(J.LE.NPMD) GO TO 530
   DO 525 K=1,6
525  PMOD(K,J)=BLANK
   GO TO 535
530  ID=ILOCPM+J
   CALL DECODE(STR(ID),4,PMOD(1,J))
535  CONTINUE
   IF(J.LE.NSMD) GO TO 545
   DO 540 K=1,6
540  SMOD(K,J)=BLANK
   GO TO 550
545  ID=ILOCSM+J
   CALL DECODE(STR(ID),4,SMOD(1,J))
550  CONTINUE
   DC 575 J=1,JUP
   WRITE(16,8) (PMOD(K,J),K=1,6),(SMOD(K,J),K=1,6)
   8 FCRMAT(16X,6A4,25X,6A4)
575  CONTINUE
600  CONTINUE
   IF(NSUB.NE.9) RETURN
COMMENT *****
700  IF(NSUB.EQ.9) GO TO 702
3750 IF(NSUB.NE.3) GO TO 800
702  IF(NSTR(3).EQ.0) GO TO 800
C    WRITE HEMATOLCGY
C
   IC1=NSTR(3)+3
   WRITE(16,703)MUS
703  FORMAT('O MSERN      RED CELLS X10E6  WHITE CELLS X10E3      PLATEL
1ETS X10E6      HEMATOCRIT      DIFFERENTIAL'/1X,I6,25X,
2 '(N.8. -1. = MISSING DATA , -2. = SAMPLE FILED )'/)
   DO 705 I=1,5
   K=ID1-1+I
   IF(I.EQ.5) GO TO 704
   WRITE(16,706)      STR(K+16),STR(K+20),STR (K+24), STR (K+29),
1 STR(K+33),TIT(I)
706  FORMAT(      7X,2F19.1,3F19.0,2X,A8)
   GO TO 705
704  WRITE(16,707) STR(K+24),STR(K+33),TIT(I)
707  FORMAT(      45X,F19.0,19X,F19.0,2X,A8)
705  CONTINUE
   L=ID1-1
   DO 708 I=1,3
   K= L +2
   L=K+3
   WRITE(16,709) TITLE(I),(STR(I),N=K,L)
709  FORMAT(3X,A4,2F19.2,2F19.1)
708  CONTINUE
800  RETURN
END

```

```

SUBROUTINE DICT(IC)
C   PATHOLOGY DICTIONARY (READ,CHECK,ENCODE,DECODE )
C
  REAL*8 DATE
  INTEGER RDICT
  DIMENSION NOMOS(100),STRNOC(6,2),STRING(6),DICTOM(6,100)
  1, RDICT(2100),NRODE(2100),DICT(6,2100),NODE(4,2100)
  DATA STRNOC/6* ' ', 'NO S', 'UCH ', 'CODE', 3* ' ' /
  1, BLANK/ ' ', XLAST/ 'LAST' /, WORD/ 'WORD' /
  IF(IC.EQ.0) GO TO 400
  N5=IC
  INOEN=0
  NOMOS(1)=0
  DO 170 I=1,100
  DO 170 K=1,6
170 DICTOM(K,I)=BLANK
100 ID=0
  NSTOP=0
  READ (N5) MLIST, DATE, FTIM, ((DICT(K,I),K=1,6),ICOM,(NODE(K,I)
  1, K=1,4),I=1,MLIST),TLAST,TWORD
  IF (TLAST .NE. XLAST .AND. TWORD .NE. WORD) GO TO 9600
  NLIST = MLIST
  DO 110 I = 1,NLIST
  DO 110 J = 1,4
110 NODE (J,I) = IABS(NODE(J,I))
  WRITE(4,4)DATE,FTIM
  4 FORMAT('ODICTIONARY DATED ',A8,' AT ',F9.3,' HOURS')
  READ (N5)MRLIS, (NRODE(I),RDICT(I),I=1,MRLIS),ILAST
  IF (ILAST .NE. -99999999) GO TO 43
  NRLIS = MRLIS
  GO TO 140
9600 WRITE (4,9601) MLIST
9601 FORMAT('ODICTIONARY ERROR, THE FORWARD DICTIONARY IS TOO LONG OR T
  100 SHORT, MLIST = ',I5)
  CALL RECLOG
COMMENT STOP 00
COMMENT *****
  43 WRITE(4,44) MRLIS
  44 FORMAT('OREVERSE DICTIONARY TOO LONG OR TOO SHORT, MRLIS = ',I5)
  CALL RECLOG
COMMENT STOP 00
COMMENT *****
  140 DO 142 I=2,NLIST
  IF(ICOMPA(DICT(1,I-1),DICT(1,I),24))142,143,143
  142 CONTINUE
  GO TO 145
143 NSTOP=1
  WRITE(4,8) I
  8 FORMAT('ODICTIONARY ERROR DUAL OR DISORDERED ENTRY I= ',I5)
  145 DO 147 I=2,NRLIS
  IF(NRODE(I-1).GE.NRODE(I))GO TO 148
  147 CONTINUE
  GO TO 150
148 NSTOP=1
  WRITE(4,9) I
  9 FORMAT('ODICTIONARY ERROR DUAL OR DISORDERED CODE I= ',I5)
  150 IF(NSTOP.EQ.0) RETURN
  CALL RECLOG
COMMENT STOP 00
COMMENT *****

```

```

400 IF(INOEN.EQ.0) RETURN
    WRITE(4,10)(( NOMOS(I),( DICTOM(K,I),K=1,6),I=1,INOEN)
10  FORMAT('ODICTIONARY OMISSIONS'/'( ' SERIAL NO=',I7,'  WORD= ',6A4))
    RETURN
COMMENT *****
    ENTRY NCODE(String,IUSE,IDNO,NCOD)
    IF(IUSE.EQ.9) GO TO 400
    NWRDS=6
    IF(IUSE.EQ.3)NWRDS=5
    DO 255 K=1,NWRDS
    IF(String(K).NE.BLANK) GO TO 270
255  CONTINUE
    NCOD=0
    GO TO 310
270  CALL ISEARM(4*NWRDS,String,DICT,6,NLIST,NLOC)
    IF(NLOC.EQ.0)GO TO 300
    NCOD =NODE(IUSE,NLOC )
    IF(NCOD.NE.0) GO TO 310
300  INOEN=MINO(INOEN+1,100)
    NOMOS(INOEN)=IDNO
    DO 305K=1,NWRDS
305  DICTOM(K,INOEN)=String(K)
    NCOD=1
310  RETURN
COMMENT *****
    ENTRY DECODE(ITEM,IUSE,String)
    INUL=0
    IF(ITEM.NE.0)GO TO 205
    INUL=1
    GO TO 211
205  CALL ISEARS(4,ITEM,NRODE,NRLIS,NLOC)
    IF(INLUL.NE.0)GO TO 225
210  INUL=2
211  CONTINUE
    DO 215 K=1,6
215  String(K)=STRNOC(K,INUL)
    GO TO 235
225  CONTINUE
    ITT = RDICT(NLOC)
    DO 230 K=1,6
230  String(K)= DICT(K,ITT)
235  RETURN
    END

```

```

COMMENT PSORTM - MULTIPLE-ITEM SIMPLE POINTER SORT
C      NUMBER OF COMPARISONS=N**2/2
C      NBYTES=NUMBER OF BYTES FOR SORTKEY
C      LIST=DOUBLY DIMENSIONED ARRAY TO BE SORTED
C      NDL1=FIRST DIMENSION OF LIST
C      NDL2=HIGHEST INDEX OF SECOND DIMENSION OF LIST
C      NPTR=POINTER ARRAY , DIMENSION GE NDL2
C      LPTR=WORKING ARRAY , DIMENSION GE NDL2
COMMENT NO CHECK ON INITIAL ORDER
C *****
SUBROUTINE PSORTM(NBYTES,LIST,NDL1,NDL2,NPTR)
  DIMENSION LIST(NDL1,NDL2),NPTR(NDL2)
  DO 105 I=1,NDL2
105 NPTR(I)=I
  IF(NDL2.LE.1) RETURN
  IUP=NDL2-1
  DO 120 I=1,IUP
  JHOLD=I
  JLOW=I+1
  DO 115 J=JLOW,NDL2
  IF(ICOMPA(LIST(1,NPTR(J)),LIST(1,NPTR(JHOLD)),NBYTES))112,115,115
112 JHOLD=J
115 CONTINUE
  NPTRIH=NPTR(I)
  NPTR(I)=NPTR(JHOLD)
  NPTR(JHOLD)=NPTRIH
120 CONTINUE
  RETURN
  END

COMMENT ISEARS - SINGLE-ITEM BINARY SEARCH ROUTINE
C      NUMBER OF COMPARISONS=LOG(N)/LOG(2)
C      NBYTES=NUMBER OF BYTES FOR SEARCHKEY (1-4)
C      ITEM=KEY FOR SEARCH
C      LIST=SINGLY DIMENSIONED ARRAY TO BE SEARCHED
C      NDL=INDEX OF HIGHEST ELEMENT IN LIST
C      NLOC=INDEX OF MATCHING ENTRY IN LIST , IF FOUND
C      =0 IF NOT FOUND
C *****
SUBROUTINE ISEARS(NBYTES,ITEM,LIST,NDL,NLOC)
  DIMENSION LIST(NDL)
  IF(ICOMPA(ITEM,LIST(1),NBYTES))160,105,110
105 NLOC=1
  GO TO 170
110 IF(ICOMPA(ITEM,LIST(NDL),NBYTES))120,115,160
115 NLOC=NDL
  GO TO 170
120 ILO=0
  IHI=NDL+1
125 MID=(ILO+IHI)/2
  IF(ICOMPA(ITEM,LIST(MID),NBYTES))140,130,145
130 NLOC=MID
  GO TO 170
140 IHI=MID
  GO TO 150
145 ILO=MID
150 IF(IHI-ILO.GE.2) GO TO 125
160 NLOC=0
170 RETURN
  END

```

```

COMMENT  ISEARM - MULTIPLE-ITEM BINARY SEARCH
C        NUMBER OF COMPARISONS=LOG(N)/LOG(2)
C        NBYTES=NUMBER OF BYTES FOR SEARCHKEY
C        ITEM=KEY FOR SEARCH
C        LIST=DOUBLY DIMENSIONED ARRAY TO BE SEARCHED
C        NDL1=FIRST DIMENSION OF LIST
C        NDL2=HIGHEST INDEX OF SECOND DIMENSION OF LIST
C        NLOC=INDEX OF MATCHING ENTRY IN LIST , IF FOUND
C        =0 IF NOT FOUND
C *****

```

```

SUBROUTINE ISEARM(NBYTES,ITEM,LIST,NDL1,NDL2,NLOC)
  DIMENSION ITEM(NDL1),LIST(NDL1,NDL2)
  IF(ICOMPA(ITEM,LIST(1,1),NBYTES))160,105,110
105 NLOC=1
  GO TO 170
110 IF(ICOMPA(ITEM,LIST(1,NDL2),NBYTES))120,115,160
115 NLOC=NDL2
  GO TO 170
120 ILO=0
  IHI=NDL2+1
125 MID=(ILO+IHI)/2
  IF(ICOMPA(ITEM,LIST(1,MID),NBYTES))140,130,145
130 NLOC=MID
  GO TO 170
140 IHI=MID
  GO TO 150
145 ILO=MID
150 IF(IHI-ILO.GE.2) GO TO 125
160 NLOC=0
170 RETURN
END

```

```

COMMENT  ISORTS - SINGLE-ITEM SIMPLE SORTING ROUTINE
C        NUMBER OF COMPARISONS=N**2/2
C        NBYTES=NUMBER OF BYTES FOR SORTFIELD (1-4)
C        LIST=SINGLY DIMENSIONED ARRAY TO BE SORTED
C        NDL=INDEX OF HIGHEST ELEMENT IN LIST
COMMENT  NO CHECK ON INITIAL ORDER
C *****

```

```

SUBROUTINE ISORTS(NBYTES,LIST,NDL)
  DIMENSION LIST(NDL)
  IF(NDL.LE.1) RETURN
  IUP=NDL-1
  DO 120 I=1,IUP
    JHOLD=I
    JLOW=I+1
    DO 115 J=JLOW,NDL
      IF(ICOMPA(LIST(J),LIST(JHOLD),NBYTES))112,115,115
112 JHOLD=J
115 CONTINUE
    LISTH=LIST(I)
    LIST(I)=LIST(JHOLD)
    LIST(JHOLD)=LISTH
120 CONTINUE
  RETURN
END

```

FUNCTION NPRES(N)
 COMMENT PRESENT DATE IN NYMD FORM
 C

DIMENSION NDATE(2)
 ND=N
 CALL ITIME(ND,ND,NDATE)
 CALL CVBCBI(NDATE(1),NDATE(1))
 CALL CVBCBI(NDATE(2),NDATE(2))
 NPR=10000*MOD(NDATE(2),100)
 NPR=NPR+NDATE(2)/1000
 NPR=NPR+10*MOD(NDATE(1),10)
 NPR=NPR+100*(NDATE(1)/100)
 NPRES=NPR
 RETURN
 END

REAL FUNCTION YRMODA*8(NDATE)
 DATE IN FORM NNAANN (YR-MO-DA)
 C

DIMENSION AMO(12)
 REAL*8 DATE,BLDAT
 DATA AMO/'JAN','FEB','MAR','APR','MAY','JUN','JUL','AUG','SEP','OC
 1 T','NOV','DEC'/
 DATA BLDAT/' '/
 IF(NDATE.EQ.0) GO TO 10
 DATE=BLDAT
 GO TO 100
 10 NDA=MOD(NDATE,100)
 NMO=MOD(NDATE/100,100)
 NYR=NDATE/10000
 CALL INTBCD (NDA,DATE,I)
 IF(I.EQ.1) CALL INSERT(DATE,1,1,1H0)
 CALL INSERT(DATE,1,3,AMO(NMO))
 CALL INTBCD (NYR,ANYR,I)
 CALL INSERT(DATE,1,I,ANYR)
 IF(I.EQ.1) CALL INSERT(DATE,1,1,1H0)
 100 YRMODA=DATE
 RETURN
 END

FUNCTION NDAGE(NBIRTH,NDEATH)
 AGE (DAYS BETWEEN DATE2 - DATE1)
 C

DIMENSION NYMD(2),NDAYR(20),NDAMO(12),NDAS(2)
 DATA NDAMO/0,31,59,90,120,151,181,212,243,273,304,334/
 DATA NDAYR/0,366,731,1096,1461,1827,2192,2557,2922,3288,3653,4018,
 14383,4749,5114,5479,5844,6210,6575,6940/
 NYMD(1)=NBIRTH-590000
 NYMD(2)=NDEATH-590000
 DO 10 I=1,2
 NC=NYMD(I)
 NDA=MOD(NC,100)
 NMO=MOD(NC/100,100)
 NYR=NC/10000
 NDAS(I)=NDAYR(NYR)+NDAMO(NMO)+NDA
 IF(MOD(NYR,4).EQ.1.AND.NMO.GT.2) NDAS(I)=NDAS(I)+1
 10 CONTINUE
 NDAGE=NDAS(2)-NDAS(1)
 RETURN
 END

```

ITIME  START
*      PROGRAM RETURNS 2 PARAMETERS TO MAIN PROGRAM.
*      1ST IS FULL WORD INTERGER GIVING CLOCK IN 1/100 OF SECOND.
*      2ND IS DIFFERENCE BETWEEN LAST TWO CALLS TO SUBROUTINE.
*
*      SAVE REGISTERS AND POST SAVE AREAS
*
STM    14,12,12(13)
BALR   10,0
USING  *,10
LA     14,SAVEAREA          LOAD NEW SAVE AREA ADDRESS
ST     14,8(13)             NEW SAVE AREA ADDR TO OLD SAVE AREA
ST     13,4(14)             OLD SAVE AREA ADDR TO NEW
L      2,4(0,13)            ADDR OF SAVE AREA USED BY CMAIN PROG
LR     13,14                NEW SAVE AREA ADDR TO REG 13
*
*      OBTAIN TIME IN BINARY AND SUBTRACT FROM PRIOR TIME
*
LR     4,1                  SAVE ADDR OF PARM LIST
TIME   BIN                  GET TIME IN REG 0
L      2,0(0,4)             LOAD ADDR OF 1ST PARM IN REG 2
ST     0,0(0,2)             STORE TIME IN PARM LIST
CLI    SW,X'FF'             IS THIS 1ST ENTRY
BC     8,BYPASS             CHECK FOR EQUAL
ST     0,TIMEAREA
BYPASS LR     3,0            LOAD TIME IN R3
S      3,TIMEAREA           SUBTRACT PRIOR TIME FROM CURRENT
ST     0,TIMEAREA           SAVE CURRENT TIME
MVI    SW,X'FF'            MOVE FF TO SWITCH
LTR    2,2                  CK FOR LAST PARM
BC     4,OUT                BRANCH IF NEG
L      2,4(0,4)             LOAD ADDR OF 2ND PARM IN REG 2
LTR    3,3                  CHECK FOR NEG ELAPSED TIME
BC     4,NEG
RC     15,POSITIVE
NEG    A      3,TWHR5        ADD 24 HRS IN 1/100 SEC
POSITIVE ST  3,0(0,2)        STORE DIFF IN PARM LIST
LTR    2,2
RC     4,OUT
*
*      GET DATE AND CONVERT TO MONTH, DAY, YEAR
*
ST     1,PDATE
MVC    PHOLD+5(2),PDATE+2
CVB    3,PHOLD
LR     5,3
MVD    PHOLD(8),PDATE+1(1)
UNPK   YEAR(2),PHOLD+6(2)
DP     PHOLD,FOUR
TM     PHOLD+7,X'F0'
DC     5,AROUND
LEAPYEAR CH  3,DATYAR+2
BC     13,AROUND
CH     3,SIXTY
BC     0,EXTRA
RCTR   3,0                  COMPARE TO 60
AROUND LA  9,22              BRANCH IF THIS IS EXTRA DAY
LA     6,0                  SUBTRACT 1 FROM DAY
LA     8,2
CHECK  CH  3,DATYAR(6)
BC     2,INCRFM

```



```

EX      LH      11,MOTABLE(6)      LOAD MONTH IN R11
        STH     11,MO
        SH      3,DAYTAB-2(6)
        CVD     3,PHOLD
        UNPK    DAY(2),PHOLD+6(2)
        OI      DAY+1,X'FO'      SET SIGN ZONE TO F
        BC      15,MOVE
EXTRA   LA      6,2      LOAD 2 IN R6
        BC      15,EX      BRANCH TO EX
INCREM  RXLE    6,8,CHECK
        MVC     DATE,ERROR
MOVE    L        2,8(0,4)      LOAD REG 2 WITH ADDR OF 3RD PARM.
        MVC     0(8,2),DATE    MOVE DATE TO 3RD PARAM
        LTR     2,2
        BC      4,OUT
        L        2,12(0,4)     LOAD R2 WITH ADDR OF 4TH PARM
        ST      5,0(0,2)     STORE DAY IN 4TH PARM

*
*      RESTORE REGISTERS AND RETURN TO MAIN PROG
*
OUT      L        13,4(13)      RESTORE REGISTER 13
        LM      14,12,12(12)   RESTORE REGISTERS
        BR      14      RETURN TO MAIN PROGRAM
TIMEAREA DS      F      STG AREA FOR TIME
SAVEAREA DS      18F
DATE     DS      0CL8
MO       DS      CL2
        DC      C'/'
DAY      DS      CL2
        DC      C'/'
YEAR     DS      CL2
DAYB     DS      F
PDATE    DS      PL4
        CNOP    0,8
PHOLD    DC      PL8'0'
FOUR     DC      PL1'4'
ERROR    DC      CL8' ERROR '
        CNOP    0,8
MOTABLE  DS      0CL24
        DC      CL2'01'
        DC      CL2'02'
        DC      CL2'03'
        DC      CL2'04'
        DC      CL2'05'
        DC      CL2'06'
        DC      CL2'07'
        DC      CL2'08'
        DC      CL2'09'
        DC      CL2'10'
        DC      CL2'11'
        DC      CL2'12'
DAYTAB   DC      H'00'
        DC      H'31'
        DC      H'59'
        DC      H'90'
        DC      H'120'
        DC      H'151'
        DC      H'181'
        DC      H'212'
        DC      H'243'
        DC      H'273'

```

```

        DC      H'304'
        DC      H'334'
        DC      H'365'
        DC      H'60'
SIXTY   DC      X'83D600'
TWHR5   DC      CL1'0'
SW       DC      ITIME
        END

```

```

HEX FOR 24 HRS IN 1/100 SEC
SWITCH TO TEST FOR 1ST PASS THRU ROU

```

```

*      EBCDIC TO BINARY NUMBER CONVERSION
*      ZONE BITS IGNORED
CVBCBIDK CSECT
        USING *,15
        ENTRY CVBCRT
CVBCRT  B       12(15)
        DC      X'7'
        DC      CL7'CVBCRT'
        STM     2,3,SAVE
        L       2,0(1)
        MVC     WORK(4),0(2)
        NI      WORK+3,X'0F'
        OT      WORK+3,X'CD'
        PACK    WORK1+4(4),WORK(4)
        CVB     3,WORK1
        L       2,4(1)
        ST      3,0(2)
        LM      2,3,SAVE
        BR      14
SAVE    DS      2F
        DS      0D
WORK1   DC      2F'0'
WORK    DS      1F
        END

```

```

INTBCD  START 0
        USING *,15
        ENTRY INTBCD
        ENTRY INTBC  THIS IS A TRUNCATED 6-LETTER FTM-IV ENTRY.
        ENTRY INTBCD THIS IS A TRUNCATED 6-LETTER FTM-IV ENTRY.
INTBC   EQU  *      THIS IS A TRUNCATED 6-LETTER FTM-IV ENTRY.
        STM 2,4,SAVERE
        LM  1,2,0(1)      X1=A(INTEGER) X2=A(BCD)
        MVI BRAN+1,X'FO'   SET TO BRANCH AT BRAN (NO LENGTH)
        LA  15,INTBCDL-INTBCD(15)
        USING INTBCDL,15
        B    INRCL1
INTBCD  EQU  *      THIS IS A TRUNCATED 6-LETTER FTM-IV ENTRY.
INTBCDL STM 2,4,SAVERE
        LM  1,3,0(1)      X1=A(INTEGER) X2=A(BCD) X3=A(LENGTH)
        MVI BRAN+1,X'OO'   SET TO NOT BRANCH AT BRAN, GET LENGTH
INRCL1  L    1,0(0,1)      X1=INTEGER
        LTR 1,1           TEST FOR SIGN
        BP  GO            NO SIGN IF +
        BM  GO1           BRANCH IF NEG., CONTINUE IF 0
        MVI 0(2),X'FO'     SET 'FO' IN STRING
        MVI 1(2),X'OO'     SET 'OO' IN STRING+1
        LA  4,1           X4=1=LENGTH
        B    BRAN         BRANCH TO SEE IF LENGTH IS REQUIRED
GO1     MVI 0(2),X'FO'     PUT MINUS SIGN IN STRING
        MVI NEG+1,X'OO'    SET NO BRANCH IN NEG TO INC LENGTH
        LA  2,1(0,2)
GO      CVD 1,DUB          CONVERT TO PACK. DECIMAL IN DUB
        NI  DUB+7,X'OF'    SET SIGN TO 'F' TO UNPK AS ZONE
        UNPK TEMP(10),DUB+2(6) UNPACK 10 DIGITS INTO TEMP
        SR  1,1           ZERO X1
        LR  0,2           SAVE X2 IN X0
        TRT TEMP(10),INTBCDL SCAN FOR NON 'FO' BYTE, X1=A(NON FO)
        LR  2,0           RESTORE X2 FROM X0
        LA  4,TEMP+10      X4=A(TEMP)+10
        SR  4,1           X4=NO OF DIGITS TO MOVE
        STC 4,M+1          STORE X4 IN M+1, WILL MOVE DIG AND 00
        MVC 0(0,2),0(1)    MOVE DIG AND 00 INTO STRING
        BR  NOL            BRANCH IF LENGTH NOT REQUIRED
        B    NON          BRANCH IF NOT NEGATIVE
        LA  4,1(0,4)      INCREASE X4 BY 1 TO GIVE LENGTH
        ST  4,0(0,3)      STORE LENGTH IN 'LENGTH'
        MVI NEG+1,X'FO'    SET BRANCH IN NEG
        LM  2,4,SAVERE
        BR  14
DUB     DS  D
SAVERE  DS  3F
TEMP    DC  3F'0'
        ORG INTBCDL+240
        DC  X'00',9X'01'
        ORG
        END

```

INTBCD10

INTBCD28

```

ICOMPARE START 0          ICOMPARE(STRB,N) =0 IF N.LE.0
ENTRY ICOMPARE THIS AN ENTRY FOR FTN IV
ICOMPARE EQU *          THIS AN ENTRY FOR FTN IV
USING *,15          ((N-1) MODULO 256) + 1 FOR N.GT.256
* BY RPR, JULY 18, 1966.
ST 2,SAVE
L 2,8(0,1)          X2 = A(N)
L 2,0(0,2)          X2=N
LA 0,1          X0=+1
SR 2,0          X2=N-1
BZ N1          BRANCH TO N1 IF N=1
BM DONE1          RETURN 0 IN N.LE.0
STC 2,COM+1          STORE N-1 IN COM+1 (COMPARE N CHAR.)
L 2,4(0,1)          X2=A(STRB)
L 1,0(0,1)          X1=A(STRB)
COM CLC 0(0,1),0(2)  COMPARE N CHAR. OF STRA WITH STRB
L 2,SAVE          THIS LOAD DOES NOT EFFECT THE CONDITION CODE
BL MONE
BE DONE
* PREVIOUS, LA SET X0=+1          X0=+1 STRA.GT.STRB
BR 14
MONE LNR 0,0          X0=-1 STRA.LT.STRB
BR 14
DONE1 L 2,SAVE
DONE SR 0,0          X0=0 STRA.EQ.STRB
BR 14
* THIS SECTION IS FOR N=1
N1 L 2,4(0,1)          X2=A(STRB)
L 1,0(0,1)          X1=A(STRB)
IC 2,0(0,2)          X2=FIRST CHAR OF STRB
STC 2,COM1+1          STORE FIRST CHAR OF STRB IN COM1
COM1 CLI 0(1),X'00'    COMPARE 1 CHAR
L 2,SAVE
BL MONE
BE DONE
*
BR 14          X0=+1 STRA.GT.STRB
SAVE DS F
FND

```

03157RPR
11186RPR
11186RPR

.85

.7

```

INSERT  START 0
*      THIS IS A VERSION OF INSERT ET AL WHICH USES THE NEW
*      FAST VERSION OF SEARCH.
      USING *,15
      ENTRY LENGTH
      ENTRY DELETE
      ENTRY SEAPCH
      ENTRY PUTIN
      ENTRY PUTINAT
      ENTRY PUTINA THIS IS A TRUNCATED 6-LETTER FTN-IV ENTRY.
      STM 2,7,SAVERE INSERT ( STRING , J , N , Q )
      ST 15,SAVE15
      LM 1,4,0(1) X1=A(STRING) X2=A(I) X3=A(J) X4=A(Q)
* ENTRY AT INSERTB X1=A(STRB) X2=A(I) X3=A(N) X4=A(STRB)+J-1
INSERTB L 2,0(0,2) X2=I
* ENTRY AT INSERTA X1=A(STRB) X2=0 X3=A(N) X4=A(STRB)+I-1
INSERTA LA 0,256
AAI LTR 6,2 X6#I
BC 12,BRANA BRA I#0 OR I MINUS
L 3,0(0,3) X3#J
LTR 3,3
BC 12,EXIT BRANCH OUT IF J .EQ.0 OR IS MINUS
LR 5,1 X5#A%STR<
TRTI TRT 0(256,1),TRTAB4
RC 6,ENDI
LA 1,256(0,1)
B TRTI
ENDI SR 1,5 X1#L
LR 2,1 X2#L
SR 2,6 X2#L-I
BM BRANC BRANCH IF L .LT. I
LA 7,2(0,2) X7#M#L-I&2
CCI ALR 5,6 X5=A(STRING)+I
SL 5,=F'1' X5=A(STRING)+I-1 = PZ
CR 7,0 M VS 256
BH BRANC BRANCH IF M .GT. 256
DDI S 7,=F'1' X7=M-1
STC 7,INSA+1
STC 7,INSB+1
INSA MVC TEMPST(0),0(5)
LR 7,3 X7#J
ALR 7,5 X7=J+PZ
INSB MVC 0(0,7),TEMPST
FEI CR 3,0 J VS 256
BH BRANC BRANCH IF J .GT. 256
S 3,=F'1' X3=J-1
STC 3,INSC+1
INSC MVC 0(0,5),0(4)
EXIT LM 2,7,SAVERE
L 15,SAVE15
BR 14
BRANA MVI 0(1),X'00'
LA 2,1 MAKE I#1
B AAT
BRANC LA 7,1 X7#M#1
LA 6,1(0,1) MAKE I#L&1
B CCI
BRAND ALR 1,5 X1=PZ+L
SLR 1,6
SL 1,=F'256'
LR 2,3 X2#J

```

2.60

```

BRANDA ALR 2,1 X2=A1+J
MVC TEMPST(256),0(1)
MVC 0(256,2),TEMPST
SR 7,0 X7#M-256
CR 7,0 M VS 256
RC 12,DDI
SLR 2,0 X2=A1-256
SLR 1,0 X1=AJ-256
B BRANDA
BRANE MVC 0(256,5),0(4)
ALR 5,0 PZ=PZ+256
ALR 4,0 QO=QO+256
SR 3,0 J-256#J
B EE1
EJECT
USING *,15 X2 X3 X4 X5 X6
SEARCH STM 2,14,SAVERE SAVE REGISTERS
* SEARCH(STRING, J, N, CHARS, I)
*
* X0 SET TO 2 FOR SOME USES
* X1 USED FOR START AND RETURN IN TRT
* X2 AFTER A(STRING) IS SAVED IN X14, X2 RETURNS FROM TRT
* X3 X3=256 AFTER FREE FROM J
* X4 N
* X5 A(CHARS)
* X6 A(I)
* X7 LOCATION IN TRT TABLE
* X8 USED WITH X1 IN MULT COMPARE
* X9 X9 = 4096
* X10 USED WITH X5 IN MULT COMPARE
* X11 X11 = 1
* X12 USED AS DOUBLE REG. WITH X13 FOR SHIFT ON N
* X13 USED AS DOUBLE REG. WITH X12 IN SHIFT OF N ETC.
* X14 USED AS SAVE REGISTER FOR A(STRING) - 1 FROM X2
EJECT
LM 2,6,0(1) X2 X3 X4 X5 X6
LA 11,1 X11=1
SR 2,11 X2=A(STRING)-1
LR 14,2 SAVE A(STRING) - 1 FROM X2
LA 0,2 X0 = 2 FOR USE IN ALL OF SEARCH
LA 9,4095(0,11) X9=4096
L 1,0(0,3) X1=J
ALR 1,2 X1=A(STRING)+J-1 (START)
LA 3,256 X3=256
SR 2,2 ZERO X2 FOR USE AS RETURN BY TRT
SP 7,7 X7=0
IC 7,0(0,5) X7=FIRST CHARACTER OF CHARS
LA 7,TRTAB4(7) X7=ADD. IN TRTAB4 FOR X'01' TO BE SET
MVI TRTAB4,X'04' SET X'04' IN START OF TRTAB4
* FOR BRANCH TO END22+4
SPACE 3
L 4,0(0,4) X4#N
SR 4,11 X4=N-1
RM RYPA BRANCH IF N.LE.0
BZ N1 IF N=1 BRANCH TO N1
CR 4,3 COMPARE N-1 TO 256
PH MULTI BRANCH IF N.GT.257
CR 4,0 COMPARE N-1 TO 2
BL N2 BRANCH FOR N=2
BE N3 BRANCH FOR N=3
* CONTINUE TO LE257 FOR NLE.257 BUT MORE THAN 3
EJECT

```

```

LE257  MVI 0(7),H2-END22  FOR N.LE.257
      SR 4,11             X4 = N-2
      STC 4,H2+1
      R LOOP2N
      SPACE 3
MULTI  MVI 0(7),06257-END22  FOR N.GT.257
      LR 12,4             PUT N-1 IN X12
      SR 13,13            ZERO X13
      SRDL 12,8           SHIFT SMALL CDM. INTO X13, X12=N256
      LTR 13,13           TEST FOR NON-INTEGRAL MULT OF 256
      BZ CONX1            PRANCH IF INTEGRAL MULT OF N*256+1
      SRL 13,24           SHIFT SMALL INTO RIGHT
      SR 13,11            X13=N-2
      STC 13,H3+1         STORE N MOD 256 IN CLC H3
      LR 13,12            STORE NO. OF 256 CYCLES IN X13
      B LOOP2N
CONX1  MVI INTMUL+1,X'F0'  SET BRANCH IN INTMUL IF INTEGRAL
      LR 13,12            STORE NO. OF 256 CYCLES IN X13
      B LOOP2N
      SPACE 3
N2     MVI 0(7),CN2-END22  FOR N.EQ.2
      IC 0,1(0,5)         INSERT SECONE CHAR. IN X0
      STC 0,CN2+1         STORE SECONE CHAR. IN CLI AT CN2
      B LOOP2N
      SPACE 3
N3     MVI 0(7),CN3A-END22  FOR N.EQ.3
      IC 0,1(0,5)
      STC 0,CN3A+1
      IC 0,2(0,5)
      STC 0,CN3B+1
      B LOOP2N
      SPACE 3
N1     MVI 0(7),FOUND-END22  FOR N.EQ.1
      B LOOP2N
      EJECT
LOOP2NI AR 1,11           INCREASE X1 BY 1
LOOP2N TRT 0(256,1),TRTAB4
      BC 6,END22(2)
      TRT 256(256,1),TRTAB4
      BC 6,END22(2)
      TRT 512(256,1),TRTAB4
      BC 6,END22(2)
      TRT 768(256,1),TRTAB4
      BC 6,END22(2)
      TRT 1024(256,1),TRTAB4
      BC 6,END22(2)
      TRT 1280(256,1),TRTAB4
      BC 6,END22(2)
      TRT 1536(256,1),TRTAB4
      BC 6,END22(2)
      TRT 1792(256,1),TRTAB4
      BC 6,END22(2)
      TRT 2048(256,1),TRTAB4
      BC 6,END22(2)
      TRT 2304(256,1),TRTAB4
      BC 6,END22(2)
      TRT 2560(256,1),TRTAB4
      BC 6,END22(2)
      TRT 2816(256,1),TRTAB4
      BC 6,END22(2)
      TRT 3072(256,1),TRTAB4

```

```

BC      6,FND22(2)
TRT     3328(256,1),TRTAB4
BC      6,FND22(2)
TRT     3584(256,1),TRTAB4
BC      6,FND22(2)
TRT     3840(256,1),TRTAB4
BC      6,FND22(2)
END22   BXH  1,9,LOOP2N      INCREASE X1 BY 4096 AND CONTINUE TRT
* THIS IS THE LOCATION TO WHICH THE BRANCH WILL GO WHEN X'00' FOUND
SR      2,2                  ZERD X2
ST      2,0(0,6)             STORE 0 IN I
MVI     0(7),X'00'           RESTORE X'00' IN TRTAB4
MVI     TRTAB4,X'02'         RESTORE X'02' AT START OF TRTAB4
LM      2,14,SAVERE          RESTORE REGISTERS
RR      14
EJECT
* THE FOLLOWING SECTION IS FOR N.LE.257
H2      CLC  1(0,1),1(5)      COMPARE STARTING AT TRT MATCH + 1
BNE     LOOP2NI
BE      FOUND
EJECT
SPACE 3
SPACE 3
* THE FOLLOWING SECTION IS FOR N.GT.257
OG257   LR   8,1              STORE CURRENT MATCH FROM TRT IN X8
LR      10,5
LR      13,12                X13=NUMBER OF MULT. OF 256 TO DO
R       FIRSTMUL
MULCOM  AR   10,3             INCREASE X10 BY 256
AR      8,3                  INCREASE X8 BY 256
FIRSTMUL CLC  1(256,8),1(10)  COMPARE 256 BYTES
BNE     LOOP2NI              INCREASE X1 BY 1 AND TRT
BC      13,MULCOM             LOOP FOR X13 GROUPS OF 256
INTMUL  BC   0,FOUNDRES        BRANCH ONLY IF INT. MULT OF 256
H3      CLC  257(0,8),257(10) COMPARE NON INT. MUL OF 256
BNE     LOOP2NI
B       FOUND
FOUNDRES MVI  INTMUL+1,X'00'    RESTORE NO BRANCH TO INTMUL
B       FOUND
EJECT
SPACE 3
* THE FOLLOWING SECTION IS FOR N=2
CN2     CLI  1(1),X'00'        COMPARE SECOND CHAR. WHEN FIRST MATCH
BNE     LOOP2NI
B       FOUND
SPACE 3
* THE FOLLOWING SECTION IS FOR N=3
CN3A    CLI  1(1),X'00'
BNE     LOOP2NI
CN3B    CLI  2(1),X'00'
BNE     LOOP2NI
B       FOUND
SPACE 3
* THE FOLLOWING SECTION IS FOR N#1
* AND THE COMMON POINT WHEN A MATCH IS FOUND.
FOUND   SR   1,14             X1=VALUE TO STORE IN I, X6=A(I)
ST      1,0(0,6)             STORE LENGTH IN IFND
MVI     0(7),X'00'           RESTORE X'01' TO X'00' IN TRTAB4
BYPA    MVI  TRTAB4,X'02'     RESTORE X'02' TO START OF TRTAB4
LM      2,14,SAVERE          RESTORE REGISTERS
BR      14

```


	EJECT	ADDRESS OF ARG LIST #X1	TIME
LENGTH	USING *,15	LENGTH %STRING, LENG<	
	STM 2,3,SAVERE	X3# A%LENGTH<	.80
	L 3,4(0,1)	X1# A%STRING<	.80
	L 1,0(0,1)	X0 # A%STRING<	.40
TRTL	LR 0,1	FINDS TERMINAL CHARACTER	3.908
	TRT 0(256,1),TRTAB4		1.13
	BC 6,ENDL	BUMP UP BY 256	.60
	LA 1,256(0,1)		1.13
	B TRTL	A%TERM)-A(STRING)#LENGTH	.40
ENDL	SLR 1,0	STORE LENG	.94
	ST 1,0(0,3)		
	LM 2,3,SAVERE		
	BR 14		1.13
	USING *,15		
DELETE	STM 2,5,SAVERE	DELETE(STRING,I,N<	
	LM 1,3,0(1)	X1#A%STR< X2#A%I< X3#A%N< 2.00	
	L 3,0(0,3)	X3=N	
	LTR 3,3	TEST FOR N.LE.0	
	BC 12,BYPASS	N.LT.0R#0 SKIP	1.13
	L 4,0(0,2)	X4=I X2 IS FREE FOR USE IN TRT	
	S 4,=F'1'	X4=I-1 TEST FOR I.LE.0	1.13
	BM BYPASS		
	LR 0,1	X0#A%STRING<	
TRTD	TRT 0(256,1),TRTAB4		
	BC 6,ENDD		
	LA 1,256(0,1)		
	B TRTD		
ENDD	SLR 1,0	X1=A(TERM)-A(STRING)=LENGTH	.40
	SR 1,4	X1=L-I+1	
	BC 12,BYPASS	IF I.GT.L	1.13
	ALR 4,0	X4=A(STRING)+I-1 MOVE DOWN TO	
	SR 1,3	X1=L-I+1+N=MM-M-1	
	BC 11,NOR1	BRANCH TO NOR1 IF MM NOT -	
	MVI 0(4),X'00'	DELETE ALL AT STR(I) PUT 00 END	
	B BYPASS		
NOR1	ALR 3,4	X3=A(STRING)+I-1+N MOVE DOWN FROM	
CYCLE1	LR 5,1	X5#MM	
	S 5,=F'255'	X5=MM-255	
	BC 12,ONCE	BRANCH IF MM.LE.255	
	MVC 0(256,4),0(3)	MOVE 256 CHAR TO NEW LOCATION	
	LA 3,256(0,3)	X3=X3+256	
	LA 4,256(0,4)	X4=X4+256	
	S 1,=F'256'	MM=MM-256	
	B CYCLE1		
ONCE	STC 1,C+1	LOAD MM#M-1 INTO MNC INST. C	
C	MVC 0(0,4),0(3)	MOVE M CHAR TO NEW LOCATION	
BYPASS	LM 2,5,SAVERE		2.20
	BR 14		1.13
	USING *,15		
PUTINA	EQU *	THIS IS A TRUNCATED 3-LETTER FTN-IV ENTRY.	
PUTINAT	STM 2,7,SAVERE	PUTINAT(STRA,I,N,STRB,J<	
	ST 15,SAVE15		
	LM 1,5,0(1)	X1=A(STRA) X2=A(I) X3=A(N) X4=A(STRB) X5=A(J)	
	A 4,0(0,5)	X4=A(STRB)+J	
	S 4,=F'1'	X4=A(STRB)+J-1	
	L 15,=A(INSERT)		
	USING INSERT,15		
	B INSERTB	CHANGE TO INSERT	
	USING *,15		
PUTIN	STM 2,7,SAVERE	PUTIN(STRA,N,STRB,I)	

```

      ST      15,SAVE15
      LM      1,4,0(1)
      L       4,0(0,4)
      ALR     4,3
      S       4,=F'1'
      LR      3,2
      SR      2,2
      L       15,=A(INSERT)
      USING   INSERT,15
      B       INSERTA
TRTAB4  DC     X'02'
      DC     255X'00'
SAVERE  DS     13F
SAVE15  DS     F
TEMPST  DS     256C
      LTORG
      END

```

X1=A(STRB) X2=A(N) X3=A(STRB) X4=A(I)
 X4=I
 X4=A(STRB)+I
 X4=A(STRB)+I-1
 X3=A(N) SET UP TO USE INSERT

SAVE REGISTERS

```

IAND    START 0          IAND(ARG1,ARG2)    AND, FIXED POINT RETURN
      USING *,15
      ENTRY AND
      MVI     WHICH+1,X'F0'    SET TO BRANCH AT WHICH
      LA      15,AND-IAND(15)
      USING   AND,15
      B       MERGE
AND MVI  WHICH+1,X'00'        AND(ARG1,ARG2) AND, FLOAT. POINT RETURN
MERGE    ST      2,PASS        SAVE X2 IN PASS
      LM      1,2,0(1)        X1=ADD. ARG1 X2=ADD. ARG2
      L       0,0(0,1)        X0=FIRST ARG.
      N       0,0(0,2)        AND SECOND ARG. AGAINST X0
      L       2,PASS          RESTORE X2 FROM PASS
      B       DONE
WHICH    ST      0,PASS
      LE      0,PASS
DONE     BR      14
PASS     DS      F
      END

```

C. Dictionary Update Program

COMMENT DICTIONARY UPDATE PROGRAM

```

C
  DIMENSION DICT(6,2100),NODE(4,2100),NPTR(2100),MCD(4),ADICT(6),SY
  N1(6),SYN2(6),          LPTR(2100),NRODE(2100),ICOM(2100)
  2,MPTR(2100)
  REAL*8 DATE
  REAL*8 USE(4)
  DATA ADD/'ADD '//,DEL/'DEL '//,SYN/'SYN '//,BLANK/' ' '//,XLAST/'LA
  1ST'/',WORD/'WORD'//
  DATA DATE/'01/01/01'//
  DATA COM/'COM '//
  DATA SOR/'SOR '//
  READ(5,43) N5,NCOP,USE
  43 FORMAT(I6, I6, 4(1X,A8))
  DO 105 K=1,4
  105 MCD(K)=0
  IF (N5.NE. 0) GO TO 217
  MLIST = 1
  FTIM = 0.
  NODE(1,1) = 1000000
  NODE(2,1) = 2000000
  NODE(3,1) = 3000000
  NODE(4,1) = 4000000
  DO 222 I = 1,6
  DICT(1,1) = BLANK
  222 DICT(1,2) = BLANK
  DICT(1,2) = XLAST
  DICT(2,2) = WORD
  ICOM(1) = 0
  GO TO 220
  217 CONTINUE
C
C      READ IN OLD DICTIONARY
C
  READ (N5) MLIST, DATE, FTIM, ((DICT(K,I),K=1,6),ICOM(I),(NODE(K,I)
  1 ,K=1,4),I=1,MLIST),TLAST,TWORD
  IF (TLAST.NE. XLAST.AND. TWORD.NE. WORD) GO TO 9600
  IF(N5.NE.5) REWIND N5
  220 MLIST = MLIST
  WRITE(6,4) DATE,FTIM
  4 FORMAT('UPDATE OF DICTIONARY DATED ',A8,' AT ',F9.3)
  219 DO 110 I = 1,MLIST
  C   WRITE(6,51)((DICT(K,I),K=1,6),ICOM(I),(NODE(K,I),K=1,4)
  C   51 FORMAT(1X,6A4,14,4(1X,I12))
  110 NPTR(I) = I
  CALL ITIME(NTIM,NOTIM,DATE,NAYR)
  FTIM=FLOAT(NTIM)/250000.
  117 DO 125 I=1,MLIST
  DO 120 K=1,4
  MCD(K)=MAX0(IABS(NODE(K,I)),MCD(K))
  120 CONTINUE
  125 CONTINUE
  C   WRITE(6,52) MLIST,MCD
  C   52 FORMAT(10I9)
  CALL PSORGM(24,DICT,6,MLIST,NPTR,MPTR)
  ID=MLIST
C
C      READ IN NEW WORD TO BE PROCESSED
C

```

```

150 READ(5,2,END=400)ADICT,ACT,NUS1,SYN1,NUS2,SYN2
2  FORMAT(6A4,A3,1X,11,1X,6A4,11,1X,6A4)
   IF(ADICT(1).EQ.XLAST.AND.ADICT(2).EQ.WORD) GO TO 400
   CALL PSEARM(24,ADICT,DICT,6,NLIST,NLOC,NPTR)
   IF(ACT.NE.SOR) GO TO 151

C
C      SORT DICTIONARY ON SORT COMMAND
C
   NLIST=ID
   WRITE(6,56) NLIST
56  FORMAT('ODICTIONARY RESORTED NUMBER OF ENTRIES ',I6)
   GO TO 117
151 ICOM(ID+1)=0
   DO 152 I=1,4
152 NODE(I,ID+1)=0
   IF(ACT.NE.ADD)GO TO 200

C
C      ADD WORD TO DICTIONARY
C
   IDD=NPTR(NLOC)
   IF(NUS1.EQ.0) GO TO 160
   IF(NLOC.EQ.0.OR.NODE(NUS1,IDD).EQ.0) GO TO 153
   WRITE (6,10) ADICT,NUS1
10  FORMAT(' PRIOR ENTRY FOR ',6A4,3X,'USE= ',I3,' NO TRANSACTION')
   GO TO 160
153 IF(NLOC.NE.0)GO TO 154
   ID=ID+1
   IDD=ID
   NPTR(ID)=ID
154 DO 155 K=1,6
155 DICT(K,IDD)=ADICT(K)
   MCODE(NUS1)-MCOD(NUS1)+1
   NODE(NUS1,IDD)=MCOD(NUS1)
160 IF(NUS2.EQ.0) GO TO 162
   IF(NLOC.EQ.0.OR.NODE(NUS2,IDD).EQ.0) GO TO 161
   WRITE(6,10) ADICT,NUS2
   GO TO 150
161 MCODE(NUS2)=MCOD(NUS2)+1
   NODE(NUS2,IDD)=MCOD(NUS2)
162 GO TO 150
200 IF(ACT.NE.DEL) GO TO 300

C
C      DELETE WORD FROM DICTIONARY
C
   IF(NLOC.NE.0) GO TO 210
201 WRITE(6,11)ADICT,ACT
11  FORMAT(' NO ENTRY FOR ',6A4,3X,A4)
   GO TO 150
210 IDLO=NPTR(NLOC)
   IF (NUS1 .EQ. 0 .AND. NUS2 .EQ. 0) GO TO 325
   IF (NUS1 .NE. 0) NODE(NUS1,IDLO) = 0
   IF (NUS2 .NE. 0) NODE(NUS2,IDLO) = 0
   DO 232 K = 1,4
232 IF (NODE(K,IDLO) .NE. 0) GO TO 234
   DO 233 K = 1,6
233 DICT(K,IDLO) = BLANK
   WRITE (6,238) ADICT
238 FORMAT(' ENTRY ',6A4,' DELETED ')
   GO TO 150
234 CONTINUE
   IF (NUS1 .EQ. 0) GO TO 237

```

```

      IF (NUS2.NE.0) GO TO 235
      WRITE (6,53) ADICT, NUS1
53  FORMAT (' CODE FOR ',6A4,' DELETED, USE= ',I3)
      GO TO 150
235  WRITE (6,236) ADICT, NUS1,NUS2
236  FORMAT (' CODE FOR ',6A4,' DELETED, USES= ',I3,' ',I3)
      GO TO 150
237  WRITE (6,53) NUS2
      GO TO 150
300  IF (ACT.NE.SYN.AND.ACT.NE.COM) GO TO 325
C
C      ADD SYNONYM TO DICTIONARY
C      ADD COMPOUND WORD TO DICTIONARY
C
      IF (NUS1.EQ.0) GO TO 325
      CALL PSEARM(24,SYN1,DICT,6,NLIST,NSYN,NPTR)
      IDLO=NPTR(NSYN)
      IF (NSYN.NE.0.AND.NODE(NUS1,IDLO).NE.0) GO TO 305
      WRITE(6,12) SYN1,ADICT,NUS1
12  FORMAT('OND SYNONYM ',6A4,' FOR '6A4,' USE = ',I3)
      GO TO 320
305  IDLO=NPTR(NSYN)
      IF (NLOC.NE.0) GO TO 306
      ID=ID+1
      IDD=ID
      NPTR(ID)=ID
      GO TO 307
306  IDD=NPTR(NLOC)
      WRITE(6,14) ADICT,SYN1,NUS1
14  FORMAT('OPRIOR ENTRY- ',6A4,' REDEFINED AS SYNONOMOUS WITH - '
1,6A4,' FOR USE ',I3)
307  DO 310 K=1,6
310  DICT(K,IDD)=ADICT(K)
      IF (ACT.EQ.COM) ICOM(IDD)=1
      NODE(NUS1,IDD)=-IABS(NODE(NUS1,IDLO))
      IF (NUS2.EQ.0) GO TO 150
      CALL PSEARM(24,SYN2,DICT,6,NLIST,NSYN,NPTR)
      IDLO=NPTR(NSYN)
      IF (NSYN.NE.0.AND.NODE(NUS2,IDLO).NE.0) GO TO 315
      WRITE(6,12) SYN2,ADICT,NUS2
      GO TO 320
315  IDLO=NPTR(NSYN)
      IF (NLOC.NE.0) WRITE(6,14) ADICT,SYN2,NUS2
      NODE(NUS2,IDD)=-IABS(NODE(NUS2,IDLO))
320  GO TO 150
325  WRITE(6,13)ADICT
13  FORMAT('OND USE SPECIFIED FOR ',6A4)
      GO TO 150
400  NLIST=ID
C
C      FINAL SORT OF DICTIONARY
C
      CALL PSORGM(24,DICT,6,NLIST,NPTR,NPTR)
      DO 140 I=1,NLIST
      IDD=NPTR(I)
      DO 135 K=1,6
      IF (DICT(K,IDD).NE.BLANK) GO TO 141
135  CONTINUE
140  CONTINUE
141  NL=I
      NL1=NL+1

```

```

C
C      CHECK ORDER OF DICTIONARY
C
      DO 145 I=NL,NLIST
      I1=NPTR(I-1)
      I2=NPTR(I)
      IF(ICOMPA(DICT(I,I1),DICT(I,I2),24)) 145,142,142
142  WRITE(6,32) (DICT(K,I1),K=1,6)
      32  FORMAT('ODUAL OR UNORDERED ENTRY',6A4)
145  CONTINUE
      NLIS=NLIST-NL+1
      FTIM=FLOAT(NTIM)/360000.
      WRITE(6,18) DATE,FTIM,NLIS,USE
19  FORMAT('MASTER DICTIONARY LISTING DATED ',A8,' AT ',F9.3,' HOURS'
2,' CONTAINS ',I6,' ENTRIES'
1/'O',4X,'WORD',18X,'COM',3X,A8,5X,A8,5X,A8,5X,A8,/' ')
C
C      WRITE OUT FORWARD DICTIONARY
C
      WRITE(7) NLIS, DATE, FTIM, ((DICT(J,NPTR(I)),J=1,6),ICOM(NPTR(I))
1, (NODE(J,NPTR(I)),J=1,4),I=NL,NLIST),XLAST,WORD
      WRITE(6,30) ((DICT(J,NPTR(I)),J=1,6),ICOM(NPTR(I)),(NODE(J,NPTR(I)
1),J=1,4),I=NL,NLIST)
30  FORMAT(1X,6A4,14,1X,112,1X,112,1X,112,1X,112)
      ID=0
C
C      FORM REVERSE DICTIONARY
C
      DO 420 I=NL,NLIST
      IL=NPTR(I)
      IMNLP1 = I - NL + 1
      DO 410 K=1,4
      IF(NODE(K,IL).LE.0)GO TO 410
      ID=ID+1
      ICOM(ID) = IMNLP1
      NRNODE(ID)=NODE(K,IL)
410  CONTINUE
420  CONTINUE
      NRLIS=ID
C
C      SORT REVERSE DICTIONARY
C
      CALL PSORGM(4,NRNODE,1,NRLIS,LPTR,MPTR)
      DO 165 I=2,NRLIS
      I1=LPTR(I-1)
      I2=LPTR(I)
      IF(NRNODE(I1).LT.NRNODE(I2)) GO TO 165
      WRITE(6,34) NRNODE(I1),ICOM(I1)
34  FORMAT('OERRONEOUS ENTRY',112,3X,112)
165  CONTINUE
      ILAST=-99999999
      WRITE(6,16) NRLIS
16  FORMAT('OINVERSE DICTIONARY CONTAINS ',I6,' ENTRIES')
C
C      WRITE OUT REVERSE DICTIONARY
C
      WRITE(7) NRLIS, (NRNODE(LPTR(I)), ICOM(LPTR(I)), I=1,NRLIS), ILAST
      END FILE 7
      REWIND 7
      RETURN
9600 WRITE(6,9601) MLIST

```

```
9601 FORMAT('DICTIONARY ERROR, THE FORWARD DICTIONARY IS TOO LONG OR T  
100 SHORT, MLIST = ',I5)  
      MLIST = MLIST  
      RETURN  
      END
```

```

COMMENT  PSORGM - MULTIPLE-ITEM POINTER SORT BY MERGING ROUTINE
C        NUMBER OF COMPARISONS=N*LOG(N)/LOG(2)
C        NBYTES=NUMBER OF BYTES FOR SORTKEY LE 4*NDL1
C        LIST=DOUBLY DIMENSIONED ARRAY TO BE SORTED
C        NDL1=FIRST DIMENSION OF LIST
C        NPTR=POINTER ARRAY , DIMENSION GE NDL2
C        LPTR=WORKING ARRAY , DIMENSION GE NDL2
C *****
SUBROUTINE PSORGM(NBYTES,LIST,NDL1,NDL2,NPTR,LPTR)
  DIMENSION LIST(NDL1,NDL2),NPTR(NDL2),LPTR(NDL2)
COMMENT  CHECK INITIAL ORDER
  I1=NPTR(1)
  IF(I1.LT.1.OR.I1.GT.NDL2) GO TO 85
  DO 80 I=2,NDL2
    I2=NPTR(I)
    IF(I1.EQ.I2) GO TO 85
    IF(I2.LT.1.OR.I2.GT.NDL2) GO TO 85
    IF(ICOMPA(LIST(1,I1),LIST(1,I2),NBYTES))79,79,85
  79 I1=I2
  80 CONTINUE
  GO TO 300
C *****
COMMENT  SET UP POINTER ARRAY
  85 DO 90 I=1,NDL2
    NPTR(I)=I
  90 CONTINUE
C *****
  L2I=1
  DO 200 I=1,20
    M=1
    L2IH=L2I
    L2I=2*L2I
    JUP=NDL2/L2I+1
    DO 190 J=1,JUP
      N=M+L2IH
      IF(N.GT.NDL2) GO TO 190
      KLO=M
      KUP=MIN0(KLO+L2I-1,NDL2)
      MUP=KLO+L2IH-1
      DO 160 K=KLO,KUP
        IF(N.GT.KUP) GO TO 130
        IF(M.GT.MUP) GO TO 140
        IF(ICOMPA(LIST(1,NPTR(M)),LIST(1,NPTR(N)),NBYTES))130,130,140
      130 NL=M
        M=M+1
        GO TO 150
      140 NL=N
        N=N+1
      150 LPTR(K)=NPTR(NL)
      160 CONTINUE
      170 DO 180 K=KLO,KUP
        NPTR(K)=LPTR(K)
      180 CONTINUE
      M=KLO+L2I
      190 CONTINUE
      IF(L2I.GE.NDL2) GO TO 300
  200 CONTINUE
  300 RETURN
  END

```


SECTION V

SAMPLE INPUT DATA

The sample input deck is for a problem to be run on the Data Set Edit Program, including STCLØT, the skeletal ALLIØ program and the service subroutine, ITIM, all shown in the program listings and compiled on the Fortran H, OPT=2 compiler and the Assembler of Version 13. The program load module is stored as A61670AAB.H99LMOUS (TEST1) as indicated on the JOBLIB DD card.

The input deck is intended to illustrate a wide variety of control cards and therefore the actual actions taken are somewhat nonsensical. The cards indicating column numbers are not part of the deck setup and should be removed.

SAMPLE INPUT DECK FOR SKELETAL ALLIØ PROGRAM SHOWN IN LISTINGS

```
//SADJED00 JOB (5375,3,0,5,140),DUNPHEY,MSGLEVEL=1
//*SETUP DDNAME=FT04F001,DEVICE=2400-9,ID=(POOL,RING,NOSAVE,BLP)
//JOBLIB DD UNIT=2321,VOLUME=SER=DC0003,DSNAME=A61670AAB.H99LMOUS, 1
//      DISP=(OLD,KEEP)
//J EXEC PGM=TEST1
//GO.FT04F001 DD DSNAME=DATA1G,UNIT=TAPE9,DISP=(OLD,KEEP),LABEL=(,BLP),1
//      DCB=(RECFM=FB,LRECL=133,BLKSIZE=532,BUFNO=2), 2
//      VOLUME=SER=POOL
//FT06F001 DD SYSOUT=A
//GO.FT15F001 DD DSNAME=HOLD.PATHDICT,VOLUME=SER=DC0011, 1
//      UNIT=2321,  X
//      DISP=(OLD,KEEP),SPACE=(CYL,(10,2),RLSE), 3
//      DCB=(RECFM=VB,LRECL=1992,BLKSIZE=2000,BUFNO=2)
//GO.FT11F001 DD DSNAME=&TEMSCR1,UNIT=2311,VOLUME=SER=555555, 1
//      DCB=(RECFM=VB,LRECL=362,BLKSIZE=3624,BUFNO=2), 2
//      SPACE=(TRK,(20,10)),DISP=(NEW,DELETE)
//GO.FT12F001 DD DSNAME=&TEMSCR2,UNIT=2311,VOLUME=SER=666666, 1
//      DCB=(RECFM=VB,LRECL=362,BLKSIZE=3624,BUFNO=2), 2
//      SPACE=(TRK,(20,10)),DISP=(NEW,DELETE)
//GO.FT13F001 DD DSNAME=&TEMSCR3,UNIT=2311,VOLUME=SER=666666, 1
//      DCB=(RECFM=VB,LRECL=362,BLKSIZE=3624,BUFNO=2), 2
//      SPACE=(TRK,(20,10)),DISP=(NEW,DELETE)
//GO.FT16F001 DD SYSOUT=A,DCB=(RECFM=UA,BLKSIZE=133)
//GO.SYSABEND DD SYSOUT=A
//GO.FT05F001 DD *
```

```

***** COLUMN NUMBER *****
*
1 2 3 4 5 6 7 8
123456789012345678901234567890123456789012345678901234567890
*
GEN DATA EDITOR 0999999 0 4 SUBJ SUB1 SUB2 SUB3 SUB4
START TWO 1 2
UPDATE SUB1 1 2
11
12
13
14
999999
LIST LOCATORS
START ONE 3
UPDATE SUB2 2 2
11
12
14
15
999999
LIST LOCATORS
CORRECT SUB2 3 1 1
11
12
14
999999
DELETE SUBJ 1 2
13
999999
DELETE SUB1 2 3
15
999999
ERASE SUBJECT 1 2 8 12
LIST SUBJ
START ONE
POOL 2 4 1 5 9 11 15
LIST SUB1
LIST SUB2
LIST SUBJ
LIST OMISSIONS 4 12
LIST SUBJ 11
LIST SUBJ 12
LIST SUBJ 13
LIST OMISSIONS
LIST OMISSIONS 1
END REQUESTS
/*

```