AD-A081062

FOR OFFICIAL USE ONLY

LIBRARY
TECHNICAL REPORT SECTION
NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA 93940

Center for Naval Analyses
an affiliate of the University of Rochester

Research Contribution 213

Programmer's Guide to the NARF Workload Planning and Budgeting Model

Institute of Naval Studies

FOR OFFICIAL USE ONLY

02 021300.00

Center for Naval Analyses
an affiliate of the University of Rochester

1401 Wilson Boulevard Arlington, Virginia 22209 703/524-9400

An Equal Opportunity Employer

1 Oct 1973

MEMORANDUM FOR DISTRIBUTION LIST

Subj: Center for Naval Analyses Research Contribution 213; forwarding of

Encl: (1) CRC 213, "Programmer's Guide to the NARF Workload Planning and Budgeting Model," Jeffrey B. Birch, USN and Ralph D. Halford, For Official Use Only, August 1973

1. Enclosure (1) is forwarded as a matter of possible interest. The co-author is one of a select number of naval officers and enlisted men with advanced degrees who, through a unique contract provision, is assigned to CNA for a normal tour of shore duty to participate in the Navy Study Program.

2. This Research Contribution is intended for use by computer programmers. The description of the model is contained in INS Study 38. Uses of the model are discussed in CNA Research Contribution 212.

3. Research Contributions are distributed for their potential value in other studies and analyses. They do not necessarily represent the opinion of the Department of the Navy.

4. This Research Contribution is not approved for public release.

Herschel Kanter

HERSCHEL E. KANTER
Director
Institute of Naval Studies

Distribution List:
Reverse page

Subj: Center for Naval Analyses Research Contribution 213; forwarding of

---

DISTRIBUTION LIST

Naval Electronics Laboratory Center (2)
Naval Air Systems Command (AIR-04)(3) & (AIR-414A2)(2)
Naval War College
Naval Postgraduate School
U. S. Naval Academy
Director of Naval History (Op-09BH)
Head, Aircraft Maintenance & Material Branch (Op-514)

CENTER FOR NAVAL ANALYSES
RESEARCH CONTRIBUTION 213

Institute of Naval Studies

PROGRAMMER'S GUIDE TO THE NARF WORKLOAD PLANNING AND BUDGETING MODEL

August 1973

Jeffrey B. Birch
Ralph D. Halford

This Research Contribution does not necessarily represent
the opinion of the Department of the Navy.

Work conducted under contract N00014—68—A—0091

PARTICIPANTS

Jeffrey B. Birch, U.S.N.
Ralph D. Halford
Richmond M. Lloyd
Nancy Spruill
Ralph J. Vanni, Cdr. U.S.N.

ABSTRACT

This guide presents a detailed description of the computer programs constituting the Naval Aircraft Rework Facility (NARF) Workload Planning and Budgeting Model. As the guide is intended for use by programmers in making detailed changes to program coding, coding receives especial attention in the form of lines-by-lines description of main program listings. A general description of each program, the program listings, and flow charts are included.

The description of the model is contained in the Center for Naval Analyses' INS Study 38, "Naval Aircraft Rework Facility Study." A discussion of the model's uses is contained in CNA Research Contribution 212, the "User's Guide to the NARF Workload Planning and Budgeting Model."

TABLE OF CONTENTS

Preface v

Chapter I – INPUT1 1
Resolution of incompatibilities 1
Program logic 3

Chapter II – INPUT2 5
Introduction 5
Program description 5
Program logic 6

Chapter III – Matrix Generator 8
Introduction 8
Matrix file format 8
Data base file 10
Physical record 10
Logical records 13
Definitions of arrays and variables 13
Common 13
Arrays, integer 13
Arrays, real*8 14
Variables, integer 15
Variables, real*8 16
Localized variables, real*8 17
Discussion of program code 17
Subroutine MOVE (A,B,C,D,E) 17
Function LJABF (A,B) 17
Function NTOI(I) 17
Subroutine SEARCH (IVAL,NN) 18
Subroutine CONVRT 18
Main 19

Chapter IV – Report Generator 35
Introduction 35
Definitions of arrays and variables 35
Common/DICT/NDR,IDICT (2,400) 35
Arrays, integer 35
Arrays, real*8 36
Variables, real*8 38
Variables, integer 38
L.P. output format 39
Retrieving L.P. output 41
Discussion of program code 42
Function IBYTE (A,B) 42
Subroutine SEARCH (ITEC,NN) 42
Subroutine CVADJ 42
Main 43

TABLE OF CONTENTS (Cont'd)

References 51

Appendix A – Program Listings A-1

Appendix B – Format of Records in Input and Data Base Files B-1

PREFACE

Navy managers need the capability to analyze the effects of changes in the forces on the support system. Thus research efforts in the development of methodologies for measuring the support required for the operating forces was requested. One of these efforts was the development of a model suitable for the long range planning and budgeting of the Navy's Aeronautical Depot Maintenance Program.

This effort has required a heavy emphasis on methodological issues and the development of the computer system, software, and programs to implement the model. The results of this effort are summarized in three documents:

1. Study report (reference (2))
2. User's guide to the model (reference (3))
3. Programmer's guide

The initial findings of the study are described in the study report. This report is intended for those concerned with the overall problems of planning depot maintenance. The scope of the Depot Maintenance Program, the current planning system used for programming and budgeting, and several major policy issues are discussed initially. This is followed by a general and non-technical explanation of the model along with a case analysis for FY 1974 illustrating the model's uses. This initial version of the model has since been revised and expanded in cooperation with, and at the request of, the users. The User's Guide documents the procedures necessary to execute the latest version of the model. Finally, the guide which follows documents the detailed program listings in order to facilitate future revisions to the model.

The model developed (figure 1) consists of the following computer programs:

Input programs
Matrix generator
Mathematical Programming System (MPSX360)
Report generator

The programmer's guide will discuss the input programs, the matrix generator and the report generator. The Mathematical Programming System (MPSX360) is an IBM developed program for solving linear programming problems on an IBM 360 computer. MPSX360 utilizes the Matrix Generator output tape as input to solve the problem specified. The output of MPSX360, L.P. output, consists of the problem solution in matrix form and is used as input to the report generator. (For details on the MPSX360 system, see references (4), (5), and (6), obtainable from IBM.)

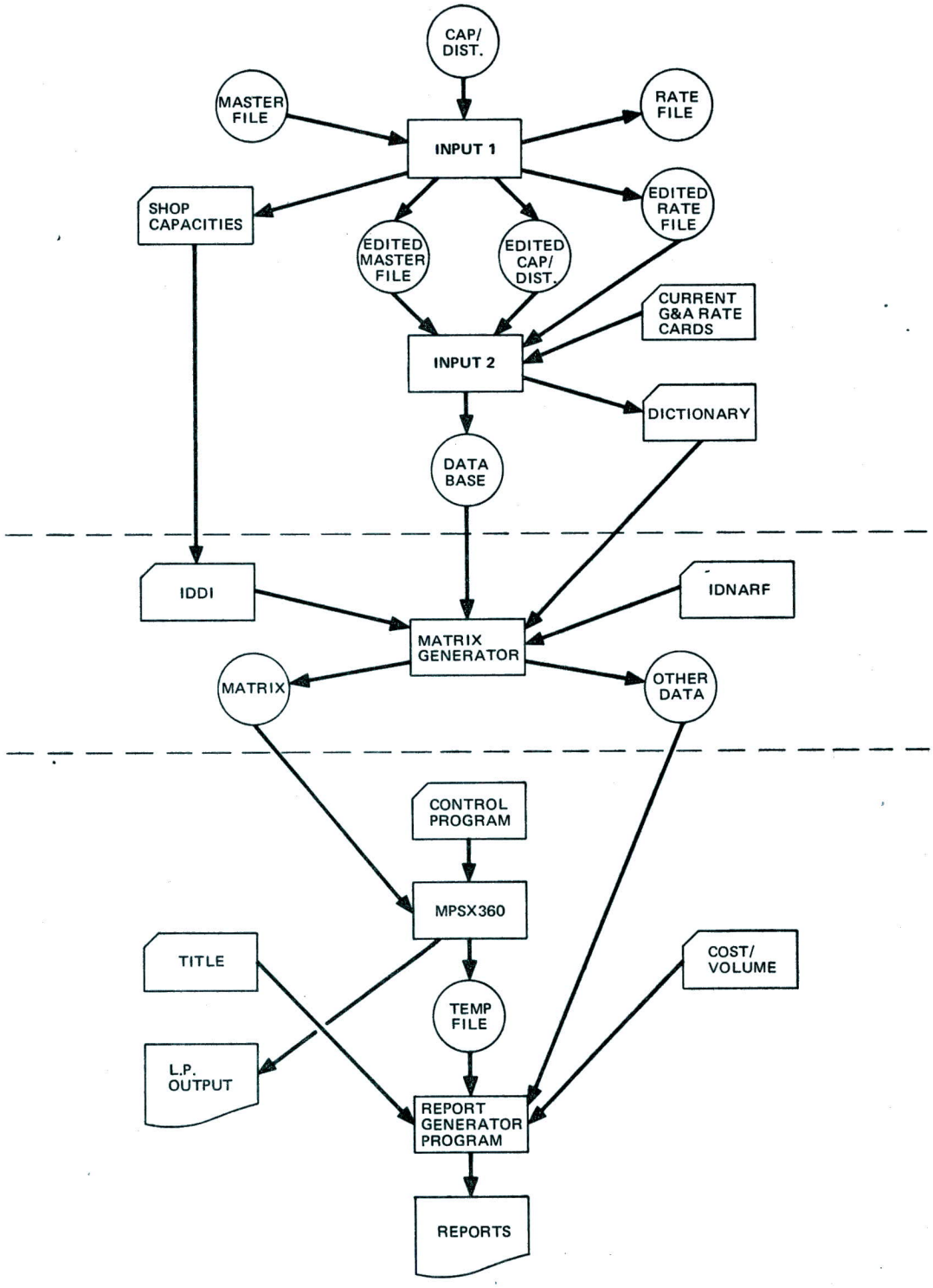


FIG. 1: FLOW CHART OF THE PLANNING MODEL

CHAPTER I

INPUT1

RESOLUTION OF INCOMPATIBILITIES

INPUT1 is designed to accept three input tapes (the Master File, the Capacity and Distribution File, and the Cost Rate File) created on an RCA 3301 computer system and to produce three output tapes that are in a format acceptable to the IBM 360 computer system. Certain discrepancies exist between the two computer systems which must be eliminated to facilitate the generation of an input tape. The techniques that INPUT1 uses to eliminate these discrepancies are explained in the following paragraphs.

The RCA 3301 is an octal machine with byte configuration of 6 bits/byte. It produces seven-track, odd-parity tapes. The IBM 360 system is a hexadecimal machine with byte configuration of 8 bits/byte. The standard IBM tapes are nine-track odd parity. Seven-track tapes can be handled by the IBM system by using proper JCL parameters (see page 25 of the User's Guide).

Specifically, to handle seven-track, odd-parity tapes the "tape recording technique" (TRTCH) subparameter of the "data control block" (DCB) parameter of the JCL for INPUT1 is set equal to translate ("T"). Under this translate option not all RCA characters are translated to the same character in IBM code. That is, 30 of the 64 RCA character codes are translated to IBM characters that are different from the original RCA characters. For example, an RCA "A" is translated to a "/" in IBM code; similarly, an RCA "B" is translated to an IBM "S". A complete list of these 30 RCA characters and their IBM translations is given in table 1.

INPUT1 handles this problem by using the COBOL "TRANSFORM" verb to change these IBM characters to the proper IBM character code to represent the original RCA characters. For example, the COBOL statement

```
TRANSFORM WORK-AREA FROM
'/S' to 'AB'
```

will replace all characters in WORK-AREA equal to '/' or 'S' with 'A' or 'B' respectively. Therefore, the 30 characters in table 1 are translated (by TRTCH = T in the JCL) to certain IBM characters and these characters are then transformed (by COBOL TRANSFORM verb) to the IBM format representing the original RCA characters.

INPUT1, in addition, compensates for incompatibilities in the tape formats. These incompatibilities are:

Labels and end-of-file mark (EOF): Each RCA tape begins with a 31-character label followed by a 3-character EOF mark. Neither the label nor the EOF mark is recognized as such by the IBM 360 system. Instead, the IBM system treats the label and the EOF mark as 31-character and 3-character records, respectively.

TABLE 1

RCA AND IBM CHARACTER DIFFERENCES USING TRTCH = T

From Octal	RCA	Machine code	To IBM character
61	/	B A 1	A
62	S	B A 2	B
63	T	B A 2 1	C
64	U	B A 4	D
65	V	B A 4 1	E
66	W	B A 4 2	F
67	X	B A 4 2 1	G
70	Y	B A 8	H
71	Z	B A 8 1	I
75	.	A 8 2 1	,
21	A	A 1	/
22	B	A 2	S
23	C	A 2 1	T
24	D	A 4	U
25	E	A 4 1	V
26	F	A 4 2	W
27	G	A 4 2 1	X
30	H	A 8	Y
31	I	A 8 1	Z
73	,	B A 8 2 1	.
00	0		space
12	space	8 2	0
55	>	B 8 4 1	)
56	<	B 8 4 2	;
60	" ','	B A	&
76	=	B A 8 4 2	+
16	)	8 4 2	=
20	&	A	: 2 8
32	+	A 8 2	0 2 8
52	['('	B 8 2	11 0
56	EF	B 8 4 2	;
72	EB	B A 8 2	12 0

Beginning and ending block marks: The first character of each physical record (block) of data written on a RCA 3301 tape drive is called an "item separator." The last character of each block is an end-of-block mark. Therefore, each block contains two extra characters. For example, if 1,000 characters were to be written on tape as one block of data, the actual physical record would consist of 1,002 characters, including the item separator and the end-of-block mark as the first and last characters in the block, respectively.

Short blocks: On the RCA 3301 system, it is possible that the final block of a tape file is shorter than the preceding blocks. This situation occurs when the number of remaining logical records to be written is less than the blocking factor that had been used throughout the file. For example, suppose there are 105 logical records, each 100 characters long, to be written on a tape file and the blocking factor is 10 records per block. The entire file would consist of 10 full blocks and 1 short block. The first 10 blocks would consist of 10 records or 1,002 characters each. The final block would consist of 5 records or 502 characters. The final block in a tape file is followed by a 21-character end-of-file label.

The above problems are resolved within INPUT1. In the file description paragraph (FD) for each tape file the recording mode is set equal to unspecified ('U'). This allows for records of varying lengths to be fetched from the tape file for each COBOL read operation. Thus, the 31-character label and 3-character EOF mark are treated as blocks of data by INPUT1.

The label and EOF records are found by checking for label and EOF identifiers in the procedure division of the program for each tape file. Once they are found, INPUT1 assumes that actual data follows. The first and last characters of each physical record are ignored by INPUT1. Finally, since the IBM 360 system does not recognize the RCA 3301 end-of-file, INPUT1 checks for the end-of-file by first checking for a short block. Whenever a short block is found it is assumed that it is the last block in the file. However, it is possible for the last block to be full. In this case, the end-of-file is recognized when the end-of-file label is identified.

PROGRAM LOGIC

The general logic used to handle each tape file is as follows:

1. Process the beginning label and EOF mark.
2. For each block of data, transform the data to the proper IBM character code representation of the original RCA characters. This is accomplished one record at a time by using the COBOL 'TRANSFORM' verb.
3. Write the transformed record onto a disk file. Check for a short length block by comparing the number of records in the block to the blocking factor of the file. If a short block is enountered it is assumed to be the last block in the file.
4. Return to step 2 until either a short block is encountered or the end-of-file label is processed. In both cases, the files are closed and step 5 is performed.

5. Sort the output disk file and write the results onto a tape file. This tape file will be a standard IBM formated, nine-track tape and will be used as input to INPUT2. The sort is accomplished by using the IBM sort utility and it is set up in the Job Control Language.

An exception to the above steps occurs during the processing of the Capacity and Distribution file. Here, during step 3, each record is checked to determine whether it is a capacity record or a distribution record. The distribution records are written on a disk file and later sorted as are the records in the Master and Cost Rate files. However, the capacity records are stored in a table until the end-of-file is reached. They are then punched onto cards to be used as card input to the Matrix Generator program.

CHAPTER II

INPUT2

INTRODUCTION

Computer program INPUT2 is a single COBOL program without subroutines. The function of INPUT2 is to create the Data Base tape file. This file is used as input to the Matrix Generator program. The three output tapes from INPUT1 (Master, Distribution, and Rate files) are used as input to INPUT2. Tables B-1, B-2, and B-3 in appendix B present the format of the three. The tape format for the generated Data Base file is given in table B-4. (Table B-5 is a matrix which assigns the appropriate Fund Code for customers 1 through 9 and A through I depending on the subprogram and program.)

PROGRAM DESCRIPTION

After the execution of INPUT1, the three tape files (Master, Distribution, and Rate) are sorted in ascending sequence by identical fields. For each record on the Master file there exists corresponding distribution and rate records on the Distribution and Rate files, respectively. There are several exceptions to the above that will be mentioned later. Each master and rate record contains information for a five-year period and each distribution record applies to all of the five years. INPUT2 will then construct one data base record for each of the five years represented on a master record from the Master file.

Each of the five data base records representing a master record is a composite of information contained on the master record and the appropriate distribution and rate records. Not all variables contained in a data base record come from these three records. Some variables such as "total-cost," "requirements," and "fund-code" are computed by INPUT2 using other information contained on the records. Each data base record contains all of the information that is desired by the Matrix Generator program.

To construct the data base records it is necessary to match each master record to specific distribution and rate records. This matching is accomplished by constructing a distribution identifier and a rate identifier from variables on the master record and searching the appropriate tape files until it is determined whether or not a match exists.

Records on the Distribution file are identified by a seven-character code composed on the variables Type Equipment Code (TEC), Program, Subprogram, and Designated Repair Point (DRP). Records on the Rate file are identified by an eight-character code composed of the variables TEC, Program, Subprogram, Fund-Code, and DRP. There are several exceptions to the above. Records with programs on the Distribution file with values of 'F,' 'H,' and 'L' and records with programs on the Rate file with values of 'F,' 'H,' 'L,' 'P,' 'R,' 'V,' and 'Y' are not associated with a TEC and the TEC variable on these files is equal to "spaces." These records are grouped together at the beginning of the Distribution and Rate files but their information applies to any master record with its program equal to the above mentioned values. These exception records from the Distribution and Rate files are read into core by INPUT2 in the first phase of the program. Both of these tape files are now positioned so that the Master file can be easily matched against it.

PROGRAM LOGIC

The logic of the program is as follows:

1. Set initial values for arrays, counters, and subscripts.
2. Read current G&A rate cards.
3. Read exception records from the Distribution and Rate files into two tables in core.
4. Read one record off the Master file. Begin constructing the five data base records corresponding to this master record by utilizing certain master record variables. Set up distribution and rate record identifiers.
5. Compute the requirement and fund-code variable. The fund-code is determined by certain combinations of the customer, program and subprogram variables from the Master file. These combinations are given in table B-5. For example, if the customer, subprogram, and program variables from the master record have values 'A,' 'l,' and 'A', respectively, the value for the data base fund-code is found to be 'A.'
6. Get appropriate information from the Distribution file for the five data base records. The master record variable program is checked for the value 'F,' 'H,' or 'L.' If it is one of these values the distribution table in core is searched for the appropriate record. If it is not one of these values, the Distribution tape file is checked by first reading into another table all distribution records with TEC, program, and subprogram equal to the corresponding variables on the master record. The distribution record identifier, composed of the variables TEC, program, subprogram, and DRP, is then checked against this table to find the correct distribution record to be used with the master record in making the five data base records. Generally, there always exists a distribution record for every master record representing a NARF and there never exists a distribution record for a master record representing other than NARF's. All master records that do not have distribution records have their distribution identifiers displayed on the printer.
7. Get appropriate information from the Rate file for the five data base records. A technique similar to that described in 6 above is used to find the rate record for a corresponding master record. The rate table in core is searched for the rate record if the master record program variable is equal to an 'F,' 'H,' 'L,' 'P,' 'R,' 'V,' or 'Y.' For programs other than the above values the Rate tape file is searched in a similar manner as the Distribution file. Generally, there exists a rate record for each master record. If a match is not made, the rate identifier is displayed on the printer.
8. Compute the total-cost variable. The total-cost variable is computed for each of the five data base records using variables from the appropriate rate

record. If no rate record is found for the master record, the total-cost variable is set equal to zero for all five data base records.

9. Replace the TEC code with a data base/CNA code. The four-character TEC code is replaced by a unique two-character code called the data base/CNA code. This results in the eight-character rework activity code composed of TEC, program, subprogram, customer, and DRP to be reduced to a unique six-character code acceptable in length to the MPSX360 program (see page 52 of the User's Guide).

10. Write the five data base records on the Data Base tape file. The records are written in unblocked format and are each 96 characters in length.

11. Return to step 4. Master file records are processed one at a time until the end-of-file is encountered.

CHAPTER III

MATRIX GENERATOR

INTRODUCTION

The Matrix Generator program has the specific function of creating the matrix file to be used by the IBM Mathematical Programming System. This file is partially under the control of the user in that many of the elements are directly specified by the user. These inputs are discussed in the code descriptions which follows. It is assumed that the reader is familiar with the inputs. The basic structure of the file is fixed and is reflected in the organization of the program.

MATRIX FILE FORMAT

The matrix is a card or card image file with a fixed format. The six sections are identified on single cards and in the order specified in table 2. In the table, the columnar format of all card records within a section follows the identifier (the numbers in parentheses are card columns).

TABLE 2

MATRIX FORMAT

1. NAME (1-4) — User name (15-n) — $15 \leqslant n \leqslant 22$
2. ROWS (1-4)
   - (2) row tape — N, E, L or G
   - (5-m) row name — $5 \leqslant m \leqslant 12$
3. COLUMNS (1-7)
   - (5-m) column name — $5 \leqslant m \leqslant 12$
   - (15-n) row name — $15 \leqslant n \leqslant 22$
   - (25-36) value – right justified
4. RHS (1-3)
   - (5-m) right-hand-side name — $5 \leqslant m \leqslant 12$
   - (15-n) row name — $15 \leqslant n \leqslant 22$
   - (25-36) value – right justified
5. BOUNDS (1-6)
   - (2-3) LO or UP — Lower or upper bound
   - (5-m) bound name — $5 \leqslant m \leqslant 12$
   - (15-n) column name — $15 \leqslant n \leqslant 22$
   - (25-36) value – right justified
6. ENDATA (1-6)

Graphically, the matrix appears as shown in figure 2a. Columns and rows are identified at the top and left respectively and each contains up to eight consecutive non-blank characters. All values entered into the matrix are identified by a column and a row name. Positions which appear blank are the unreferenced elements and are assumed to be zero. All these values represent the left-hand side of the system of simultaneous equations for which the object row (N) cost is to be minimized. All non-zero right-hand side (column RIGHT) values complete the equations (E) and inequalities (L

										ID NARF's	SHIFTS			MANPOWER				T-	RHS
									1	CNA CODE FOR THE TEC (TMS)	U	V	W	H	L	G	K		R
									2		FACILITIES A THROUGH G								I
								COLUMNS	3	FUND SOURCE	SHOP 1 THROUGH 9			SHOP 0 OR 1 THROUGH 9					G
									4	YEAR	YEAR(S) IYA THROUGH IYB								H
									5	PROGRAM									T
									6	SUB PROGRAM									
									7	CUSTOMER									
									8	FACILITY									
								ROWS											
								1 2 3 4 5 6 7 8											
OB	COST									UNIT COSTS	INCREMENTAL MAN-POWER COST PER HOUR FOR ONE YEAR			HIRING AND LAYOFF COSTS PER HOUR FOR ONE YEAR				N	
ID'S	CNA CODE FOR THE TEC (TMS)		FUND SOURCE	YEAR	PROGRAM	SUB PROGRAM	CUSTOMER			111 1111 11 11111								E	ID REQUIREMENTS
SHIFTS	F	FACILITIES A THROUGH G	SHOP 1–9	YEAR(S) IYA THROUGH IYB						XXX XXXX XX XXXXX	–% –% –%	–% –% –%	–1 –1 –1					L	SHOP ADJUSTED CAPACITY
	S										+1 +1 +1							L	2nd SHIFT SHOP PERCENT CAPACITY
	T											+1 +1 +1						L	3rd SHIFT SHOP PERCENT CAPACITY
MANNING	M		0 OR 1–9							XXX XXXX XX XXXXX	1–% 1–% 1–%	1–% 1–% 1–%		–1 –1 –1	+1 +1 +1	–1 –1 –1	+1 +1 +1	E	NARF (OR SHOP) ADJUSTED MANNING
	N															+1 +1 +1		L	NARF (OR SHOP) HIRING PERCENT TOTAL MANNING
	P																+1 +1 +1	L	NARF (OR SHOP) LAYOFF PERCENT TOTAL MANNING

FIG. 2a: GRAPHIC MATRIX

and G) as identified by the row type. Corresponding figure 2b is used later for reference purposes in discussing the codes.

Figure 2c is an expansion of Blocks U thru Z of figure 2b at the element level for a particular NARF. Elements are written into the matrix by column starting with column H in year 1 and finishing with column K in year n, where n is the range of years requested for minimization. Only those elements above the horizontal line, which represent the range of years requested, are entered.

DATA BASE FILE

Physical Record

The physical record making up the data base file is 96 bytes in length using a fixed EBCDIC code.

TABLE 3

PHYSICAL RECORD FORMAT

		Columns	Description
			Identification
		1-2	CNA/Data Base code corresponding to the TEC
		3	Fund source (table look-up)
		4	Year (1 thru 5)
		5	Program
		6	Subprogram
		7	Customer
		8	NARF (or facility)
			Data Format
1-9	3.0	9-35	Shop percentage
10	4.0	36-39	TQS–total quantity serviced
11	4.0	40-43	TME–total mission essential
12	3.0	44-46	M/E–mission essential
13	5.0	47-51	NORM–average hours required
14	3.0	52-54	M/N–mission non-essential
15	6.2	55-60	REQ–requirement = f (TQS, TME, M/E, M/N)
16	4.2	61-64	DLR–direct labor rate
17	5.2	65-69	DMR–direct material rate
18	4.2	70-73	POR–production overhead rate
19	5.4	74-78	GAR–general and administrative rate
20	6.0	79-84	UMC–unit material cost
21	9.2	85-93	COST–total cost = f (DLR, DMR, POR, GAR, UMC)
		94-96	blank

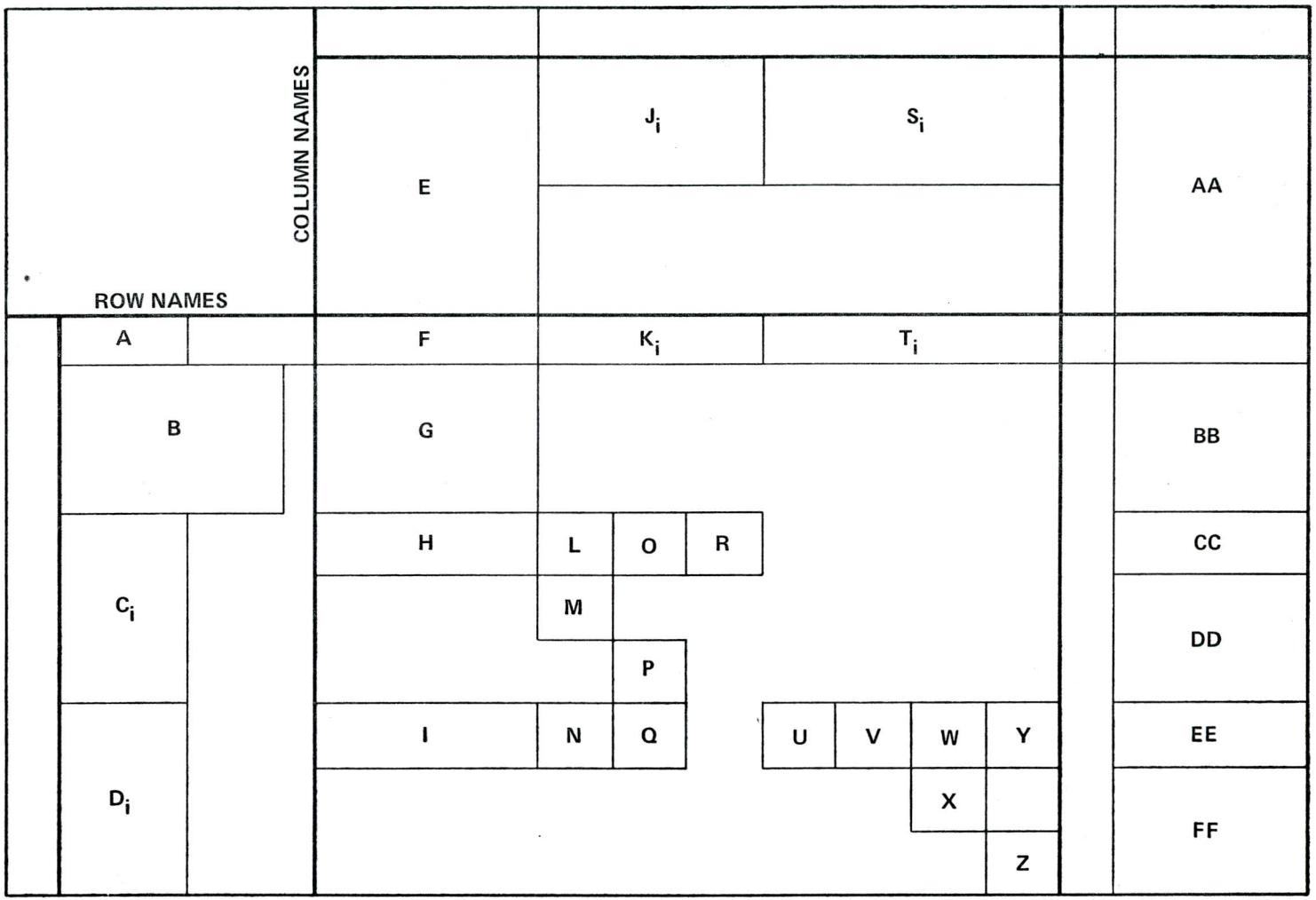


FIG. 2b: REFERENCE BLOCKS ON GRAPHIC MATRIX

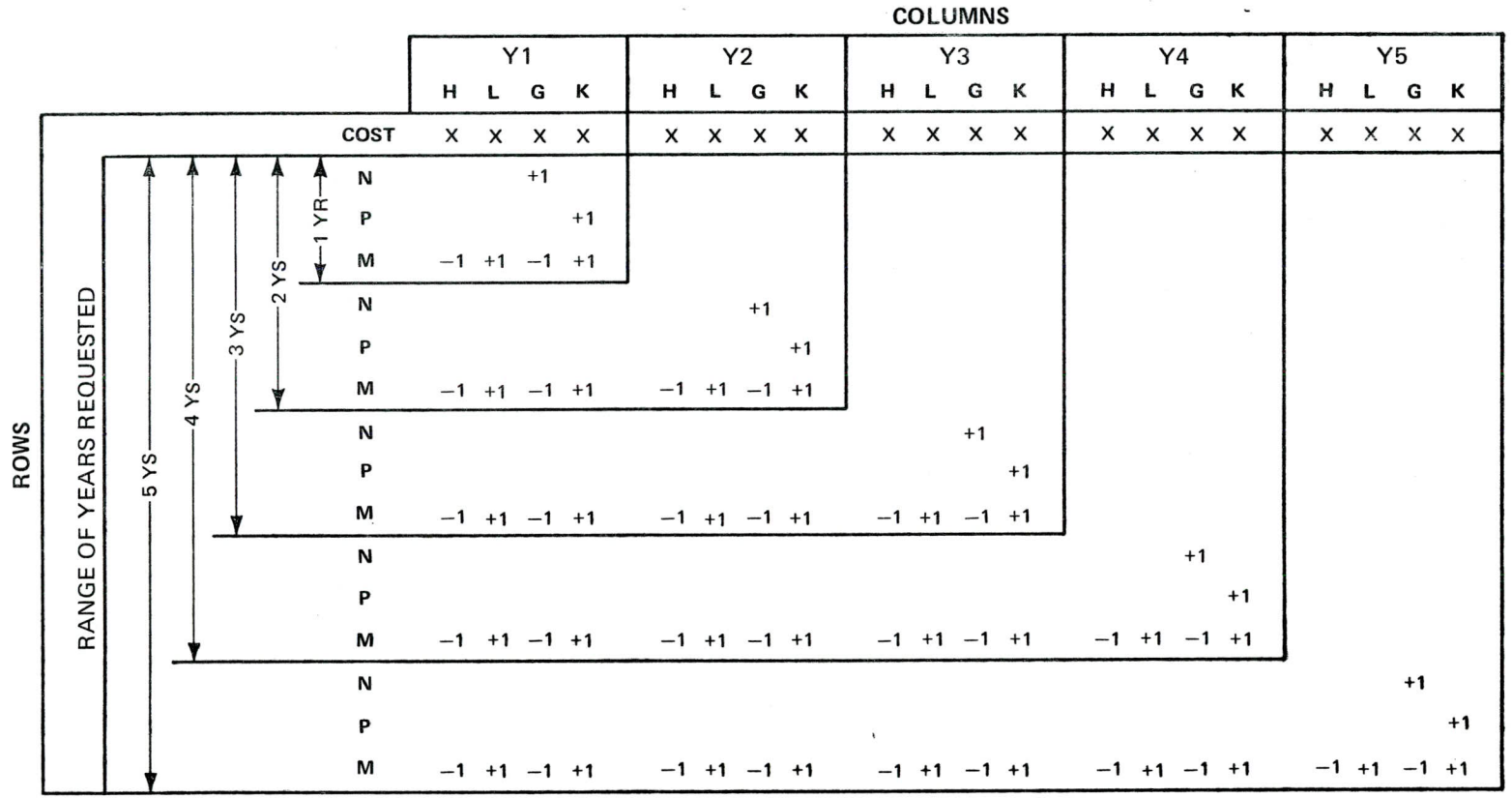


FIG. 2c: MATRIX MANPOWER ENTRIES IN REFERENCE BLOCKS U THROUGH Z, FOR ONE NARF AND OVER ALL YEARS REQUESTED

Logical Records

A logical record is composed of five consecutive physical records for the five years of data on the file. All bytes in the identification are identical except for the year which varies from one to five. The yearly identification is always given even though a particular year may not contain any data; that is, the data's worth is determined by data itself.

DEFINITIONS OF ARRAYS AND VARIABLES

Common

DICT used in the main routine and in subroutine SEARCH for conversion between the TEC and the CNA-generated code representing the TEC.

Arrays, Integer

IBNDA (2,100) Column name for exceptions to the rule for bounds percent variation.

IBNDC (100) Bound exception code for designating which set of bounds and whether the bound is upper and/or lower.

IBNDX (2,5) Name for standard bound percent variation. Up to five sets of bounds may be included.

IBOUND (2,2000) Name of the columns upon which the bounds are set.

ICD (64) Temporary area for data base update information (if any) as stated on IDN cards.

ICHK (21) Initially reset to zero. Appropriate element set to 1 when an IDN card has non-blank data in corresponding locations from ICD array.

ICLRW (2) Temporary storage for columns six thru 13 of user input control cards. Data is generally a column name, row name or an identification for a group of elements.

IDBREN (21) The name associated with the 21 elements from a data base record.

IDN (2,200) The names of all activities to be included in the optimization.

IDNA (2,5,12) A temporary location that is filled with consecutive activity names from array IDN, which have the same ID. IDNA (name, year, facility).

IDSAVE (2,1000) Save area for the name of each row under the heading IDs.

IFUNDS (8) Hollerith representation for each of the fund source codes.

IOTF (5) Hollerith representation for each of the five years.

IPROGS (10) Hollerith representation for each of the ten programs.

IRECA (2,5,12) A temporary location that is filled with the activity names of consecutive records with equivalents IDs as read from the data base.

IRIGHT (2,26,5) An identifier and up to 25 row names which will have their right-hand sides modified.

ITMP (2,5) A temporary location into which the activity name is read for the five years from the data base.

ITQS (5) Indicator during a data base update which notes that the TQS for all facilities has already been modified in the given year.

JDENT (38) An expanded array for holding data to be used during comparisons and output.

JDICT (4,400) A temporary location for the T/M/S as read from the Dictionary.

KASTID (2) Storage for the current ID representing the data base records under consideration.

LASTID (2) Storage for the current ID representing the requests for optimization under consideration.

MIXUP (3,100) The data base activity name and year which is to be updated.

NARFX (12) Hollerith representation for each of the 12 rework facilities.

NARFY (12,2) Hollerith representation for lower and upper row identification when parametric equations are produced.

NBAD (13) Temporary noting of requests for optimization when there is no corresponding data base record.

NBASE (13) Temporary noting of data base records for which there was no request for optimization.

NMATCH (13) Temporary noting of data base records for which a request for optimization was made.

NRIGHT (5) Quantity of right-hand side modifications for each of up to five requests.

Arrays, Real*8

BNDB (5,100) Up to 100 bound percent variation exceptions over five years.

BNDY (5,5) One to five sets of standard bound percent variations over five years.

BOUNDA (2) Temporary locations for computed values used as column-row elements when parametric equations are called for.

BOUNDS (2,2000) Save area for (1) the requirement as given in the data base activity record and (2) the sum of all requirements having the same ID and requested for optimization. Data is later used with bound percent variations to produce hard physical column bounds.

CAPB (5,9,7) Adjusted capacity of a NARF shop for a given year. CAPB (year, shop, NARF) = TOTCAP minus all Base Workloads.

CML (9,7,5) Current manning level in manhours for a NARF shop and spread over five years.

CMLA (9,7,5) Current manning level in manhours adjusted by the workload from data base records not requested for optimization.

COSTB (5,9,7,8) Costs and incremental costs for all shift and manpower changes.

EF2 (5) Efficiency factor for second shift.

EF3 (5) Efficiency factor for third shift.

FRIGHT (5,25,5) Parametric multipliers used to alter current values for the right-hand side of specified rows.

FIXUP (21,100) User supplied data to replace specific elements of corresponding data base records.

GAR (7,5) The general and administrative rate for each NARF over five years.

HCOST (4) Temporary array used for hire/layoff incremental cost computations.

HPY (7,5) Hours per year when multiplied by the CML (9,7,5) array, is the direct labor hours for each NARF shop over five years.

PCAP2 (5) A percentage applied to first shift total capacity to generate second shift total capacity. The percentage is applied to all NARFs for the particular year.

PCAP3 (5) Same as PCAP2 (5) but third shift total capacity is generated.

PHPY (5) Pay hours per year–currently not used by the program.

PVAR (5) Percent variations applied when parametric equations are used in place of column bounds.

RECB (21,5,12) (sec. IVa) A temporary array to hold the 21 numerical data base elements over five years. From one to 12 facilities that all have the same ID are put here at one time.

REQID (1000) The values associated with the name in array IDSAVE (2,1000) that became the right-hand side in the matrix under the heading IDs.

TEM (15) A temporary area for the data read from the initial user control cards.

TMP (21,5) A temporary area into which data base record values are stored before being moved to array RECB.

TOTCAP (5,9,7) The original total capacity of each NARF shop for five years.

SUMN (8,10,12,5) An initialized array where data base values not subjected to L.P. optimization are summed according to eight fund sources, ten programs, twelve facilities, and five years. The final array is passed to the program which produces reports.

SUMR (3) A temporary array for computing the limits of increased or decreased work force allowed before a cost for such changes is either first applied or increased. That is, Phase I limit.

VALU (21) The temporary area where values used to update data base record elements are initially stored.

Variables, Integer

IAY A user input for the actual first year (e.g. 75) in the range of years for which a solution is requested.

IBOP An indicator for the bounds (1) or parametric (2) user supplied option.

IDENT A 4-byte identifier for the users' input cards.

IEND Indicator for data base end-of-file.

IMCBB User indicator regarding initial hire/layoff cost input cards. 1 means that cards are not included and the cost is assumed to be zero. 2 means that cards are included and they contain the cost information.

IMOV Set during data base processing.
1 = move next set of L.P. requests to array IDNA
3 = move next set of data base records to arrays IRECA and RECB
2 – both 1 and 3

INOS User supplied indicator designating hire/layoff by shop or by NARF.

INC User indicator (1,2,3,4 or 5) for the number of cards following the first and containing yearly information.

ISET Switch used when reading the data base and moving record contents from ITMP and TMP to IRECA and RECB.

ISETR Localized switch for preventing error message duplication.

ISTOP Set to 1 upon occurrence of an error and periodically tested to stop job processing.

IUPLO Indicator on user request cards (column 15) which indicates how information on this card is to be used.

IWPR Total number of numeric data fields on a data base record.

IYA The first year (1,2,3,4 or 5) in the range of years to be subjected to optimization.

IYB The last year in the range of years to be subjected to optimization such that IYB $\geqslant$ IYA.

IYC The specific year the data on an input card represents.

JBDS Number of bound names and values saved in arrays IBOUND and BOUNDS.

$$0 \leqslant \text{JBDS} \leqslant 2000$$

JBNDS Number of bound exceptions for all sets of bounds requested.

$$0 \leqslant \text{JBNDS} \leqslant 100$$

JID Number of ID row names and right-hand side values saved in arrays IDSAVE and REQID.

$$0 \leqslant JID \leqslant 1000$$

JIDN Number of unique ID NARFs the user requests for optimization.

$$0 \leqslant JIDN \leqslant 2000$$

KC Data base record field for the computed total cost.

KG Data base record field for the general and administrative rate.

KR Data base record field for the computed requirement.

LUD Logical input unit from which the TEC–TMS dictionary is read.

LUI Logical input unit from which the data base is read.

LUØ Logical output unit to which the matrix is written.

LUQ Logical output unit to which the dictionary is written.

LUU Logical output unit to which the other data used in the report generator is written.

LUV Logical input unit from which the manning level capacity and cost data is read.

NBS Number of sets of bounds.

$$0 \leqslant NBS \leqslant 5$$

NFACIL=12 Number of Facilities.

NFUNDS=8 Number of Fund Sources.

NNARFS=7 Number of NARFs.

NPROGS=10 Number of Programs.

NRS Number of sets of parametrics.

$$0 \leqslant NRS \leqslant 5$$

NSHOPS=9 Number of shops per NARF.

NYEARS=5 Number of years of information on the data base.

All other integer variables are localized and follow the standard FORTRAN type conventions:

I	J	M
II	JJ	MM
III	JRHS	MMM
ICODE	K	MXX
IGOTD	KK	N
ITEMQ	KKK	NN
ITEMP	KIDN	NF
ITEMR	L	NP
IYD	LL	NXX
	LN	

Variables, Real*8

SUM Sum of all requirements for a particular ID that is requested for optimization.

XLML User supplied percentage applied to the current manning level. The product is the layoffs in manhours allowed before costs are either incurred or increased.

XUML User supplied percent applied to the current manning level. The product is the manhours of hiring allowed before costs are either incurred or increased.

Localized Variables, Real*8

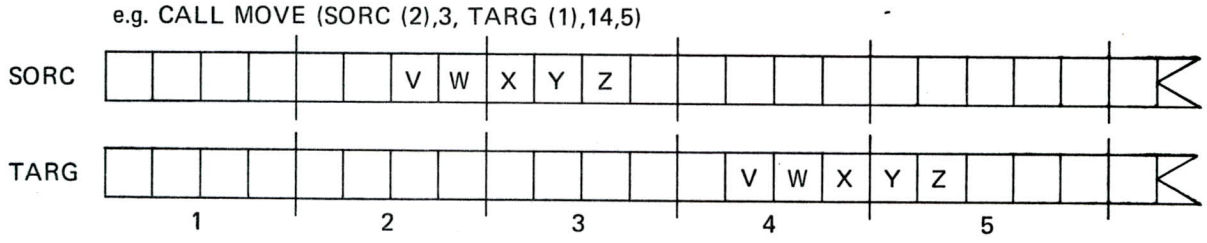
BVAL
PERCAP
PROD
TEMP

DISCUSSION OF PROGRAM CODE

This section first considers the subroutines and functions used by the main routine. Then the main program is discussed using the entries in the location field plus or minus a number of lines, and the blocks in figure 2b, as references.

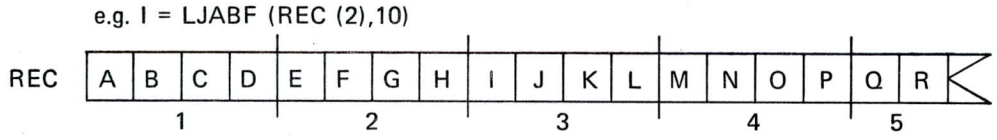
Subroutine MOVE (A,B,C,D,E)

This assembler language subroutine handles the movement of a string of bytes from one location to another. It makes use of the MVE instruction which limits the number of bytes to 256. The source string begins at the B^{th} byte of the A^{th} address. The target string begins at the D^{th} byte of the C^{th} address. E characters are moved where $1 \leqslant E \leqslant 256$.



Function LJABF (A,B)

This assembler language function is for isolating the B^{th} byte of the A^{th} address. The byte is left justified in Register 0 and blank filled on the right.



The content of I after returning from the function is $N_{\Delta\Delta\Delta}$

$$I = D5404040_{16}$$

Function NTOI(I)

This assembler language function moves bits 4 thru 7 of a word to bits 28 thru 31 of the same word, and then zeros bits 0 thru 27. The function is called under the assumption that a numeric byte resides left-justified in the word. The procedure converts the byte to an integer.

e.g. A seven is read under an A1 format

F7404040
000000F7
00000007

e.g. A blank is read under an A1 format

40404040
00000040
00000000

e.g. A V is read under an A1 format

E5404040
000000E5
00000005

No provision is made for data checking. This should be done before using the function.

Subroutine SEARCH (IVAL,NN)

The dictionary produced during creation of the data base was read partly into array IDICT in common block DICT. The number of dictionary records, IDR, is also maintained in the array and is assumed to be less than or equal to 400. Each record within IDICT is eight bytes long and contains a two byte CNA/Data Base assigned code (character position 1 and 2) to represent each four byte TEC code (character positions 5-8) for the type-model-series. The dictionary is required because the data base has the CNA-assigned code whereas the users will be supplying the TEC. Therefore, before data comparisons are made, a translation from the TEC to the CNA code is necessary. A conversion from the CNA code back to the user identifiable code is necessary when error messages are printed.

Since both sets of codes are in ascending order, a binary search is performed. NN is either the value one or two when entering the subroutine. If NN equals one, the CNA code portion of the dictionary (IDICT) is searched until a match with IVAL is found. If NN equals two, the TEC portion of the dictionary (IDICT) is searched until a match with IVAL is found. In both cases upon a match with IVAL, NN is set to the record number of the dictionary for use in the main program.

Subroutine CONVRT

Columns 17 thru 80 of a card used from file IDNARF contain either blank or numeric data. To distinguish between blank and zeros the 64 characters are first stored in array ICD using a 64A1 format. At entry point CONVRT 18 distinct floating point values are constructed in array VALU from the hollerith in ICD. Array ICHK is initialized to and remains zero when a field contains all blanks. Any non-blank entry forces one of the 21 positions of ICHK to one. Locations 15, 19 and 21 are not used. They are included to allow for alignment of data base elements with elements

from the cards to be used in updating the data base. The meaning and position of each field can be obtained from the user's manual.

Function NTOI converts the hollerith number to an integer so that an arithmetic operation can be performed. A blank becomes a zero. No check is made for a decimal point since none is expected. All data has an assumed decimal position and it is accounted for in the main routine.

At entry point BNDRHS five years of blank or numeric data is expected. The first five positions of arrays VALU and ICHK are used in the same manner described for entry CONVRT. The five data fields are always 17-24, 25-32, 33-40, 41-48, and 49-56.

Main

The program is initialized by reading the CNA/Data Base tape file. This constitutes establishing CNA codes for all TECs, sorting the hit parade*, incorporating option cards, establishing costs, and marrying constraints with the proper TEC. The row and column names for the matrix are established. The above information is then used to generate the matrix. (A flow chart of the program is shown in figure 3.)

Ø	Establish and initialize variables and arrays.
91–1	Read the first control card.
91+1	If the last year in the range IYA to IYB is not given, set IYB equal to IYA.
91+2	If variable INC on the first control card is not given, the default is 1. Print error message.
93	ISET is used to indicate that information about the first year is given.
93+1	Read INC data cards with yearly information. If IYC equals IYA then set ISET to 1 and test the data.
89	Move the card data to permanent appropriate storage areas.
99+1	If ISET is 1 then the required first year has been given.
97	The number of yearly data cards must be less than or equal to the number of years in the range from IYA to IYB. If equal, it is assumed that all years are accurately given.
103	Fill missing years with the previous year's data.
108	Percentage for allowable manning variance is converted to a decimal.
50	Read all cards that belong to the hit parade (IDN), bounds specifications (BNDS), right-hand side modifications (RHS), general and administrative rate changes (GARC), and last card (END). All IDENT except END must be stated on the first card and may thereafter be stated or left blank. A group of cards belonging to a given category, such as BNDS, is terminated only by recognition of a different but valid identifier such as END or GARC. An invalid IDENT causes an error message. An unestablished IDENT is a blank with no valid IDENT previously given.
52+1	Check for IDN.
52+2	Check for data on the IDN card.
53	Check for BNDS.

*Hit parade is the term used to define the TMSs/TECs to be considered by the planning model (see page 38 of the User's Guide).

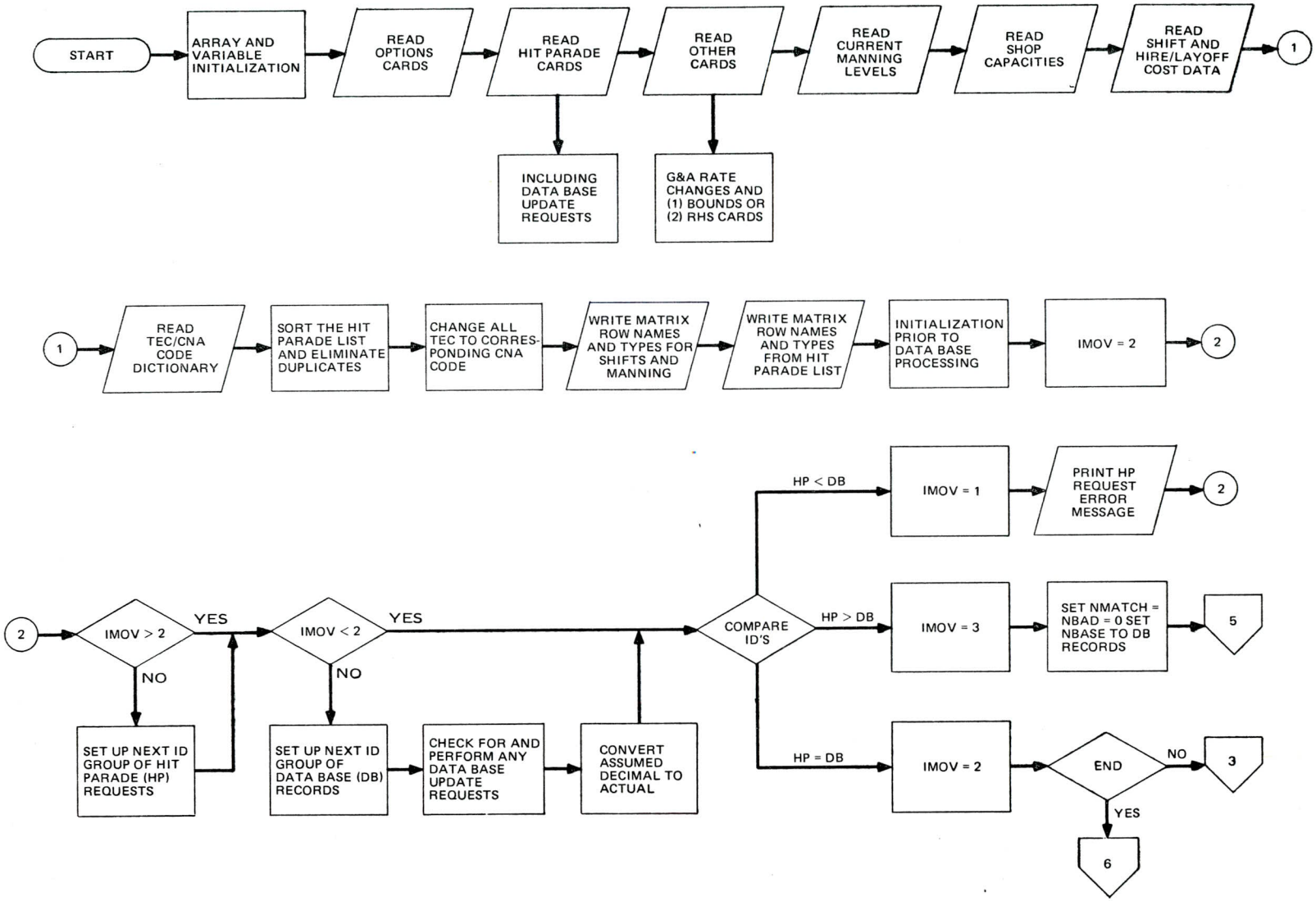


FIG. 3: MACRO FLOW CHART OF THE MATRIX GENERATOR PROGRAM (SHEET 1)

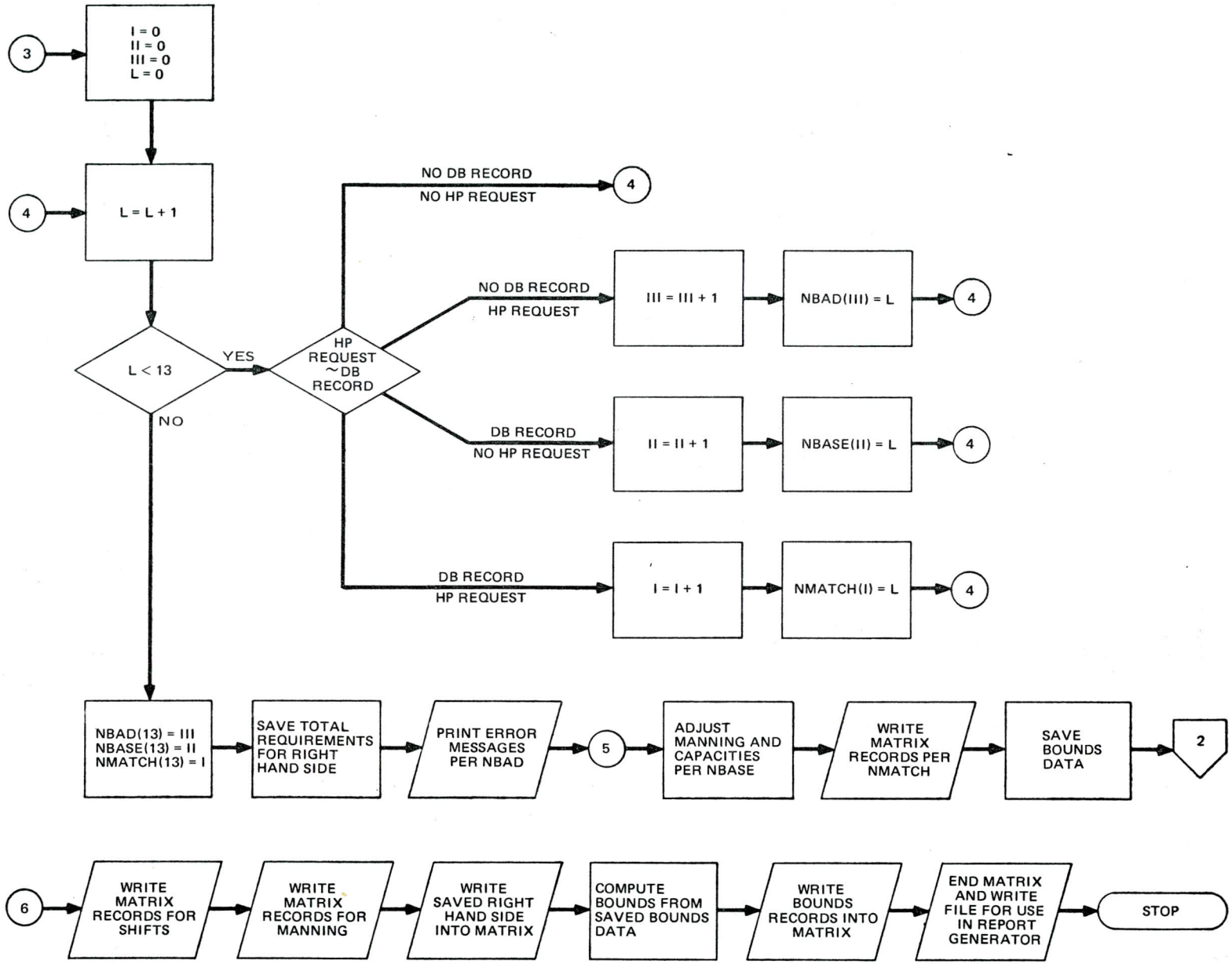


FIG. 3: MACRO FLOW CHART OF THE MATRIX GENERATOR PROGRAM (SHEET 2)

55 Check for a name on the BNDS card.

55+1 Save up to five sets of bound names and percent variation in IBNDX and BNDY.

54 Check for RHS.

57 Check for a name on the RHS card.

57+1 Save up to five sets of right-hand side modification names in IRIGHT.

56 Check for GARC.

69 Check for data on the GARC card.

69+1 Put general and administrative rate changes for the given NARF and all specified years into GAR.

58 Check for END.

58+1 Check for blank.

58+2 Print error message for invalid IDENT.

72 Go to location determined by ICODE or print unestablished IDENT message.

110 Under IDN (ICODE=1) check for code (blank, L, B or D) in variable IUPLO. If B or D go to 111 and don't include name ICLRW in hit parade list, B is a data base record modification and D is a record deletion.

114 Include the rework activity in the hit parade list.

114+3 If the code is a blank, no data base updates are included. Otherwise (L) updates are included.

111 Put data base updates into arrays MIXUP and FIXUP.

120 If the limit on number of sets of bound has already been reached, ignore this card. Otherwise, include this card's data with the bound exceptions in arrays IBNDA, IBNDC, and BNDB. No check is made for the maximum allowed.

130 If the limit on the number of sets of right-hand side modifications has already been reached, ignore this card. Otherwise, accept up to 25 modification statements and put the row names and data in arrays IRIGHT and FRIGHT.

150 The card or card image fixed length file called IDDI contains current manning levels (63 cards), shop capacities (63 cards), first shift manhour costs (63 cards), incremental second, third and post third shift manhour cost (3*63 cards), manhour hire and layoff costs resulting from manpower changes beyond a predetermined range (2*63 card) and optional specifications for hire and layoff costs resulting from manpower changes within the predetermined range (126 cards).

158 Read the current manning level and convert men to direct manhours.

156+1 Read shop capacities and spread all year n to year n+1 if year n+1 is not given. n=1,2,3,4.

155+1 Zero that portion of the cost array where initial hire and layoff costs are kept.

152+1 If IMCBB on the first control card was a 2, then initial hire/layoff costs are included in the IDDI file.

152+3 Read the cost cards in file IDDI into array COSTB and spread all year n to year n+1 if year n+1 is not given. n=1,2,3,4.

133 Check that the initial hire/layoff costs are always less than the costs outside the initial manpower hire/layoff range.

151+1 Modify the incremental costs for second and third shifts according to their efficiency factors and the cost of first shift operations.

134–1 The dictionary is read into common array IDICT and program array JDICT. A count NDR is also maintained in the common block. Array JDICT is used solely for reproducing the T/M/S on file LUQ. The first dictionary record is printed as an indication that file IDDI contained the correct number of cards when both the IDDI file and the dictionary are input from the card reader. The first two bytes of the first record in array IDICT must be AA.

170 The nature of the input under IDENT IDN allows for duplicate ID NARF specification. That is, when an ID NARF is to be included in the hit parade and data base changes are also requested for individual years, separate entries with the same ID NARF are put into the hit parade list. This section, using a bubble sort, orders the list and eliminates any duplicate entries.

160 Since the user specifies his code as a TEC and the data base has the CNA code, word 1 of each record in array is changed from the TEC to the CNA code. An invalid TEC is eliminated from the list. The dictionary and hit parade list are passed sequentially since both are in ascending order.

164+2 Compare the TEC in the dictionary with the TEC in the hit parade list.

163 Change the TEC to the CNA code.

162 Eliminate an invalid TEC from the hit parade list.

140 Three more arrays (IBNDA, IRIGHT, and MIXUP) contain the user specified TEC. These must also be changed to the CNA code. Subroutine Search is used to locate the matching TEC in the dictionary array.

143 Change the TEC in a bounds exception request to the corresponding CNA code.

147 Change the TEC in the right-hand side modification requests to the CNA code.

128 Change the TEC in the data base update request to its CNA code.

180 All previous inputs have now been prepared for creating the matrix. This card image file follows a very specific sequence of events under rigid format specifications (see the "Matrix File Format" section). The first section names the matrix and provides for naming and typing all rows. These rows are somewhat a function of user inputs. The options on the first control card provide for (a) the use of bound (IBOP=1) or the use of parametric equation (IBOP=2) and (b) manning changes at either the NARF level (INOS=1) or the shop level (INOS=2). Variable IBOP affects the rows generated in block B (figure 2b) and variable INOS affects the rows generated in block D of the matrix. Also affecting the rows is the number of years (IYA to IYB) for which optimization is requested. Excluding the COST row, a two year request will cause twice as many rows to be generated as a single year request.

187 NAME of the matrix is MINUM.

181+1 Identification that the following cards contain row names and types.

182+1 The objective function (type N) is called COST and is represented as block A.

183+1 Blocks C and D are produced next. The number of rows here is a function of the number of years. A minimum of 189 rows is created at statement number 186 for block C. Block D is also a function of variable INOS. When INOS=2, a row is established for each shop using the shop numbers 1 through 9. Otherwise INOS=1 and only one row is established for all shops using a zero shop number. Blocks C and D are produced together and will appear as shown in the following list depending on the value of INOS.

INOS=1		INOS=2	
FA11	Initial Rows	FA11	Initial Rows
SA11		SA11	
TA11		TA11	
MAØ1		MA11	
NAØ1		NA11	
PAØ1		PA11	
FA21	Repetition	FA21	Repetition
SA21		SA21	
TA21		TA21	
FA31		MA21	
SA31		NA21	
TA31		PA21	
.		.	
.		.	
.		.	
FA91		FA91	
SA91		SA91	
TA91		TA91	
FB11		MA91	
SB11		NA91	
TB11		PA91	
MBØ1		FB11	
NBØ1		SB11	
PBØ1		TB11	
FB21		MB11	
SB21		NB11	
TB21		PB11	
.		.	
.		.	
.		.	

190 Block B is dependent on the number of years and the variable IBOP. When IBOP=1, bounds are requested and one row is established for each unique ID from the hit parade list in array IDN. When IBOP=2, parametric equations are used to control assignments. In addition to the one row established for each unique ID, two more rows are created for each IDN in the hit parade list. Block A, for the different values of IBOP, will appear as follows when given partial example hit parade.

		ID NARF
Sample hit parade array IDN		$AMN_{\Delta}ABCB$
		$AMN_{\Delta}ABCE$
		$AMN_{\Delta}ABCI$
	IBOP=1	IBOP=2
Rows generated	$AMN_{\Delta}ABC_{\Delta}$	$AMN1ABC_{\Delta}$
		AMN1ABCN
		AMN1ABCZ
		AMN1ABCQ
		AMN1ABC2
		AMN1ABCU
		AMN1ABC6

The sample IDN list shows the TEC translated to the CNA code AM. The fund source N also came from the dictionary as did the blank which will be filled with the year. Program A, Subprogram B, and Customer C follow. These seven bytes make up the ID. The last bytes (B,E,I) are the NARFs. For a group of n ID NARFs with the same ID, one row is made when IBOP=1. Otherwise 2n+1 rows are created for that group.

The following list gives the bytes substituted for the facility in the two additional rows generated when parametric equations are used. The * mark the facilities used in the previous example.

		*			*				*			
Facility	A	B	C	D	E	F	G	H	I	J	K	L
Parametric Row	M	N	O	P	Q	R	S	T	U	V	W	X
Substitution	Y	Z	Ø	1	2	3	4	5	6	7	8	9

200–1 The heart of the program begins with identification of the columns section. An additional hit parade element of all 1s is tacked onto the end of the list to facilitate in the handling of remaining data base records after all hit parade requests have been processed. Several variables, local to data base processing while creating the column section of the matrix, are initialized. A header for the data base updates is also printed if any were requested.

The data base processing begins with variable IMOV being used to control the movement of groups of records containing the same ID. When IMOV equals 1 or 2, the next set of hit parade requests is moved from array IDN to appropriate locations in array IDNA. And, when IMOV equals 2 or 3, the next set of data base records are read and ultimately stored in arrays IRECA and RECB. Thus, the hit parade and the data base are passed sequentially from beginning to end with the next group of records that contain the same ID being considered. The ID representing each group is stored in LASTID for the current group of hit parade requests and in KASTID for the current group of data base records. Bytes 1,2,3,5,6 and 7 from LASTID and KASTID are compared to determine the next step. If LASTID is less than KASTID, a hit parade request was made for which no data base record exists. An error message for the ID is printed, IMOV is set to 1 and the next group of hit parade requests is moved to array IDNA. If LASTID is greater than KASTID, no hit parade request was made for all current data base records being considered. This data base information will be used to adjust the current manning level in array CMLA and the shop capacities in array CAPB. The final adjusted values will later become part of the matrix right-hand side. Values are also accumulated in array SUMN for use in the Report Generator. IMOV is set to 3 and the next group of data base records is input to arrays IRECA and IRECB. If LASTID and KASTID are equal, further comparison at the facility level is needed. That is, how does each hit parade request under the given ID compare to the records read from the data base? When a facility request exists for which there is no data base record, an error message for the ID NARF is printed. When a data base record exists for which no hit parade request was made, the data base record values are accumulated in array SUMN and used to adjust CMLA and CAPB. Finally, when the hit parade request exactly matches the data base record, entries will be written into the matrix at blocks E, F, G, H, and I shown in figure 2b. Block E is the eight-byte column name from array IDNA and for which an exact data base match occurred. The column for block E will compare with a previously established row in block B with the exception of byte number eight. A value of '1' will be entered

in block G linking the column to its corresponding row. Once a value has been entered for a given column, all values for that particular column must be entered sequentially. When a new column is identified, no more values can be entered for the previous column. The same column then has the cost entry (block F) from the data base entered at the COST row (block A). If the particular facility is a NARF (A thru G), entries are also made for blocks H and I. They are otherwise omitted since shift capacities and manning are not of concern in the other facilities (H thru L).

A sum of the requirements is found for all matches and the row name (block B) or ID along with the sum is saved for later entry into the right-hand side at block FF. Also, if bounds are to be used, the individual requirements for each match, the column name, and the sum of all requirements are saved for later computation of the bounds to be placed upon the columns. Otherwise parametric entries for the additional rows generated in the rows section are produced.

220+1 The next group of hit parade requests with the same ID is to be moved into the IDNA array which is first initialized to zero. Array LASTID is set equal to the first request from the group.

228+1 Array IDN is checked for total depletion.

222 This point is returned to from 219+1. It is jumped over during transfer of the first ID NARF from array IDN to array IDNA. That is because a group of hit parade requests must contain at least one element.

224 The numeric value for the facility (A=1, B=2, . . . L=12) is determined.

224+3 Array IDNA is filled from array IDN in all years requested (from IYA to IYB). The year in byte position four is filled in with the call to MOVE.

219 The current location KIDN in array IDN is incremented and a jump to 222 is taken.

230 IMOV is tested to determine whether the next group of data records with the same ID is to be moved to arrays IRECA and RECB. If it is, arrays IRECA and RECB are first initialized to zero.

231 Variable ISET is checked. It is used to control the reading of data base records into arrays ITMP and TMP, and then the movement of the record from ITMP and TMP to appropriate position in arrays IRECA and RECB. ISET=1 indicates the beginning of the data base. ISET then switches between zero and two.

232 Five years of data contained in five consecutive data base records are read into arrays ITMP and TMP from the data base.

210 End-of-file has been read and array ITMP is set to all 1s to signify such.

218 Continuation after reading. A check is made for the particular facility A thru L.

237 Check to determine if the General and Adminstrative Rate has already been set. This record element is constant throughout the data base for a particular year and a particular NARF. Part or all of array GAR may have been initialized by the user. This allows for changing the G&A rates that may have been incorrect on the data base.

243 Variable LN is set to the particular facility just read into array ITMP. This is done since an exit may be taken at statement 239+6 and the facility is still to be used at statements 241 and 242.

243+1 Variable ISET is 2 if and only if the record being considered is not the first of a group containing the current data base ID.

240 If this is the first record of a group, set array KASTID. It contains the ID for the current group of data base records and will be used later. Also exit from this program section if end of file (IEND=1) had been used.

239 Check the ID from the record read into array ITMP against the ID representing the current group as stored in array KASTID. Exit from this program section if the IDs are different.

236 If ITMP and TMP hold the first records of a group or hold records that have the same ID as the first record of the group, move the data from array ITMP and TMP to arrays IRECA and RECB, respectively.

250 Once arrays IRECA and RECB have been filled with all data for a particular ID, the arrays, MIXUP and FIXUP, containing information about data base updates, are checked for ID matches. A total of 18 modifications could have been made for each record read from the data base. It is theoretically possible to recreate the entire data base through use of these updates. The only requirement is that a new ID cannot be introduced. The data for a particular facility in a particular year and for a currently existing ID may be added, modified or deleted. Updates are entered using the hit parade cards in file IDNAR. A code indicates whether the rework activity is to be included in the hit parade and/or update list.

Column	What	Comment
6-12	ID	Must match an ID on the data base.
13	FACILITY	A thru L. It need not match an existing facility for the ID.
14	YEAR	Δ – all years. n – particular year: $1 \leqslant n \leqslant 5$.
15	CODE	Δ – rework activity to be included in the hit parade list. Update positions are ignored since none are expected. L – rework activity to be included in the hit parade list. Update will also be made to the data base as requested. B – do not include the rework activity in the hit parade list. This request is for data base updates only. D – delete the data base record. Do not include the rework activity in the hit parade list.
17-72	UPDATES	Eighteen potential data base modifications can be made. Only non-blank field will cause any updating. Given that ID's match, a facility is *added* when it is not part of the data base but it is entered in the hit parade list with a code of B or L. All 18 of the update items should be specified. Any that are not are assumed to be zero. Again, given the ID's match, a facility may be *deleted* when a D code is used. This option zeroes the appropriate location in array IRECA to signify that no data base record exists here. The values contained in the corresponding position of array RECB are not altered. They are used only when a non-zero value is contained in IRECA. The final data base update capability is to *modify* an existing data base entry. This is identical to the add except that only those of the 18 fields that are to change need to be stated. All others will remain unchanged.

250+1 The TEC corresponding to the CNA code is located in the dictionary. Array ITQS is initialized. It controls modification of the Total Quantity Supported (TQS) when updating data base records.

2500+1 Begin search of data base, update array for matching IDs.

2500+2 Testing of particular IDs for a match.

2504+1 If a match is found for the current data base ID, MXX, and NXX are set to the range of years on the control card, from file IDNARF or to the particular year for which the update is a delete.

2502 The numeric equivalent for the facility is noted in variable L.

2506 Begin update for the number of years requested.

2506+1 If the update is a delete request, zero the appropriate position in array IRECA.

2503 A change or add update was specified. If IRECA (1,K,L) is zero, an addition is being made to the data base and IRECA is filled at that location from the update array MIXUP. The call to MOVE fills in the particular year.

2517 Array ICHK is initialized and later used to indicate any changes that are made.

2507+1 All location in array FIXUP are checked for a value of negative 1 which indicates no update. All non-negative entries represent updates. If the TQS is to be modified (I=10) it is done to all facilities for that year. Otherwise, the modification is for the facility requested and in the year being processed.

2501+1 Update for a given facility in a given year.

2521+4 Update of the TQS for all facilities in a given year if it had not previously been updated (ITQS(K)=1).

2523 Note that an update was made to the I^{th} element.

2508+1 The 15th element from a data base record is the requirement and is computed from elements 10, 11,12, and 14. If any of these four items changed, the requirement must be recomputed.

2508+2 Check for update affecting the requirement.

2514 Recompute the requirement as a function of the Program (byte 5 from the ID).

2514+2 Recomputation of the requirements for Programs F, H or L.

2512 Recomputation of the requirements for Programs other than F, H or L.

2520 Since decimal points are assumed in all data, this section is primarily for converting to the correct value.

2520+2 Variable ISETR is set to one if the program from the ID is an A, N or T.

251 Shop percentages are additionally converted to the number of hours the rework activity normally spends in the shop.

251+1 The requirement is reconverted.

251+2 If the norm is zero, the total cost is zero.

251+4 If the norm is non-zero, the total cost is computed as a function of Program.

252 The computed total cost is set.

252+1 If the Program is not an F, H or L, the requirement is rounded.

248 Comparisons now begin. They are on two levels. The first is for the ID alone and the second looks at the facilities given that the IDs matched. Since the hit parade list and the data base records are both in ascending order, arithmetic comparisons can be made on the bytes. Variable IMOV is set to 1, 2 or 3 for the hit parade ID, being respectively less than, equal to or greater than the data base ID. IMOV is later used to indicate the results of the comparison.

248+4 Compare the bytes of the IDs from the hit parade and the data base.

255+1 The ID are equal. If both the hit parade list and data base file have been exhausted, jump to 400. Otherwise enter the DO loop for making entries into the matrix file after setting IMOV to 2.

254 The hit parade ID is less than the data base ID. Print an error message, set IMOV to 1 and go back for another pass.

253 The data base ID is less than the hit parade ID. IMOV is set to 3 and the DO loop is entered.

246 This DO statement ranges over the years requested for minimization. Either entries are to be made in the matrix file (IMOV=2) or adjustments are to be made to arrays CMLA, CAPB, and SUMN (IMOV=3). This is immediately check at location 246+1.

246+2 The arrays NBAD and NMATCH (see the next paragraph for details on their use) are initialized to zero. Array NBASE is filled with data regarding which and how many facilities are in this group of data base records.

286 Three work arrays, NBAD, NBASE, and NMATCH, are set according to the way the data base records and the hit parade requests compare. Consecutive locations of each become numeric integer values representing each of the 12 facilities. Given an ID, for example, the data base might contain facilities B, D, E, F, I, and J with facilities B, D, F, G, and I requested in the hit parade.

	1	2	3	4	5	6	7	8	9	10	11	12
D.B.		B		D	E	F			I	J		
H.P.		B		D		F	G		I			

NBAD (1) = 7
NBASE (1) = 5,10
NMATCH (1) = 2,4,6,9

NBAD (13) = 1
NBASE (13) = 2
NMATCH (13) = 4

There are at most 12 facilities that can be assigned to the first 12 locations in each of the arrays. Location 13 is used to tell how many (if any) consecutive locations are filled.

297+5 Since arrays IDNA and IRECA are initialized to zero and then altered according to the hit parade requests and the data base records, tests are made against zero to determine the entries in arrays NBAD, NBASE, and NMATCH.

265 The requirements in all data base records for which there was a corresponding hit parade request are summed.

257+1 If the sum of the requirements is zero, nothing needs to be written into the matrix file. This is insured by setting NMATCH (13) to zero. (There is no need to move the elements from NMATCH to NBASE. If the sum of the requirements is zero, each individual requirement is zero. All adjustments made from NBASE are a function of the requirement and, since each requirement is zero, the resulting adjustment is zero.)

268 The sum of the requirement is positive. Array LASTID is given a year with the call to MOVE, and the ID in array LASTID along with the requirements sum is saved in the next locations of arrays IDSAVE and REQID respectively. These items are the row name and right-hand side value to be entered into the matrix file at a later time.

249 Error messages are printed as a result of the non-zero entries in array NBAD.

272 Adjustments and accumulations are made as a result of the non-zero entries in array NBASE. The total capacity originally stored in array CAPB and the current manning levels originally stored in array CMLA are adjusted by the shop workloads computed in variable PROD. The final values in CAPB and CMLA are the remaining capacities and remaining

men that may be used in solving the problem. Given the unused space and men, the L.P. considers additional shift and hire/layoff criteria respectively. Additional quantities are accumulated in array SUMN as a function of the year, facility, fund source (determined at location 278), and program (determined at location 274). Array SUMN is passed to the Report generator where base workload (non-L.P. data) is also accounted for.

300 Entries into the matrix file are made as a result of the non-zero entries in array NMATCH.

300+4 If the total cost value for the ID NARF in the year being considered is zero, no matrix entries are made for that ID NARF in that year.

300+5 If all shop values for that ID NARF in the year being considered are zero, no matrix entries are made for that ID NARF in that year.

305 There is a non-zero total cost and at least one non-zero shop value. A value of '1' is entered into block G.

301+1 A value for the total cost from the data base is written into the matrix at block F.

302+1 If the facility is not a NARF (A-G), then no more entries are made into the matrix for the column name given in block E.

302+2 An entry is made into the matrix at block H for each non-zero shop entry.

307+1 The entries made at block H are also entered into block I if manning is at the shop level (INOS=2). When manning is at the NARF level (INOS=1) only one entry is made at block I. It is the sum of all entries made at block H.

320 There are two methods of placing limitations on the assignment of work by the L.P. The first is hard bounds (IBOP=1) computed from the individual requirement at an ID NARF and the sum of all requirements for the ID. The second uses additional rows called parametric equations to constrain the assignments. Values for the ID NARF column are rows that have the same ID (see 190).

320+1 When bounds are produced (IBOP=1) the column name (ID NARF), the requirement currently assigned to it and the sum of all requirements assigned for the ID are saved in arrays IBOUND and BOUNDS. The bounds are computed during creation of the bounds section of the matrix file. The reason for not computing the bounds at this point is that more than one set may be created using different percent variations and resulting in different ranges over which assignments are made at a facility.

324 When parametric equations are used, a factor of two times the number of columns having the given ID are made. That is, if an ID is being entered into the matrix at three facilities, six additional values will be entered for each ID NARF column and, since there are three columns, a total of 18 additional entries will be made. If n facilities are used for a given ID, then $2 \times n^2$ additional values are written into the matrix for the entire ID when parametric equations are used.

The entries are, like hard bound, based on a percent variation. However, unlike bounds which place a physical limitation on individual requirement at a particular facility, parametric equations specify a percentage range of the sum of all requirements. Suppose that 100 is the sum of all requirements for a particular ID, 30 is the assignment at a particular facility and a 10 percent variation will be allowed.

$$30 \pm .10 \times 100 = 20 \text{ and } 40.$$

Since the parametric method allows the user to modify the sum of all requirements (right-hand side) assume it is increased from 100 to 200. The net result is an increase from 30 to 60 assigned to the particular facility. Then, again given the 10 percent variation

$$60 \pm .10 \times 200 = 40 \text{ and } 80.$$

The values entered into the matrix are computed and put into array BOUNDA prior to entry into the matrix file. Consult the appendix of the user's manual for a detailed description of the values.

400 Shift control columns in block J and each associated row value are written into the matrix file from location 400 through location 415. Column names for block J are identical to row names in block C except for first character. The number of columns is 189 times the number of years requested for optimization. The four-byte code has a shift identification (U, V or W), Facility (A to G), Shop (1 to 9), and Year (IYA to IYB). The total number of card image records produced is 630 (10 x 9 x 7) times the number of years requested. That is, blocks K_U, K_V, K_W, L, M, N, O, P, Q, and R each contain 63 records.

400+3 This is the first of 10 write statements for the 10 blocks of data. Again, all value associated with a particular column must be entered into the matrix file before a new column is identified. The columns are entered as shown.

Block Entries

Column	Row	Value
J_U	A	K_U
	C_F	L
	C_S	M
	D_M	N
J_V	D_M	Q
	C_F	O
	C_T	P
	A	K_V
J_W	C_F	R
	A	K_W

The table is repeated 63, 126, 189, 252 or 315 times depending on the number of years to be optimized. Block K_U is the incremental cost for the second shift as read from user input file IDDI and adjusted by the efficiency factor for all second shift operations as given on the control cards from file IDNARF. Block K_V is similarly input and adjusted for third shift operations. And block K_W is input with no adjustment for any post third shift operations.

Block L is the negative of the efficiency factor for second shift. All 63 entries contain the same values. Block O is similar to block L but designates third shift. Blocks M, P, and Q are fixed at +1, +1, and −1 respectively. Blocks N and Q are the difference between 1 and the percentage for the respective shift efficiency factor. Within a given block all values are identical.

415+1 Manpower control columns are names in block S_i. They are similar in composition to the shift columns. However, manning is not likely to be modified at the shop level. But when it is, nine entries are made for each shop in a given NARF with byte position three assuming the values 1 through 9. Manning at the NARF level is more likely. That is, men are hired or layed off from a NARF, not from the shop within the NARF. In this case, one entry is made for the NARF (all shops). The shop designator in byte position three is set to zero. The two possibilities are a function of the request on the user control card. If it is assumed that manning is at the NARF level and byte position three is fixed at zero, then entries at blocks T, U, V, W, X, Y, and Z will be in multiples of seven rather than 63 as in the shift blocks. That is blocks T_H, T_L, T_G, T_K, X, and Z will each have seven entries for a given year. Blocks U, V, W, and Y will have 7, 14, 21, 28 or 35 entries for that year depending on two items. The first is the number of years being optimized. The

second is the number of years remaining, relative to the year being entered. An example: Years 2 through 4 on the data base are to be processed. That is, variable IYA and IYB are 2 and 4 respectively. Assume that year 3 is being entered into the matrix file. Then 14 entries are made for each of block U, V, W, and Y. The extra seven entries carries the manpower changes made in year 3 to year 4. That is, people are not hired or layed off for only one year. What happens to them in one year must be considered in all succeeding years.

422–1 Hourly NARF shop costs for hire and layoff are temporarily stored in array HCOST.

422+1 Variable M is set to the particular shop or to zero depending on whether manning is at the shop or NARF level.

422+6 When manning is at the NARF level, the cost over the nine shops is averaged in array HCOST.

430 Four types of columns are identified: initial (Phase I) hire and layoff (G and K) and post-initial hire and layoff (H and L). As usual, all data values for a given column must be entered into the file before a new column is identified. The following table shows all entries for a particular NARF in a particular year X when Y is the last year of the total years requested. Refer to figure 2c.

Block Entries

Column	Row	Value
S_H	A	T_H
	$D_M(X)$	U
	.	
	.	
	$D_M(Y)$	U
S_L	A	T_L
	$D_M(X)$	V
	.	
	.	
	$D_M(Y)$	V
S_G	A	T_G
	D_N	X
	$D_M(X)$	W
	.	
	.	
	$D_M(Y)$	W
S_K	A	T_K
	D_N	Z
	$D_M(X)$	Y
	.	
	.	
	$D_M(Y)$	Y

If only one year had been requested then X equals Y. Again, assuming that manning is by NARF, the table is repeated 7, 14, 21, 28 or 35 times depending on the number of years requested.

All values entered in blocks U thru Z are either positive one or negative one. All values in block T were previously read from the IDDI file.

500 Following completion of all values entered by column and row, the right-hand side (blocks AA thru FF) is generated. Block AA is much like a column name and is called 'RIGHT.'

This name is used exclusively unless parametric equations have been generated. Then and only then will block AA contain a user supplied name(s) other than 'RIGHT.' The new names represent from one to five different sets of right-hand side parametric changes to the values entered under the name 'RIGHT.' Only one set is used during execution of the linear program.

501 The right-hand side section of the matrix file is identified.

501+1 The right-hand side for the ID row (block B) is written into the matrix file. The row names and right-hand side values have been saved in arrays IDSAVE and REQID respectively during data base processing.

507+1 Written into the matrix file for each shop in each NARF and in each year reqeusted is the adjusted capacity (block BB) for the first shift. When positive, this number gives the number of hours of work that may be assigned to first shift by the linear program. Assignments beyond this number are automatically assigned to second, third, and post-third shift and in that order. When negative, the first shift has already been filled with assignments not being subjected to optimization in the linear program. The absolute value of the number is the minimum workload that has already been assigned to additional shifts. All assignments by the linear program will begin in the shift that has some remaining capacity.

507+4 Write the first shift adjusted capacity into the matrix file of block CC.

511+1 Compute and write into the matrix file at block DD the percentage of the first shift total capacity that will be allowed for the second and third shifts.

518+1 Manning values are entered at blocks EE and FF. The current manning level in hours was adjusted by the same quantities used to adjust capacities. Array CMLA is the quantity of men in manhours not already used by those workload assignments that are not to be optimized by the linear program. Thus, the remaining manhours in array CMLA will be used by the linear program as assignments are made. The difference between the current manning levels and the manning required as a result of all assignments gives hire/layoff information.

518+5 Compute the manpower variation for hiring and for laying off that will be allowed before Phase 2 costs are incurred.

518+7 Compute the shop variable KK used in Phase 2 identifying the row name.

518+9 Recompute the right-hand side values in array SUMR when manning is being done at the facility level.

526 Enter the manning figures into blocks EE (adjusted manning level) and FF (initial Phase 1 range of manning changes).

535+1 If parametric equations were generated (IBOP=2), additional right-hand side entries may be made. They represent alterations to be made to the right-hand side values which were just established. These entries are a function of the users' requests stored in arrays IRIGHT and FRIGHT. This section will not be generated when bounds have been requested.

542 The final section is for bounds. Bounds are the limits on the range of quantities that may be assigned by the linear program. That is, if 30 is the assignment before requesting optimization of a particular ID NARF, then 20 and 40 could be the lower and upper bounds respectively. The bounds are computed from the current assignment. From that point on, only the bounds have any meaning and the current assignment is ignored. Data for the bounds were saved in arrays IBOUND, BOUNDS, IBNDX, BNDY, IBNDA, BNDY, and IBNDC.

If the number of pairs of bounds JBDS or the number of sets of bounds NBS is zero, no bounds are to be generated.

542+2 Identify the beginning of the bounds section.

543+1 NBS sets of bounds will be written into the matrix.

543+2 JBDS paris of bounds (lower and upper) will be written into each set.

543+3 Isolate the program ITEMR and the year ITEMP for the column the particular bound represents.

543+5 Determine the numeric value for the year.

548 Begin the computation for the lower and upper bound values.

548+1 Are there any bound exceptions to be checked in arrays IBNDA, IBNDB, and IBNDC.

548+2 Sequentially search arrays IBNDC and IBNDA for (1) the corresponding set the bound exception represents, (2) whether the exception is for the lower only, upper only or both bounds and (3) the column to which the exception applies.

550+4
555+1 Set BVAL equal to the value of the bound if a bound exception was found.

558 No bound exceptions were requested or none existed for the particular column upon which the bounds are being placed. Compute in BVAL the bound as determined by the standard percent variation contained in array BNDB.

566+1 If the program is not F, H or L, the value of the bound is truncated to an integer.

569 The bound is written into the matrix.

800 The terminal record is written into the matrix.

808–1 Much of the data and information produced by the matrix generator is to be used and/or reported in the report generator. That program takes the linear programming output and the data written onto tape here and produces meaningful results.

CHAPTER IV

REPORT GENERATOR

INTRODUCTION

Execution of the Linear Program (L.P.) and the Report Generator Program is the final step in the optimization problem. The matrix data set created by the Matrix Generator Program is fed into the L.P. and an optimal solution is found. This result is saved as a direct access data set in "Communication Format." The data is then accessible through a routine call READCOMM.

The reports are composed of data from both the solution produced by the L.P. and the Base information. Base information is all the data that is *not* put through the L.P. for optimization. It is the unmodified quantities that must be included in order to produce complete and thorough reports.

The individual reports will sometimes contain only L.P. results. Otherwise, both Base and L.P. and their aggregates are included. Samples of all reports can be found in the User's Guide. Only a discussion of the program code is presented here.

DEFINITIONS OF ARRAYS AND VARIABLES

COMMON/DICT/NDR, IDICT (2,400)

NDR Number of dictionary records.

IDICT (2,400) Dictionary entries containing the TEC and its CNA-designated code. Both represent the T/M/S contained in array JDICT.

Arrays, Integer

ICTU (3) The word 'CONTINUED' is put into this array when a report extends beyond one page. Otherwise, it is blank.

IDATE (5) Contains the date as read from the first card of an L.P. run.

IFUNDX (8) Left-justified and blank filled code for each of the eight possible fund sources.

IFY (5) Filled with the five actual years that an L.P. run would represent (e.g., 74, 75, 76, 77, 78).

INARFX (12) Left-justified and blank filled code for each of the 12 rework facilities (A thru L).

IPROGX (10) Left-justified and blank filled code for each of the 10 programs.

IRNN (13) Contains the report header as read from the first card of an L.P. run. This name is written on every report.

ISHOPX (10) Left-justified and blank filled code for each of the nine shops within a NARF. The tenth location contains a zero.

ITEC (7) Temporary locations for containing the ID previously considered. Used for comparison with ITED.

ITED (8) Contains the ID and facility currently being considered. Compared with ITEC to identify consecutive equivalent IDs.

IYEARX (5) Left-justified and blank filled code for the one thru five years that may have been requested.

JCTU (3) Data array containing the work 'CONTINUED'.

JDENT (26) Various constants upon which tests are made.

JDICT (4,400) Array for holding the type/model/series as given in the dictionary file.

Arrays, Real*8

BAJ (2,9,7) Initially the manpower adjustment from years prior to the report year. Finally used as bounds away from the new Current Manning Level (DML) to indicate Phase I hire/layoff limits.

BASE (10,8) Base workload for all shifts. Sum of BSWKLD and S23BW.

BSWKLD (9,7) The first shift base workload. Computed as the difference between the total capacity of a shop (TOTCAP) and the adjusted capacity (CAPB).

CAPB (9,7,5) Adjusted capacity. Work space available after all workloads NOT subject to L.P. optimization have been accounted for. A negative CAPB indicates additional shifts are needed for all the base workload.

CNL (9,7) Current manning level.

CVA (14) Cost/volume adjustment.

DML (10,8) New current manning level following the hire/layoff adjustments from all previous year hire/layoffs.

DOP (13) Hollerith data for the 12 facilities plus a total.

DOPCS (7,5) Total workload for a given NARF in a given year.

DOPGT (12,5) Total dollars for a given facility in a given year.

FACTOR (5) Second shift efficiency factor by which the second shift capacity is multiplied to determine the maximum shift workload.

FACT3 (5) Third shift efficiency factor by which the third shift capacity is multiplied to determine the maximum third shift workload.

FPN (9,11,14) L.P. workload assignments by Fund (B), Program (10), and NARF (12). The extra dimensions provide for totals.

FUNDN (9) Hollerith data describing the eight individual fund sources plus an entry for all fund sources.

GCOST (9,12) Phase 1 yearly hiring cost by shop.

GDOL (9,7) Phase 1 yearly dollars required for hiring by shop.

GSN (9,12) Phase 1 yearly men to be hired by shop.

HCOST (9,7) Phase 2 yearly hiring cost by shop.

HDOL (9,7) Phase 2 yearly dollars required for hiring by shop.

HPY (7,5) Hours per year for each NARF.

HSN (9,7) Phase 2 yearly men to be hired by shop.

HSUM (14) Phase 1 and Phase 2 yearly dollars required for hiring by NARF.

KCOST (9,12) Phase 1 yearly layoff cost by shop.

KDOL (9,7) Phase 1 yearly dollars required for layoff by shop.

KSN (9,12) Phase 1 yearly men to be laid off by shop.

LCOST (9,7) Phase 2 yearly layoff cost.

LDOL (9,7) Phase 2 yearly dollars required for layoff by shop.

LPWKLD (10,8) L.P. workload.

LSN (9,7) Phase 2 yearly men to be laid off by shop.

LSUM (14) Phase 1 and Phase 2 yearly dollars required for layoff by NARF.

MEN (10) Number of men required for the total workload (TWKLD).

NARFN (2,14) Hollerith data containing 12 facility names plus total data.

PB (9,200) Production bounds array. Up to 2000 ID NARF selected column data from the L.P. output in any given year.

PHPY (5) Pay hours per year. Currently not used in the program.

PROGN (10) Hollerith data for the ten program names.

PROID (13) Temporary locations for the L.P. workloads assigned to the various facilities for a given ID.

PUTIL (10,8) Percent utilized. Ratio of space used to space available based on the first shift.

RC (9,7,4,4) Reduced cost. Hire/layoff column data by NARF (9), Shop (7), Phase (4), and selected data (4).

RR (6,1000) Rework requirements. Up to 1000 ID selected row data from the L.P. output in any given year.

SHOPN (9) Hollerith data for the nine shops.

SHPNRF (2) Temporary location which holds either the shop name or the NARF name just prior to printing.

SLACK (9,7) Slack. Difference between the RHS and the assigned workload. Represent the unused first shift work space.

SP (9,7,5,4) Shadow price. Row data by NARF (9), Shop (7), Shops and Manpower (5), and selected data (4).

SUBT (12) Temporary location for computed subtotals just prior to printing the Workload Variance Report.

SUMN (9,11,14,5) Base workload by Fund (8), Program (10), NARF (12), and Year (5). The extra dimensions provide for totals.

S23BW (9) Computed base workload assigned to second and third shifts.

S1WKLD (9,7) First shift L.P. workload. By shop and NARF.

TOTCAP (10,8,5) Total capacity by Shop (9), NARF (7), and Year (5). Extra dimensions provide for totals.

TOTL (9,11,14) Summed base and L.P. workloads by Fund (8), Program (10), and NARF (12). Extra dimensions provide for totals.

TWKLD (10,8) Total workload by Shop (9) and NARF (7). Extra dimensions provide for totals.

U (9,7) Workload assigned to the second shift by the L.P.

UBND (9,7) Total capacity of the second shift by shop and NARF.

UBND3 (9,7) Total capacity of the third shift by shop and NARF.

USUM (14) Incremental accumulated L.P. second shift costs by NARF.

V (9,7) Workload assigned to the third shift by the L.P.

VALUES (12) Area where the L.P. values are temporarily stored. That is, elements of inputed vectors.

VSUM (14) Incremental accumulated L.P. third shift costs by NARF.

W (9,7) Workload assigned to post-third shift by the L.P.

WSUM (14) Incremental accumulated L.P. post-third shift costs by NARF.

Variables, Real*8

ACOST Adjustment cost–dollars resulting from AIIC.

AIIC Manning adjustment included in cost–(1) Phase 2 only or (2) both Phase 1 and Phase 2.

AIRCFT Data location with the word AIRCRAFT.

ANAME Dummy parameter used in the CALL ARRAY statements.

BLANKD Eight hollerith blanks.

TAIIC Total for all AIIC above.

TCOST Total for all ACOST above.

TEMP, TEMQ, TEMR Temporary localized variables.

TGPA Gross personnel adjustment.

TOTAL Data location with the word TOTAL.

TMA Total manning adjustment–computed value for the Workload Variance Report.

INPA Net personnel adjustment.

TSUM Temporary location for totals in DOP and Program Cost Reports.

SLML Percentage of CML used to compute Phase 1 layoff bound.

XUML Percentage of CML used to compute Phase 1 hire bound.

Variables, Integer

IAY The year (eg. 75) in hollerith represented by IYA.

IBOP 1 or 2 for bounds 1 or bounds 2 option respectively.

IFILE Logical units where the L.P. solution is stored.

INOS 1 or 2 for manning by NARF or shop resectively.

IRQQ Blank or non-blank for non-production or production of Group 2 reports respectively.

IYA Integer for the first year to be reported. ($1 \leqslant IYA \leqslant 5$).

IYB Integer for the last year to be reported. ($IYA \leqslant IYB \leqslant 5$).

K Primarily used to represent the year of the reports being generated.

LUQ Logical unit where the dictionary is stored.

LUU Logical unit where the other data (non-L.P.) is stored.

NFACIL Number of Facilities (12).

NFUNDS Number of Funds (8).

NNARFS Number of NARFs (7) (NNARFS is a subset of NFACIL).

NPROGS Number of Programs (10).

NSHOPS Number of Shops (9).

NYEARS Number of Years (5).

All remaining integer variables take on temporary values and are localized in their use.

L.P. OUTPUT FORMAT

The output from the L.P. is dependent upon the requests included in the problem. SOLUTION must always be included in order to generate the first set of reports. RANGE must be included if and only if the second set of reports is ever to be generated from this particular L.P. output. The results are eight blocks of data as follows:

Group	Array	Comment
	SOLUTION	– ignored by R.G.
SOLUTION	RSECTION	– rows
	CSECTION	– columns
	RANGE	– ignored by R.G.
RANGES	SECTION1	– rows at limit level
	SECTION2	– columns at limit level
	SECTION3	– rows at intermediate level
	SECTION4	– columns at intermediate level

Only those called SOLUTION are used in producing the first set of reports. Both SOLUTION and RANGES are used for the second set. The second set may not be generated without the first.

Arrays named SOLUTION and RANGE are header information for each of the two groups and are ignored by the report generator. RSECTION, SECTION1, and SECTION3 contain row data. CSECTION, SECTION2, and SECTION4 contain column data. As each array is passed through, the column (or row) vectors are encountered in the same order in which they were produced in the matrix generator. However, a column (and row) is mentioned only once in each of the two groups. That is, if a row is in SECTION1, it will not appear in SECTION3. For example, given that rows and columns are entered into the matrix in the order shown below,

Rows	Columns
AA	A1
BB	A2
CC	B1
	B2
	B3
	C1
	C2

the row and column vectors might be encountered in the L.P. output as follows:

Arrays	Vectors	Comment
SOLUTION		ignored
RSECTION	AA	rows
	BB	
	CC	
CSECTION	A1	columns
	A2	
	B1	
	B2	
	B3	
	C1	
	C2	

Arrays	Vectors	Comment
RANGE		ignored
SECTION1	BB	rows
SECTION2	A2	columns
	B1	
	B3	
SECTION3	AA	rows
	CC	
SECTION4	A1	columns
	B2	
	C1	
	C2	

The appearance of the vector names in RSECTION and CSECTION is fixed. Their appearance in the other sections is a function of the solution to the problem.

Each vector is further subdivided into elements which provide the actual answers or data to be used in the reports. Each element belongs to a vector and each vector belongs to an array. The number of elements in a vector depends on which array the vector belongs to. Each element of all vectors in a given array are named and described below.

Element number	Name	Description
RSECTION		
1	*	ROW name
2	STATUS	Ignored
3	ACTIVITY	Value associated with ROW name
4	SLACK	Difference between RHS and ROW ACTIVITY
5	LLIMIT	The algebraically lowest value that the ROW ACTIVITY can take and remain feasible
6	ULIMIT	The algebraically highest value that the ROW ACTIVITY can take and remain feasible
7	DUALACT	Ignored
8	NUMBER	Ignored
CSECTION		
1	*	COLUMN
2	STATUS	Ignored
3	ACTIVITY	Value associated with COLUMN name
4	ICOST	Inputed cost of COLUMN name
5	LLIMIT	The algebraically lowest value that the COLUMN ACTIVITY can take and remain feasible
6	ULIMIT	The algebraically highest value that the COLUMN ACTIVITY can take and remain feasible
7	RCOST	Reduced cost–rate of increase in the objective function per rate of increase in the COLUMN name
8	NUMBER	Ignored

Element number	Name	Description
SECTION1 and SECTION3		
1	*	ROW name
2	LOACT1	Level to which the ROW ACTIVITY may be decreased at a cost per unit of decrease given by the unit cost.
3	UNCOST1	Change in the objective function per unit of decrease in the ROW ACTIVITY
4	LIMPROC1	Ignored
5	AT1	Ignored
6	UPACT2	Level to which the ROW ACTIVITY may be increased at a cost per unit of increase given by the unit cost
7	UNCOST2	Change in the objective function per unit of increase in the ROW ACTIVITY
8	LIMPROC2	Ignored
9	AT2	Ignored
SECTION2 and SECTION 4		
1	*	COLUMN name
2	LOACT1	Lower bound on shadow price
3	UNCOST1	Reduced cost–the change in the objective function per unit of decrease in activity level down to LOACT1
4	UPCOST1	Ignored
5	LIMPROC1	Ignored
6	AT1	Ignored
7	UPACT2	Upper bound on shadow price
8	UNCOST2	Reduced cost–the change in the objective function per unit of increase in activity level up to UPACT2
9	LOCOST2	Ignored
10	LIMPROC2	Ignored
11	AT2	Ignored

RETRIEVING L.P. OUTPUT

The task is to read the data produced in the L.P. solution and put it into the proper data array for the report generation. The reading is simplified by a routine called READCOMM which is specifically designed for reading data produced in the format of the L.P. solution. Three entry points in READCOMM are used. Their function and parameters are described below.

CALL POSITN (IFILE,INDIC,NARY). Position logical unit IFILE for inputing the data in Array NARY. INDIC is returned and ignored.

CALL ARRAY (IFILE,INDIC,ANAME). Prepare the previously positioned array for reading. The array name ANAME and an indicator INDIC are both returned and ignored. Succeeding calls to ARRAY prepared succeeding arrays for input.

CALL VECTOR (IFILE,INDIC,VALUES). Read a vector of elements into VALUES. The number of double precision elements is determined by the array being processed. VALUES is dimensioned to hold all elements. If INDIC=1 upon returning, there are no more vectors in the array. Succeeding call to VECTOR returns the elements of succeeding vectors.

All data produced by the L.P. may be referenced using the above calls to subprogram READCOMM. Various dimensioned data arrays are filled with the element read. The particular data

array depends on the vector and array from which the elements came. Quite often the name of the vector which is contained in VALUES(1) must be investigated to determine the elements positions in the data arrays. This is especially true when reading SECTION1, SECTION2, SECTION3, or SECTION4 since a vector's location is function of the solution.

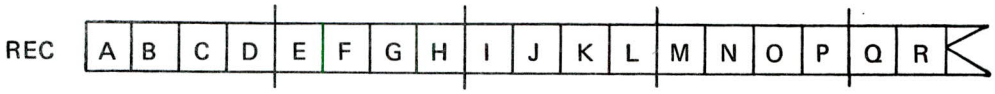
DISCUSSION OF PROGRAM CODE

(Subroutine MOVE and function LJABF are described in chapter III.)

Function IBYTE (A,B)

This assembler language function is for isolating the B^{th} byte of the A^{th} address. The byte is right-justified in register Ø and zero filled on the left.

e.g., I = IBYTE (REC (2),1Ø)



The content of I after returning from the function is ØØØN

$$I = ØØØØØØD5_{16}$$

Subroutine SEARCH (ITEC,NN)

This FORTRAN subroutine is for performing a binary search of the NDR dictionary records stored in array IDICT. Array ITEC contains the two hollerith characters used as the CNA designated code for TEC–T/M/S. Character positions 1 and 2 of each dictionary record contain a similar code. When the exact match is found the records location in IDICT is returned in variable NN. If no match occurs, NN is set to zero.

Subroutine CVADJ

This subroutine does the computations described in appendix B of the User's Guide. The arrays for this subroutine are described below.

Definition of Arrays

CVA (7) Computed cost/volume adjustment returned to the main program for the particular NARF.

DOPCS (7,5) Total workload for a given NARF in a given year.

TABLE (2,8,7) Rate Variance Table.

CARD (2,7) Negotiated DLH and G&A rates.

GART (2,8) G&A Rate Table computed from TABLE and CARD.

GAR (7,5) The original G&A rates by NARF and year.

Code

5–5	Initialization at entry point CVADJ.
5	Read the original G&A rates.
40	Read the negotiated DLH and G&A rates.
43	Read the Rate Variance Table.
41+2	Cost/volume computation at entry point CVADJ1. JJJ is the NARF and LLL is the year.
41+3	'Compute G&A Rate Table called GART.
150+1	Set ZMANH to the workload in thousands of hours.
150+2	Compute GARATE from ZMANH and GART.
230	Compute the cost/volume adjustment.
	If the original G&A rate in array GAR and the negotiated G&A rates in array CARD are equal, then CVA(J) will equal zero.

Main

Main of the Report Generator Program generates the required reports as part of MPSX360. The Report Generator Program is called as a subroutine of the L.P. when an optimal solution is found and the L.P. output generated. Cost/volume information and report titles are also required as input. Two sets of reports will be discussed. The first set is always a product of the Report Generator while the second set is optional. Reference will again be made to entries in the location field of the FORTRAN source listing.

Solution Data Input and Workload Assignment Report

∅	Common, Arrays, Variables, etc.
5–4	Initialization.
3	The dictionary is read from logical limit LUQ into arrays IDICT and JDICT.
8+8	Initialization of arrays which have a year subscription.
25+1	Various data is read from logical unit LUU. This data is not put through the L.P. but is necessary for producing meaningful reports. It is read here in the same format under which it was written in the Matrix Generator Program.
20+1	Initialization of subroutine CVADJ where cost/volume adjustments are computed.
20+2	Current manning level is converted from manhours per year to workers per year within a given shop.
18+1	If manning is to be considered only on the facility level (not shop category level), accumulate all men in shop category 1.
11	Compute a hollerith equivalent for the five years over which the reports could be generated. That is, if the reports start in year 3 (IYA) and that year represents 1976 (IAY) then array IFY=74,75,76,77, and 78.
16+1	Modify arrays FACTOR and FACT3.
12+1	Outermost loop controls the number of years requested when the matrix was created. The reports generated will be for all those years.
12+2	Initialization of all arrays that are used only on a yearly basis.

Read and save all required elements from RSECTION

21	Position to and prepare the second array (RSECTION) for input. Call ARRAY automatically points to vector one.
21+2	If IEND2=1, then RSECTION has already been read in.
21+2	Initialize counters.
22	Get the next vector of elements.
22+1	Split the vector name (VALUES(1)) into bytes for testing purposes.
23+1	Is it the correct year?
23+2	Test for N, P or M.
26+1	Increment NR and save an element in the appropriate array. Since the rows were created in groups identified by F, S, and T for first, second, and third shifts they will be read back in the same way. Only rows identified as N, P or M or the incorrect year are eliminated. The result is the filling of arrays SLACK, for the first shift slack, UBND and UBND3, for upper bounds on the second and third shifts, respectively.
33	Counters are incremented and tested. When NN is greater than NNARFS, the arrays have been filled.
33+8	Set M equal to the number of remaining rows for shift (F,S,T) or manpower (N,P,M) data. This number is computed as shown.
33+11	Skip M vectors.
37	Get the next vector of elements from array RSECTION. The first time this call to VECTOR is performed, the first seven character ID will be returned.
37+1	Was a vector of elements returned?
37+2	Isolate and test the year byte for year being processed.
37+4	Isolate and test the eighth byte for a blank. If it is non-blank then parametric equations were used to put bounds on the columns. Parametric rows are ignored here. Their use is confined to execution of the L.P.
37+6	Save the row name (ID) and the activity (manhours of work) which the L.P. determined was necessary for that ID.

Following the complete input of RSECTION, the next array, CSECTION, is prepared. This array is prepared and input NPROGS (10) times. With each pass, a workload assignment report is generated for a particular program (byte 5 of the ID NARF) if at least one ID NARF contains that program. The ID NARFS are the first set of vector contained in array CSECTION.

42	Initialization.
42+4	Clear to CONTINUE data array that is used in the header of each report.
43+1	Prepare the third array, CSECTION, for input.
44+2	Have all reports been produced?
44+3	Clear array PROID.
45	Get the next vector of elements in array CSECTION.
45+1	Split the column name into bytes for testing.
46+1	Is the third byte (fund source) a one (ISHOPX(1))? If it is, there are no more ID NARFS in CSECTION. This test is made since all ID NARFS will have a fund source other than a

one and the first column vector beyond the set of ID NARFS will always have a one in the third position.

48 Is it the correct year?

48+1 If this is the first pass through CSECTION, save the column name and other values in data array PB.

52 Does this ID NARF contain the same program as is being reported?

52+1 Set J to the particular facility (NARF) in byte eight.

56 Set I to the particular fund source in byte three.

58 Array FPN is sub-dimensioned as Fund Source (8), by Program (10), by Facility (12), and is used to accumulate the products of workload assigned (values (3)) by Unit Cost (values (4)). The array is used later for totaling and reporting.

58+1 If this is the first line of a report for a particular program, save the ID in array ITEC.

59 Test the current ID in array ITED with the previous ID in array ITED. If they are different, a line should be printed. If they are the same, the workload assigned should be saved in the facility position within array PROID.

62 Print the header for the 'workload assignment' report when appropriate.

68+1 Put the word CONTINUED into array ICTU.

65+1 The column header varies according to the program being reported. Programs 3, 4, and 5 (F, H, and L) will be modified to represent percentages of the total rather than the quantities.

70 The line count is incremented and the total is put into PROID (13).

71+1 If program F, H or L is being reported, the quantities are changed to a percentage of the totals.

64 The values in PROID are rounded to the nearest integer and tested against the original value in PROID. If the difference is less than Ø.1, PROID is set to its rounded value. That is, if PROID is within one-tenth of being an integer value, it is set to the integer value.

82 Locate the dictionary record NN containing the CNA code in ITEC(1) and ITEC(2).

82+2 Print a line in the report and zero the PROID data array.

73+1 IEND was set to 1 if last ID NARF in CSECTION was read.

After all 'workload assignment' reports have been produced, the remaining vectors in array CSECTION are inputed with specific elements being transferred to appropriate data arrays.

90 Initialize counters.

92 Get the next vector of elements.

92+1 Have all the ID NARF vectors been bypassed yet?

92+2 Test bytes 3 and 4 for shop 1 and the correct year, respectively.

93 Once the correct second shift (U) column (first encounter of shop 1 and the right year) has been read in, 189 consecutive vectors are used to fill data arrays U, USUM, V, VSUM, W, and WSUM. Arrays U, V, and W contain the total number of hours for the given NARF shop in the second, third and post-third shifts, respectively. Arrays USUM, VSUM, and WSUM are the total cost for the entire NARF (not by shops) of the second, third, and post-third shifts, respectively.

97+9 Compute M and then skip the next M vectors.

125 Hire and layoff vectors in CSECTION are considered next. Manpower changes may be done at the shop level (JS=NSHOPS). However, they will normally be moved in and out at the

NARF level (JS=1). That is, a man would be hired to work at a NARF, not a particular shop within the NARF.

125+3 The data array BAJ is zeroed. It will contain the net manhours changes that occurred in all years prior to the year being processed.

91 Assuming JS=1, the call to VECTOR is executed four times.

91+3 Has the year of the current reports been reached yet?

91+4 No. Isolate the year and test it for the current year.

94 If the data in the vector being processed is for a year prior to the year for the reports currently being generated, accumulate the hire (+VALUES(3)) and layoff (–VALUES(3)) manhours in data array BAJ.* Hire and layoff is done in two phases (see User's Manual).

I=1, phase 2 hire
I=2, phase 2 layoff
I=3, phase 1 hire
I=4, phase 1 layoff

123 Once the current year has been reached, TEMP is set to a dollar cost; manhours (VALUE(3)) times Unit Cost (VALUE(4)). The appropriate arrays are then filled with the elements from the vector. HSUM, LSUM, GSUM, and KSUM are the total cost over the entire NARF for manpower changes. HSN, LSN, GSN, and KSN are the number of men to be hired and layed off within each shop. HCOST, LCOST, GCOST, and KCOST are the total dollars required to hire or layoff one man from a given shop. HDOL, LDOL, GDOL, and KDOL are the costs for manpower changes by shop. All of these arrays are used later for appropriate reports.

The remaining Group I reports will be discussed only where a question might arise regarding what was done.

Workload Variance Report

107+2 The base workload (BSWKLD) is the difference between the total available first shift capacity (TOTCAP) and the adjusted capacity (CAPB).

107+4 If the adjusted capacity (CAPB) is less than zero, then total available space on first shift was insufficient. That is, some of the base workload is already assigned to additional shifts. The exact amount is computed and entered into S23BW.

109 The total first shift workload S1WKLD is computed from the total first shift capacity TOTCAP minus the unused space SLACK.

109+1 The total workload TWKLD can then be computed. It is the sum of the first (S1WKLD), second (U), third (V), and post-third (W) shift workloads.

109+2 Then the workload (LPWKLD) assigned by the L.P. can be computed.

113 The percentage of utilization (PUTIL) is the total workload relative to the first shift capacity.

131 A line is printed (see User's Guide).

134+ Subtotals are computed and printed.

149 The total base workload for all shifts is saved in array BASE for later use.

*This occurs only when the set of reports being produced contains more than one year.

All NARF Summary Report

171–3 Capacities and total workloads are summed by NARF and shop and percent utilization is computed for the totals and subtotals. Finally, all items for the capacities, workloads, and utilization are printed.

Required Manning Level Report

402–1 ,Print report headers.

407–1 The subtotals in array TWKLD are updated to include the workloads assigned to shop "other" which were not reported in the previous report but are reported here.

409+1 Begin report data printing.

422 Array TWKLD is printed to show the total workload in hours.

426–1 TWKLD is then modified to represent the workload in workers (i.e., number of men).

426 Totals by shop are accumulated in array MEN.

426+1 Array TWKLD is again printed to show the total workload in workers.

425 The total workload in workers for all shop categories is printed.

DOP Cost Reports and Program Cost Reports

209–7 The arrays VSUM, USUM, WSUM, HSUM, LSUM, SUMN, FPN, and TOTL are all over-dimensioned. The additional space in each array is provided so that subtotals and totals may be computed and then entered as part of the array. That is, rather than an eight by ten by twelve array for the fund, program, and facility, respectively, the arrays SUMN, FPN, and TOTL were all expanded to nine by eleven by fourteen to accommodate the subtotal and total. Thus, each DOP cost report has (1) subtotals at the right for each fund source over all programs and (2) subtotals at the bottom for each program over all fund sources. All DOP cost reports are totaled into the Program Cost Reports by all 12 facilities and by the seven NARFs. Finally, cost/volume adjustments, additional shift incremental costs, and manpower change costs are added to the totals at the lower right of each of the seven NARF DOP Cost Reports and the Program Cost Reports.

220+1 The appropriate header is printed.

227+1 Because of the structure of arrays SUMN, FPN, and TOTL, it is now a simple matter to print the body of each report.

235+1 The cost/volume adjustment is computed and printed along with the shift and manpower changes.

Manning Level Variance Report

251–4 This routine also makes use of expanded arrays for reporting subtotals and totals. It, furthermore, has the distinction of a variable number of reports. That is, if manning adjustments are made by NARF, only 7 reports are generated. Otherwise, manning is changed by shop and 63 reports will be produced.

251–3 Change the base workload, L.P. workload and previous year manning adjustments from manhours to men.

251+6 Compute NARF or NARF shop new current manning level resulting from the L.P. generated previous year manning adjustment (BAJ).

251+8 Subtotal and total the new manning levels.

251+11 Reuse array BAJ to represent the bound in men of Phase I hire and Phase I layoff.

257 Compute subtotals and totals for the base and L.P. workloads.

253+1 Variables MXX and NXX are set to represent the quantity of reports to be generated. If manning is by NARF (INOS=1), then one set (NXX=1) of seven (MXX=NNARFS) reports is produced. Otherwise, manning is by shop (INOS=2) and seven sets (NXX=NNARFS) of nine (MXX=NSHOPS) reports is generated.

258+1 The outer DO loop for the number of sets of reports.

262+4 Variables JJ, JX, LL, LX are set at appropriate positions and then used to represent the correct array position when producing a particular report within a particular set of reports.

266+1 Initialize variables for gross and net personnel adjustments, for manning adjustment included in the cost and for the total manning adjustment costs.

266+5 Inner DO loop for each report in a given set.

259+1 Compute the total manning adjustment.

259+2 Computation of AIIC is dependent on two items: (1) is hire or layoffs produced by the L.P. and (2) is Phase I hire/layoff costed.

259+2 Any Phase I layoffs?

259+3 Any Phase I hiring costs?

273 Compute Phase I and Phase II hiring costs.

275 Any Phase I layoff costs?

278 Compute Phase I and Phase II layoff costs.

280+1 Compute gross and net manning adjustments.

280+3 Print a line.

295+1 Print totals.

Ranges Data Input

Following Group I reports is the optional production of Group II reports. They require further data from the optimal solution provided by the L.P. The arrays called SECTION1, SECTION2, SECTION3, and SECTION4 contain the necessary additional information. The identifying element of each vector from each array is questioned and the appropriate data array is filled with other elements from the vector. No reports are produced until all the necessary L.P. data has been input.

SECTION1 and SECTION3 together contain as many vectors as RSECTION. The elements are stored according to whether a limiting value was reached in finding the optimal solution. SECTION1 holds those assignments that reached a limit while SECTION3 contains those results which did not extend to the bounds. A similar process is used with the column data contained in SECTION2 and SECTION4.

513–6 If Group II reports are not requested (IRQQ=IBLANK), then go to 999.

513–6 SECTION1 and SECTION3 are the fifth and seventh arrays respectively.

513 Get the next vector of elements.

513+2 Isolate and test the third byte of the row name. If it is less than 4HØ$\Delta\Delta\Delta$ (ISHOP(10)), then a letter has been encountered (from 4HA$\Delta\Delta\Delta$ to 4HZ$\Delta\Delta\Delta$). This means that all shift and manpower row vectors have been input and the *first* rework activity row has just been read.

513+4 Isolate and test the fourth byte. If it is the same year for which reports are being produced, then continue. Otherwise, get the next vector of elements.

513+1 Isolate the first byte. If it is an F, S, T, N or P, then continue. Otherwise, get the next vector of elements.

531 Determine which Shop (I) is represented by the elements.

518+1 Determine which NARF (J) is represented by the elements.

523 Enter the wanted elements into the proper positions in data array SP.

537 Get another vector of elements for the rework activities.

543 Isolate and test the fourth byte for the year of the reports being generated.

541 IROW is incremented until corresponding rework activity names (ID) are identical.

541+2 The wanted elements from the row vector are saved in data array RR along with elements previously saved while generating the Group I reports.

560 A process quite similar to that used for the row arrays, discussed above is used to save data from the column arrays (SECTION2 and SECTION4). Column vectors are in the sixth and eighth arrays, respectively.

573 Get the next vector of elements for the rework activities.

573+2 Isolate and test the third byte of the column name. If it is greater than or equal to 4HØ$\Delta\Delta\Delta$ (ISHOP(10)), then a number has been encountered (from 4HØ$\Delta\Delta\Delta$ to 4H9$\Delta\Delta\Delta$). This means that all rework activity columns vectors have been input and the first of the shift and manpower columns has just been read.

573+4 Isolate and test the fourth byte for the correct year.

582 ICOL is incremented until corresponding rework activity names (ID NARF) are identical.

582+2 The desired elements from the column vectors are saved in data array PB along with elements previously saved while generating the Group I reports.

579 Get the next vector of elements for the shifts and manpower.

601 Isolate and test the fourth byte for the correct year.

601+2 Isolate the first and third bytes of the row name.

601+4 Test the first byte for an H, L, G or K. That is, is this a manpower column? (The shift data (U,V,W) from SECTION2 and SECTION4 are not saved or reported.)

621 Determine which Shop (I) is represented by the elements.

608+1 Determine which NARF (J) is represented by the elements.

613 Enter the desired elements into the proper positions in data array RC.

Five more reports, Group II, are produced. Their primary contents are the data stored in arrays RC, SP, RR, and PB.

Production Bounds and Rework Requirements

650 The first two reports are very similar. They are both produced in the code presented here.

650+1 Variable NXX is set accordingly.

650+3 The reports are produced by program with 22 lines per page.

651−1 Blank the array used as a continue statement.

651+1 The NXX vectors of either array PB or array RR will be searched sequentially for the program being reported.

651+2 Put the ID or ID NARF into array ITED.

654+1 Test for the correct programs.

654+2 If LCT is 22 then a header should be printed.

670 Increment the line count.

670+2 If this is a Production Bounds Report, set J to the particular facility. Then check the current ID against the previous ID. If they are equal and LCT is not equal to one, the T/M/S–TEC is not repeated in the print statement.

689 Save the current ID in array TEC.

671+1 Search the dictionary for the T/M/S.

671+3 Print a line using either the data in array PB or array RR.

Shop Category Constraints

700 There are always seven of these reports; one for each NARF.

700+1 Print the headers.

705+2 Print four lines for each shop. First, second, third, and post-third shift information is produced.

Manpower Variance

750 This report and the next are dependent on whether manpower changes are by shop or NARF. Variables NXX and MXX are set accordingly.

761 Print NXX Manpower Variance Reports.

761+1 Print the headers.

768+1 Initialize the total cost TCOST.

768+2 Print MXX segments in each report.

768+3 Variables LL and JJ and array SHPNRF are also set according to manpower changes by shop or by NARF.

775 Either KSN (JJ,LL) or GSN (JJ,LL) will be equal to zero. That is, either Phase I layoff or Phase I hire was produced by the L.P. Thus, either layoff or hire information is printed.

790+1 The accumulated total cost is printed at the bottom of each report.

Manning Level Constraints

800 Print NXX Manning Level Constraint Reports.

800+1 Print the headers.

812+1 Print MXX segments in each report.

812+2 Variables LL and JJ and array SHPNRF are set according to manpower changes being by shop or by NARF. If manning is by NARF, the total workload TWKLD is also accumulated in the first location of TWKLD.

820 Print the appropriate data.

DOP Workload and Cost Summary Report

After all reports in all years have been produced, a summary is generated. It is a NARF and facility report by year of the data contained in arrays DOPCS and DOPGT.

REFERENCES

(1) CNO Ltr. Ser. 00502P96, CNO FY-1972 Study Program, 29 Nov 1971.

(2) Center for Naval Analyses, Institute of Naval Studies Study 38, "Naval Aircraft Rework Facility Study–An Applied Model for Workload Planning Budgeting," 1 Jun 1972.

(3) Center for Naval Analyses, Research Contribution 212, "User's Guide to the NARF Workload Planning and Budgeting Model," Jan 1973.

(4) Mathematical Programming System/360, Application Description, IBM manual H20-0136-3.

(5) Mathematical Programming System/360, (360A-CO-14X) Version 2, Control Language, User's Manual, IBM manual H20-0290-3.

(6) Mathematical Programming System/360, (360A-CO-14X) Version 2, Linear and Separable Programming, User's Manual, IBM manual H20-0476-1.

APPENDIX A

PROGRAM LISTINGS

ANNEX A-1

INPUT1

```
  START CONVERSION * IBM TO CDC * 029 TO 026 HOLLERITH
//PGNAIN1 JOB (3525,20M,5,5),CLASS=L
//COB     EXEC COBACLG,AREA=BUPERS
//COB.SYSIN  DD  *

       IDENTIFICATION DIVISION.
       PROGRAM-ID. INPUT1.
       AUTHOR. JEFFREY BIRCH.
       ENVIRONMENT DIVISION.
       CONFIGURATION SECTION.
       INPUT-OUTPUT SECTION.
       FILE-CONTROL.
           SELECT MASTER-FILE
               ASSIGN TO UT-S-SYS001.
           SELECT MASTER-OUT-FILE
               ASSIGN TO UT-S-SYS002.
           SELECT CAP-DIS
               ASSIGN TO UT-S-SYS003.
           SELECT DIS-OUT-FILE
               ASSIGN TO UT-S-SYS004.
           SELECT RATE-FILE
               ASSIGN TO UT-S-SYS005.
           SELECT RATE-OUT-FILE
               ASSIGN TO UT-S-SYS006.
           SELECT PUNCH-IT
               ASSIGN TO UR-S-SYSPCH.
           RESERVE NO ALTERNATE AREA.

       DATA DIVISION.
       FILE SECTION.
       FD  MASTER-FILE.
           LABEL RECORDS ARE OMITTED.
           RECORDING MODE IS U.
           DATA RECORDS ARE MBTL, MEF, MASTER, MCKPK, MCKPK-HDR.


       01  MBTL.
           02  FILLER                  PICTURE IS X.
           02  MBTL-ER                 PICTURE IS X.
           02  FILLER                  PICTURE IS X(29).
A 31 CHARACTER RECORD USED TO CHECK FOR THE BEGINNING TAPE LABEL.


       01  MEF.
           02  FILLER                  PICTURE IS X.
           02  MEF-CHAR                PICTURE IS X.
           02  FILLER                  PICTURE IS X.
A 3 CHARACTER RECORD USED TO CHECK FOR THE EOF MARK.


       01  MASTER.
           02  FILLER                  PICTURE IS X.
           02  MASTER-GOOD.
               03  M-G-D  OCCURS 20 TIMES.
                   04  M-FIRST         PICTURE IS X.
                   04  FILLER          PICTURE IS X(299).
           02  FILLER                  PICTURE IS X.
A 6002 CHARACTER RECORD USED TO STORE ONE BLOCK OF DATA FROM THE MASTER FILE
PLUS THE BEGINNING AND ENDING BLOCK IDENTIFIERS.
```

```
       FD  CAP-DIS,
           LABEL RECORDS ARE OMITTED,
           RECORDING MODE IS U,
           DATA RECORDS ARE DBTL, DEF, DISTRIB, DCKPK, DCKPK-HDR,


       01  DBTL,
           02  FILLER                  PICTURE IS X,
           02  DBTL-EB                 PICTURE IS X,
           02  FILLER                  PICTURE IS X(29),
A 31 CHARACTER RECORD USED TO CHECK FOR THE BEGINNING TAPE LABEL,

       01  DEF,
           02  FILLER                  PICTURE IS X,
           02  DEF-CHAR                PICTURE IS X,
           02  FILLER                  PICTURE IS X,
A 3 CHARACTER RECORD USED TO CHECK FOR THE EOF MARK,

       01  DISTRIB,
           02  FILLER                  PICTURE IS X,
           02  DIS-GOOD,
               03  D-G-D  OCCURS 20 TIMES,
                   04  CAP-REC,
                       05  D-FIRST PICTURE IS X,
                       05  FILLER  PICTURE IS X(7),
                       05  CARD-TYPE,
                                    PICTURE IS 9,
                       05  FILLER  PICTURE IS X(71),
                   04  FILLER      PICTURE IS X(20),
           02  FILLER                  PICTURE IS X,
A 4002 CHARACTER RECORD USED TO STORE ONE BLOCK OF DATA FROM THE CAPACITY
AND DISTRIBUTION FILE PLUS THE BEGINNING AND ENDING BLOCK IDENTIFIERS,

       FD  RATE-FILE,
           LABEL RECORDS ARE OMITTED,
           RECORDING MODE IS U,
           DATA RECORDS ARE RBTL, REF, RATE, RCKPT-HDR, RCKPT,

       01  RBTL,
           02  FILLER                  PICTURE IS X,
           02  RBTL-EB                 PICTURE IS X,
           02  FILLER                  PICTURE IS X(29),
A 31 CHARACTER RECORD USED TO CHECK FOR THE BEGINNING TAPE LABEL,

       01  REF,
           02  FILLER                  PICTURE IS X,
           02  REF-CHAR                PICTURE IS X,
           02  FILLER                  PICTURE IS X,
A 3 CHARACTER RECORD USED TO CHECK FOR THE EOF MARK,

       01  RATE,
           02  FILLER                  PICTURE IS X,
           02  RATE-GOOD,
               03  R-G-D  OCCURS 20 TIMES,
                   04  R-FIRST     PICTURE IS X,
```

```
                04  FILLER          PICTURE IS X(199).
            02  FILLER              PICTURE IS X.
A 4002 CHARACTER RECORD USED TO STORE ONE BLOCK OF DATA FROM THE COST RATE
FILE PLUS THE BEGINNING AND ENDING BLOCK IDENTIFIERS.

        FD  MASTER-OUT-FILE,
            LABEL RECORDS ARE STANDARD,
            RECORDING MODE IS F,
            BLOCK CONTAINS 20 RECORDS,
            RECORD CONTAINS 300 CHARACTERS,
            DATA RECORDS ARE MASTER-OUT.
        01  MASTER-OUT              PICTURE IS X(300).
FILE DESCRIPTION FOR THE OUTPUT MASTER TAPE FILE.

        FD  DIS-OUT-FILE,
            LABEL RECORDS ARE STANDARD,
            RECORDING MODE IS F,
            BLOCK CONTAINS 20 RECORDS,
            RECORD CONTAINS 100 CHARACTERS,
            DATA RECORDS ARE DISTRIB-OUT.
        01  DISTRIB-OUT             PICTURE IS X(100).
FILE DESCRIPTION FOR THE OUTPUT DISTRIBUTION TAPE FILE.


        FD  RATE-OUT-FILE,
            LABEL RECORDS ARE STANDARD,
            RECORDING MODE IS F,
            BLOCK CONTAINS 20 RECORDS,
            RECORD CONTAINS 200 CHARACTERS,
            DATA RECORDS ARE RATE-OUT.
        01  RATE-OUT,
            02  R-O-TEC             PICTURE IS X(4).
            02  R-O-WC              PICTURE IS XX.
            02  R-O-FS              PICTURE IS X.
            02  R-O-N               PICTURE IS X.
            02  R-O-FILLER          PICTURE IS X(192).
FILE DESCRIPTION FOR THE OUTPUT COST RATE TAPE FILE.

        FD  PUNCH-IT,
            RECORD CONTAINS 80 CHARACTERS,
            LABEL RECORDS ARE OMITTED,
            DATA RECORD IS PUNCH-CARD.
        01  PUNCH-CARD,
            02  FILLER-A            PICTURE IS X(16).
            02  CAP-RECORD  OCCURS 5 TIMES,
                03  FILLER          PICTURE IS X.
                03  CAP-DATA        PICTURE IS X(7).
            02  FILLER-B            PICTURE IS X(24).
FILE DESCRIPTION FOR THE OUTPUT CAPACITY CARD FILE.


        WORKING-STORAGE SECTION,
        77  K                       PICTURE IS 9.
        77  I                       PICTURE IS 9(2), COMP-3.
        77  J                       PICTURE IS 9(2), COMP-3.
        77  II                      PICTURE IS 9.
        77  JJ                      PICTURE IS 9.
        77  KK                      PICTURE IS 9(7),  VALUE IS 9999999.
        77  ISKIP                   PICTURE IS 9  VALUE IS ZERO.
```

```
       77  LABEL-COUNT                 PICTURE IS 9, VALUE IS ZERO,
LEVEL 77 ENTRIES USED AS COUNTERS AND SUBSCRIPTS.


       01  CAPACITY.
           02  FILLER                  PICTURE IS X(10).
           02  CAPACITIES              PICTURE IS X(56).
           02  FILLER                  PICTURE IS X(34).
CAPACITY IS USED TO STORE ONE CAPACITY RECORD.


       01  CAP-TABLE.
           02  CAP-WARE  OCCURS 7 TIMES.
              03  CAP-YEAR  OCCURS 5 TIMES.
                  04  CAP-SHOP  OCCURS 8 TIMES.
                             PICTURE IS X(7).
CAP-TABLE IS USED TO STORE THE 35 CAPACITY RECORDS FROM THE CAPACITY AND
DISTRIBUTION FILE.

       01  WORK-AREA-M.
           02  FILLER                  PICTURE IS X(300).
       01  WORK-AREA-D.
           02  FILLER                  PICTURE IS X(100).
       01  WORK-AREA-R.
           02  R-G-N                   PICTURE IS X.
           02  R-G-WC                  PICTURE IS XX.
           02  R-G-TEC                 PICTURE IS X(4).
           02  R-G-FS                  PICTURE IS X.
           02  R-G-FILLER              PICTURE IS X(192).
WORK-AREAS-M, D, AND R ARE USED TO STORE ONE RECORD FROM THE MASTER, DISTRIBUT-
ION AND RATE FILES RESPECTIVELY.

       01  360-TABLE.
           02 FILLER                        PICTURE X(24)
              VALUE        ≠/STUVWXYZ,ABCDEFGHI,0 ><≠.
           02 FILLER                        PICTURE X
              VALUE QUOTE.
           02 FILLER                         PICTURE IS X(5).
              VALUE ≠=)+-(≠.
360-TABLE STORES THE CHARACTER CODES USED IN THE COBOL TRANSFORM VERB,
SEE FIGURE 1.

       PROCEDURE DIVISION.


PROCESS THE MASTER FILE.


           OPEN INPUT MASTER-FILE.
           OPEN OUTPUT MASTER-OUT-FILE.
OPENS BOTH INPUT AND OUTPUT FILES.


       READ-MASTER-FILE.
           READ MASTER-FILE, AT END GO TO END-PROGRAM.
           MOVE 1 TO I.
           IF MBTL-EB = ≠<≠  GO TO MLABEL-CHECK.
           IF MBF-CHAR = ≠↓≠  GO TO READ-MASTER-FILE.
           IF ISKIP = 0,  MOVE 1 TO ISKIP,  ADD 1 TO I.
```

```
STEP 1,  THE SECOND CHARACTER OF LABEL AND EOF MARK RECORDS IDENTIFIES THE
RECORDS,
ISKIP IS USED SO THAT THE FIRST RECORD OF THE FIRST BLOCK IS SKIPPED AS IT IS
ONLY A CONTROL RECORD AND IS NOT NEEDED IN THE OUTPUT TAPE,

       EOFM-SEARCH,
           IF M-FIRST (I) = ≠+≠  GO TO CLOSE-MASTER,
           MOVE M-G-D (I) TO WORK-AREA-M,
           TRANSFORM WORK-AREA-M
           FROM  ≠ABCDEFGHI,/STUVWXYZ, 0)-+*=;]∨≠
               TO 360-TABLE,
STEP 2,  TRANSFORM RECORDS,


       WRITE-MASTER-FILE,
           WRITE MASTER-OUT FROM WORK-AREA-M,
           IF I < 20  ADD 1 TO I,  GO TO EOFM-SEARCH
               ELSE  GO TO READ-MASTER-FILE,
       MLABEL-CHECK,
           IF LABEL-COUNT < 1, ADD 1 TO LABEL-COUNT,
               GO TO READ-MASTER-FILE,
           ELSE
               GO TO CLOSE-MASTER,
STEP 3 AND 4,  WRITE TRANSFORMED RECORDS AND CHECK FOR SHORT BLOCK,
LABEL-COUNT IS USED TO COUNT THE NUMBER OF LABELS PROCESSED FOR THE FILE,
THE END-OF-FILE MAY ALSO BE DETECTED WHEN THE SECOND LABEL IS FOUND,

       CLOSE-MASTER,
           CLOSE MASTER-FILE, MASTER-OUT-FILE,
CLOSE THE FILES,


PROCESS THE CAPACITY AND DISTRIBUTION FILE,


       OPEN-CAP-DIS,
           MOVE 1 TO II, JJ
           MOVE ZERO TO LABEL-COUNT,
           OPEN INPUT CAP-DIS,
           OPEN OUTPUT DIS-OUT-FILE,  PUNCH-IT,
           MOVE 0 TO ISKIP,
OPEN FILES AND SET COUNTERS,


       READ-DIS-FILE,
           READ CAP-DIS AT END GO TO END-PROGRAM,
           MOVE 1 TO I,
           IF DBTL-EB = ≠<≠  GO TO DLABEL-CHECK,
           IF DEF-CHAR = ≠+≠  GO TO READ-DIS-FILE,
           IF ISKIP = 0,  MOVE 1 TO ISKIP,  ADD 1 TO I,
STEP 1,  SIMILAR TO THAT DESCRIBED ABOVE FOR THE MASTER RECORD,

       EOFD-SEARCH,
           IF D-FIRST (I) = ≠+≠  GO TO WRITE-CAPACITY,
           MOVE D-G-D (I) TO WORK-AREA-D,
           TRANSFORM WORK-AREA-D
           FROM  ≠ABCDEFGHI,/STUVWXYZ, 0)-+*=;]∨≠
               TO 360-TABLE,
STEP 2,  SIMILAR TO THAT DESCRIBED ABOVE FOR THE MASTER RECORD,
```

```
        CHECK-CARD-TYPE.
            IF CARD-TYPE (I) = 5  GO TO MOVE-CAP-TABLE.
CKECK TO DETERMINE WHETHER RECORD IS A CAPACITY OR DISTRIBUTION RECORD,
IF IT IS A CAPACITY RECORD IT IS STORED IN CAP-TABLE.

        WRITE-DIS-FILE.
            WRITE DISTRIB-OUT FROM WORK-AREA-D.
        CHECK-I.
            IF I < 20  ADD 1 TO I,  GO TO EOFD-SEARCH
                ELSE  GO TO READ-DIS-FILE.
STEP 3,  SIMILAR TO THAT DESCRIBED ABOVE FOR THE MASTER RECORD,

        MOVE-CAP-TABLE.
            MOVE WORK-AREA-D TO CAPACITY.
            MOVE CAPACITIES TO CAP-YEAR (II, JJ).
            IF JJ <     5,  ADD 1 TO JJ
                ELSE
            IF II <     7  ADD 1 TO II, MOVE 1 TO JJ.
            GO TO CHECK-I.
STORE CAPACITY RECORDS IN CAP-TABLE.

        DLABEL-CHECK.
            IF LABEL-COUNT < 1, ADD 1 TO LABEL-COUNT.
                GO TO READ-DIS-FILE.
            ELSE
                GO TO WRITE-CAPACITY.
LABEL-COUNT IS USED AS DESCRIBED ABOVE FOR THE MASTER RECORD,

        WRITE-CAPACITY.
            MOVE 1 TO I, J, K.
            MOVE SPACES TO FILLER-A, FILLER-B.
        MOVE-CAP-SHOP.
            MOVE CAP-SHOP (I, J, K) TO CAP-DATA (J).
            IF J < 5,  ADD 1 TO J,  GO TO MOVE-CAP-SHOP
                ELSE
            WRITE PUNCH-CARD.
            IF K <      8, ADD 1 TO K, MOVE 1 TO J, GO TO MOVE-CAP-SHOP.
            IF K =      8, MOVE KK TO CAP-DATA (1), CAP-DATA (2),
            CAP-DATA (3), CAP-DATA (4), CAP-DATA (5),
                WRITE PUNCH-CARD.
            IF I <      7, ADD 1 TO I, MOVE 1 TO K, J,
                GO TO MOVE-CAP-SHOP.
THE CAPACITY RECORDS ARE PUNCHED IN A FORMAT ACCEPTABLE TO THE MATRIX
GENERATOR PROGRAM,
THIS FORMAT IS DESCRIBED FULLY IN REFERENCE    , PAGE    .

        CLOSE-DIS.
            CLOSE DIS-OUT-FILE, PUNCH-IT, CAP-DIS.
CLOSE THE FILES.

PROCESS THE COST RATE FILE.

        OPEN-RATE-FILE.
```

```
            OPEN INPUT RATE-FILE,
            OPEN OUTPUT RATE-OUT-FILE,
            MOVE ZERO TO LABEL-COUNT,
            MOVE 0 TO ISKIP,
OPEN FILES AND SET COUNTERS,


        READ-RATE-FILE,
            READ RATE-FILE AT END GO TO END-PROGRAM,
            MOVE 1 TO I,
            IF RBTL-BB = ≠<≠  GO TO RLABEL-CHECK,
            IF REC-CHAR = ≠↓≠  GO TO READ-RATE-FILE,
            IF ISKIP = 0,  MOVE 1 TO ISKIP,  ADD 1 TO I,
STEP 1,  SIMILAR TO THAT DESCRIBED ABOVE FOR THE MASTER RECORD,

        EOFR-SEARCH,
            IF R-FIRST (I) = ≠+≠  GO TO CLOSE-RATE,
            MOVE R-G-D (I) TO WORK-AREA-R,
            TRANSFORM WORK-AREA-R
            FROM   ≠ABCDEFGHI,/STUVWXYZ, 0)↓+*=;]↵≠
                TO 360-TABLE,
STEP 2,  SIMILAR TO THAT DESCRIBED ABOVE FOR THE MASTER RECORD,

        WRITE-RATE-FILE,
            MOVE R-G-N      TO R-O-N,
            MOVE R-G-WC      TO R-O-WC,
            MOVE R-G-TEC      TO R-O-TEC,
            MOVE R-G-FS      TO R-O-FS,
            MOVE R-G-FILLER      TO R-O-FILLER,
            EXAMINE R-O-TEC REPLACING ALL ZEROS BY SPACES,
            WRITE RATE-OUT,
            IF I < 20  ADD 1 TO I,  GO TO EOFR-SEARCH
                ELSE  GO TO READ-RATE-FILE,
STEP 3,  THE RATE VARIABLES OF TEC, PROGRAM, SUBPROGRAM, FUND-CODE,
AND DRP ARE REORDERED ON THE OUTPUT RATE FILE,  THIS IS DONE SO THAT THEIR
ORDER WILL CORRESPOND WITH THAT ON THE OUTPUT MASTER AND DISTRIBUTION FILES,

        RLABEL-CHECK,
            IF LABEL-COUNT < 1, ADD 1 TO LABEL-COUNT,
                GO TO READ-RATE-FILE,
            ELSE
                GO TO CLOSE-RATE,
LABEL-COUNT IS USED AS DESCRIBED ABOVE FOR THE MASTER RECORD,


        CLOSE-RATE,
            CLOSE RATE-FILE,  RATE-OUT-FILE,
CLOSE THE FILES,


        END-PROGRAM,
            STOP RUN,

//GO,SYS001   DD DSN=PCNA.UMASTFI1,DISP=(OLD,KEEP),
// UNIT=(SEVEN,,DEFER),LABEL=(,NL),
// DCB=TRTCH=T,VOL=SER=992077
//GO,SYS003   DD DSN=PCNA.UCAPDIS1,DISP=(OLD,KEEP),
// UNIT=AFF=SYS001,LABEL=(,NL),
```

```
// DCB=TRTCH=T,VOL=SER=992409
//GO.SYS005   DD DSN=PCNA.UCOSTRA1,DISP=(OLD,KEEP),
// UNIT=AFF=SYS001,LABEL=(,NL),
//  DCB=TRTCH=T,VOL=SER=992071
//GO.SYS002  DD  DSN=PCNA.UMASTFI2,DISP=(NEW,PASS),UNIT=2314,
// SPACE=(TRK,200)
//GO.SYS004  DD  DSN=PCNA.UCAPDIS2,DISP=(NEW,PASS),UNIT=2314,
// SPACE=(TRK,200)
//GO.SYS006  DD  DSN=PCNA.UCOSTRA2,DISP=(NEW,PASS),UNIT=2314,
// SPACE=(TRK,200)
//GO.SYSPCH DD  SYSOUT=B
//SORT1 EXEC  PGM=IERRCOOO,REGION=100K
//SORTLIB    DD DSN=SYS1.SORTLIB,DISP=SHR
//SYSPRINT   DD SYSOUT=A
//SYSOUT DD SYSOUT=A
//SORTW<01  DD UNIT=2314,SPACE=(TRK,(150),,CONTIG)
//SORTW<02  DD UNIT=2314,SPACE=(TRK,(150),,CONTIG)
//SORTW<03  DD UNIT=2314,SPACE=(TRK,(150),,CONTIG)
//SORTW<04  DD UNIT=2314,SPACE=(TRK,(150),,CONTIG)
//SORTIN    DD DSN=PCNA.UMASTFI2,DISP=(OLD,DELETE)
//SORTOUT   DD DSN=PCNA.UMASTFI3,DISP=(OLD,KEEP),UNIT=(TAPE,,DEFER),
//  LABEL=(,SL),DCB=(RECFM=FB,BLKSIZE=6000,LRECL=300),VOL=SER=990006
//SYSIN  DD  *
 SORT FIELDS=(1,10,CH,A)
/*
//SORT2 EXEC  PGM=IERRCOOO,REGION=100K
//SORTLIB    DD DSN=SYS1.SORTLIB,DISP=SHR
//SYSPRINT   DD SYSOUT=A
//SYSOUT DD SYSOUT=A
//SORTW<01  DD UNIT=2314,SPACE=(TRK,(150),,CONTIG)
//SORTW<02  DD UNIT=2314,SPACE=(TRK,(150),,CONTIG)
//SORTW<03  DD UNIT=2314,SPACE=(TRK,(150),,CONTIG)
//SORTW<04  DD UNIT=2314,SPACE=(TRK,(150),,CONTIG)
//SORTIN    DD DSN=PCNA.UCAPDIS2,DISP=(OLD,DELETE)
//SORTOUT   DD DSN=PCNA.UCAPDIS3,DISP=(OLD,KEEP),UNIT=(TAPE,,DEFER),
//  LABEL=(,SL),DCB=(RECFM=FB,BLKSIZE=2000,LRECL=100),VOL=SER=992410
//SYSIN  DD  *
 SORT FIELDS=(1,10,CH,A)
/*
//SORT3 EXEC  PGM=IERRCOOO,REGION=100K
//SORTLIB    DD DSN=SYS1.SORTLIB,DISP=SHR
//SYSPRINT   DD SYSOUT=A
//SYSOUT DD SYSOUT=A
//SORTW<01  DD UNIT=2314,SPACE=(TRK,(150),,CONTIG)
//SORTW<02  DD UNIT=2314,SPACE=(TRK,(150),,CONTIG)
//SORTW<03  DD UNIT=2314,SPACE=(TRK,(150),,CONTIG)
//SORTW<04  DD UNIT=2314,SPACE=(TRK,(150),,CONTIG)
//SORTIN    DD DSN=PCNA.UCOSTRA2,DISP=(OLD,DELETE)
//SORTOUT   DD DSN=PCNA.UCOSTRA3,DISP=(OLD,KEEP),UNIT=(TAPE,,DEFER),
//  LABEL=(,SL),DCB=(RECFM=FB,BLKSIZE=4000,LRECL=200),VOL=SER=990222
//SYSIN  DD  *
 SORT FIELDS=(1,10,CH,A)
JOB CONTROL LANGUAGE FOR INPUT1,
THE SETUP FOR THE UTILITY SORTS IS CONTAINED IN THE JOB CONTROL LANGUAGE.


  CONVERSION COMPLETE,      490 CARDS PUNCHED.
```

ANNEX A-2

INPUT2

```
SEQUENCE     1 STARTED  PRINTING 12/04/72 AT 130041 ON LP00
W=  00  PRINT,249,BIRCH,S
//PCNAIN1 JOB (3525,20M,5,5),CLASS=I,REGION=100K
//COB     EXEC COBACLG,AREA=AUPERS
//COB.SYSIN  DD  *


00101  IDENTIFICATION DIVISION.
00102  PROGRAM-ID. INPUT2.
00103  AUTHOR. JEFFREY BIRCH.
00104  DATE-COMPILED.
00105  ENVIRONMENT DIVISION.
00106  CONFIGURATION SECTION.
00109  INPUT-OUTPUT SECTION.
00110  FILE-CONTROL.
00111       SELECT MASTER-FILE
                  ASSIGN TO UT-S-SYS001.
00113       SELECT CAP-DISTRI
                  ASSIGN TO UT-S-SYS002.
00115       SELECT RATE-FILE
                  ASSIGN TO UT-S-SYS003.
            SELECT DATA-BASE-FILE
                  ASSIGN TO UT-S-SYS004.
            SELECT DICTIONARY-FILE,
                  ASSIGN TO UR-S-SYSPCH.
               RESERVE NO ALTERNATE AREA.
00201       SELECT PRINTFILE
                  ASSIGN TO UR-S-SYSPRT,
               RESERVE NO ALTERNATE AREA.
            SELECT GA-RATE-FILE
                  ASSIGN TO UR-S-SYSRED,
               RESERVE NO ALTERNATE AREA.
INPUT2 IS WRITTEN IN ANSI COBOL.
IT UTILIZES THREE INPUT FILES AND PRODUCES FOUR OUTPUT FILES.


00203  DATA DIVISION.
00204  FILE SECTION.
00205  FD  MASTER-FILE,
           RECORDING MODE IS F,
00207      BLOCK CONTAINS 20 RECORDS,
00208      RECORD CONTAINS 300 CHARACTERS,
00209      LABEL RECORDS ARE STANDARD,
00210      DATA RECORD IS MASTER.
00211  01  MASTER.
00212       02  MID-NAME.
                03  MID-NAME2.
                    04  MTEC          PICTURE IS X(4).
                    04  MWORK-CODE.
                        05  MPRO      PICTURE IS X.
                        05  MSUB      PICTURE IS X.
                03  MCUST             PICTURE IS X.
                03  MCUSIN  REDEFINES MCUST.
                                       PICTURE IS 9.
00218           03  MNARF             PICTURE IS X.
            02  FILLER                PICTURE IS XX.
            02  MTMS                  PICTURE IS X(15).
            02  FILLER                PICTURE IS X(55).
00220       02  MFIELDA  OCCURS 5 TIMES.
```

```
                03  MTQS               PICTURE IS  9(4).
                03  FILLER             PICTURE IS X(3).
                03  MTME               PICTURE IS 9(4).
                03  FILLER             PICTURE IS X(3).
            02  M-FILLER1.
                03  MFIELDD OCCURS 5 TIMES.
                    04  MQTYME         PICTURE IS 9(3).
                    04  MNORM          PICTURE IS 9(5).
                    04  MQTYNM         PICTURE IS 9(3).
            02  M-FILLER2 REDEFINES M-FILLER1.
                03  FILLER  OCCURS 5 TIMES.
                    04  MQTYMEFT       PICTURE IS 9V99.
                    04  FILLER         PICTURE IS 9(5).
                    04  MQTYNMFT       PICTURE IS 9V99.
            02  FILLER                 PICTURE IS X(95).
FILE DESCRIPTION FOR THE MASTER FILE.  SEE FIGURE 2.


00306  FD   CAP-DISTRI.
            RECORDING MODE IS F,
00308       BLOCK CONTAINS 20 RECORDS,
            RECORD CONTAINS 100 CHARACTERS,
00310       LABEL RECORDS ARE STANDARD,
00311       DATA RECORDS ARE            DISTRIB.
00401  01   DISTRIB.
            02  DID-NAME2.
                03  DTEC               PICTURE IS X(4).
                03  DWORK-CODE.
                    04  DPRO           PICTURE IS X.
                    04  DSUB           PICTURE IS X.
00406       02  FILLER                 PICTURE IS X.
00407       02  DNARF                  PICTURE IS X.
00408       02  DCARD-TY               PICTURE IS X.
00409       02  FILLER                 PICTURE IS X.
            02  DFIELDA                PICTURE IS X(27).
            02  FILLER                 PICTURE IS X(63).
FILE DESCRIPTION FOR THE DISTRIBUTION FILE.  SEE FIGURE 3.


00414  FD   RATE-FILE,
            RECORDING MODE IS F,
00416       BLOCK CONTAINS 20 RECORDS,
            RECORD CONTAINS 200 CHARACTERS,
00419       LABEL RECORDS ARE STANDARD,
00420       DATA RECORD IS RATE.
00501  01   RATE.
            02  RATE-ID.
                03  RID-NAME2.
                    04  RTEC           PICTURE IS X(4).
                    04  RWORK-CODE.
                        05  RPRO       PICTURE IS X.
                        05  RSUB       PICTURE IS X.
                03  RFS                PICTURE IS  X.
                03  RNARF              PICTURE IS X.
            02  RFY                    PICTURE IS  99.
            02  FILLER                 PICTURE IS X(40).
            02  RFIELDC                PICTURE IS X(145).
            02  FILLER                 PICTURE IS X(5).
FILE DESCRIPTION FOR THE RATE FILE.  SEE FIGURE 4.
```

```
       FD  DATA-BASE-FILE.
           RECORDING MODE IS F,
           LABEL RECORDS ARE STANDARD,
           DATA RECORD IS DATA-BASE.
       01  DATA-BASE                  PICTURE IS X(96).
FILE DESCRIPTION FOR THE DATA BASE FILE.  SEE FIGURE 5.


00617  FD  PRINTFILE,
           RECORDING MODE IS F,
00619      LABEL RECORDS ARE OMITTED,
00618      RECORD CONTAINS 132 CHARACTERS,
00620      DATA RECORD IS PRINTALL.
00701  01  PRINTALL.
           02  FILLER                  PICTURE IS X(2).
           02  P-MESSAGE               PICTURE IS X(40).
           02  P-DATA                  PICTURE IS X(8).
           02  FILLER                  PICTURE IS X(82).
FILE DESCRIPTION FOR THE PRINT FILE.
USED TO DISPLAY ON THE PRINTER THE MESSAGES CORRESPONDING TO EITHER NO
DISTRIBUTION RECORD OR NO RATE RECORD BEING FOUND FOR A PARTICULAR
MASTER RECORD.


       FD  DICTIONARY-FILE,
           RECORDING MODE IS F,
           LABEL RECORDS ARE OMITTED,
           DATA RECORD IS DICTIONARY.
       01  DICTIONARY.
           02  DICT-ID                 PICTURE IS XX.
           02  DICT-FS                 PICTURE IS X.
           02  DFILLER1                PICTURE IS X.
           02  DICT-TEC                PICTURE IS X(4).
           02  DICT-TMS                PICTURE IS X(15).
           02  DFILLER2                PICTURE IS X(57).
FILE DESCRIPTION FOR THE DICTIONARY FILE.
THIS IS A CARD FILE PUNCHED BY INPUT2 TO GIVE THE RELATIONSHIP BETWEEN
THE FOUR CHARACTER TEC CODE AND THE NEW TWO CHARACTER DATA BASE CODE.
THE FORMAT FOR THE DICTIONARY CARDS IS DESCRIBED IN REFERENCE B, PAGE    .


       FD  GA-RATE-FILE,
           RECORDING MODE IS F,
           LABEL RECORDS ARE OMITTED,
           DATA RECORD IS GA-RATES.
       01  GA-RATES.
           02  GA-INFO                 PICTURE IS X(25).
FILE DESCRIPTION FOR THE CURRENT G A RATE FILE.
THE FORMAT FOR THESE CARDS IS DESCRIBED IN REFERENCE B,  PAGE    .


00703  WORKING-STORAGE SECTION.
       77  II                          PICTURE IS 9999.
       77  JJ                          PICTURE IS 9999.
       77  KK                          PICTURE IS 999.
       77  I                           PICTURE IS 9999.
       77  J                           PICTURE IS 9999.
```

```
        77  K                          PICTURE IS S999.
        77  L                          PICTURE IS S999.
        77  M                          PICTURE IS S9(4).
        77  N                          PICTURE IS S9(4).
        77  NDT                        PICTURE IS 99.
        77  NRT                        PICTURE IS 99.
        77  EOFD                       PICTURE IS 9.
        77  EOFR                       PICTURE IS 9.
00711   77  RSW                        PICTURE IS S99.
        77  HIGH-V                     PICTURE IS S9(4).
        77  LOW-V                      PICTURE IS S9(4).
        77  SUB                        PICTURE IS S9(4).
        77  GARATE                     PICTURE IS S9V9(4).
THE ABOVE LEVEL 77 VARIABLES ARE USED AS SUBSCRIPTS AND COUNTERS.
THERE PURPOSES WILL BE DESCRIBED LATER IN THE PROCEDURE DIVISION.


        01  IDATA-RECORD.
            02  IDATA-ID               PICTURE IS XX.
            02  IDATA-FS               PICTURE IS X.
            02  IDATA-YR               PICTURE IS 9.
            02  IDATA-WC               PICTURE IS XX.
            02  IDATA-C                PICTURE IS X.
            02  IDATA-N                PICTURE IS X.
            02  IDATA                  PICTURE IS X(85).
          02  FILLER                 PICTURE IS XXX.   VALUE IS SPACES.
STORAGE AREA FOR DATA BASE RECORD.


00601   01  INPUT-RECORD.
00603       02  IFIELD  OCCURS 5 TIMES.
                03  ISHOPS             PICTURE IS X(27).
                03  ITQS               PICTURE IS  9(4).
                03  ITME               PICTURE IS  9(4).
                03  IFIELDB.
                    04  FILLER         PICTURE IS X(3).
                    04  INORM          PICTURE IS S9(5).
                    04  FILLER         PICTURE IS X(3).
                03  IREQ               PICTURE IS  9(4)V99.
                03  IRATE.
                    04  IFIELDC.
                        05  I-RATES PICTURE IS 9(13).
                        05  IGA-RATE
                                        PICTURE IS 9V9(4).
                        05  I-UMRATE
                                        PICTURE IS 9(6).
                    04  ITOT-COST      PICTURE IS 9(7)V99.
INPUT-RECORD CONTAINS THE FIVE DATA BASE RECORDS CORRESPONDING TO EACH
MASTER RECORD.


        01  DFIELD                     PICTURE IS X(27).
DFIELD CONTAINS NINE THREE CHARACTER FIELDS REPRESENTING THE NINE SHOP
CATEGORY DISTRIBUTION FACTORS.


        01  MID-RATE.
            02  MPSCNF.
                03  MWC                PICTURE IS X(2).
```

```
                03  MFUSO              PICTURE IS X.
                03  MNAR               PICTURE IS X.
MID-RATE IS A FOUR CHARCATER IDENTIFICATION MADE FROM MASTER RECORD VARIABLES
AND IS USED TO LOCATE THE ASSOCIATED RATE RECORD STORED IN COST-TABLE.


00713  01  MSORT-KEY.
00714          02  MTEC-SK            PICTURE IS X(4).
               02  MS-KEY.
                   03  MWORK-CODE-SK   PICTURE IS XX.
                   03  MNARF-SK        PICTURE IS X.
MSORT-KEY IS USED TO LOCATE  DISTRIBUTION RECORDS IN THE DISTRIBUTION
TABLE (D-TABLE<.


           01  COST-TABLE.
               02  COST-REC OCCURS  300 TIMES.
                   03  COST-ID.
                       04  CWORK-CODE.
                           05  CPRO    PICTURE IS X.
                           05  CSUB    PICTURE IS X.
                       04  CFS         PICTURE IS  X.
                       04  CNARF       PICTURE IS X.
                   03  CFY             PICTURE IS 99.
                   03  CFIELDC.
                       04  COST-RATES  OCCURS 5 TIMES.
                           05  COST-DATA.
                               06  CDIRLABCT
                                        PICTURE IS 99V99.
                               06  CDIRMATFJ
                                        PICTURE IS 999V99.
                               06  CPROOVHD
                                        PICTURE IS 99V99.
                           05  CGAOVHD
                                        PICTURE IS 99V99.
                           05  CUNIMAT
                                        PICTURE IS 9(6).
                           05  CGFM
                                        PICTURE IS 9(6).
COST-TABLE IS USED TO STORE UP TO 300 RATE RECORDS.
THEY ARE THOSE RATE RECORDS WITH PROGRAMS EQUAL TO F, H, L, P, R, V, OR Y.


           01  DIS-TABLE.
               02  DIS-REC OCCURS  50 TIMES.
                   03  DT-ID.
                       04  DTWORK-CODE.
                           05  DTPRO   PICTURE IS X.
                           05  DTSUB   PICTURE IS X.
                       04  DTNARF      PICTURE IS X.
                   03  DTFIELDA.       PICTURE IS X(27).
DIS-TABLE CONTAINS UP TO 50 DISTRIBUTION RECORDS.THEY ARE
THOSE DISTRIBUTION RECORDS WITH PROGRAMS EQUAL TO F, H, OR L.


           01  P-ID-NAME2.
               02  P-TEC               PICTURE IS X(4).
               02  P-W-C               PICTURE IS X(2).
P-ID-NAME2 CONTAINS THE TEC,PROGRAM,AND SUBPROGRAM REPRESENTING THE MOST
```

RECENTLY USED MASTER RECORD.

```
       01  D-BUFFER.
           02  D-B-ID2                 PICTURE IS X(6).
           02  D-B-N                   PICTURE IS X.
           02  D-B-DATA                PICTURE IS X(27).
```

D-BUFFER IS AN INTERMEDIATE STORAGE AREA FOR THE PRIMARY DATE REPRESENTING THE CURRENT DISTRIBUTION RECORD IN CORE.

```
       01  D-TABLE.
           02  D-DATA  OCCURS 12 TIMES.
               03  D-T-ID.
                   04  D-T-ID2         PICTURE IS X(6).
                   04  D-T-N           PICTURE IS X.
               03  D-T-DATA            PICTURE IS X(27).
```

D-TABLE IS USED TO STORE UP TO 12 DISTRIBUTION RECORDS. ALL RECORDS STORED IN D-TABLE HAVE IDENTICAL VALUES FOR THE VARIABLES TEC, PROGRAM AND SUBPROGRAM.

```
       01  R-BUFFER.
           02  R-B-ID.
               03  R-B-ID2             PICTURE IS X(6).
               03  R-B-FS              PICTURE IS X.
               03  R-B-N               PICTURE IS X.
           02  R-B-FY                  PICTURE IS 99.
           02  R-B-FIELDC              PICTURE IS X(145).
```

R-BUFFER IS AN INTERDEDIATE STORAGE AREA FOR THE CURRENT RATE RECORD IN CORE.

```
       01  R-TABLE.
           02  R-DATA  OCCURS 24 TIMES.
               03  R-T-ID.
                   04  R-T-ID2         PICTURE IS X(6).
                   04  R-T-FS          PICTURE IS X.
                   04  R-T-N           PICTURE IS X.
               03  R-T-FY              PICTURE IS 99.
               03  R-T-FIELDC.
                   04  R-T-FIELDS  OCCURS 5 TIMES.
                       05  R-T-RATES.
                           06  RDLC
                                        PICTURE IS 99V99.
                           06  RDM
                                        PICTURE IS 999V99.
                           06  RPO
                                        PICTURE IS 99V99.
                       05  RGAO
                                        PICTURE IS 99V99.
                       05  RUM
                                        PICTURE IS 9(6).
                       05  RGFM
                                        PICTURE IS 9(6).
```

R-TABLE IS USED TO STORE UP TO 24 RATE RECORDS ALL WITH IDENTICAL VALUES OF THE VARIABLES TEC, PROGRAM, AND SUBPROGRAM.

```
       01  IDENT.
```

```
           02  IDEN1                  PICTURE IS X.
           02  IDEN2                  PICTURE IS X.
IDENT CONTAINS THE CURRENT VALUE OF THE DATA BASE CODE.


       01  NEW-CODE2.
           02  FILLER                 PICTURE IS X(35).  VALUE IS
         #ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789#.
       01  NEW-CODE  REDEFINES NEW-CODE2.
           02  N-C OCCURS 35 TIMES PICTURE IS X.
THE ABOVE VARIABLES ARE USED IN COMPUTING THE DATA BASE CODE.


       01  FUND-SOURCE-ARRAY.
           02  FILLER                 PICTURE IS X(40). VALUE IS
         #1ANA2ANA3A A4FPA5H A6L A7T C8T C9T C=T A#.
           02  FILLER                 PICTURE IS X(40).  VALUE IS
         ##A A+A AAA ABA ACA ADA AEA AFA AGANAHA A#.
           02  FILLER                 PICTURE IS X(40).  VALUE IS
         #+Y D-Y DJY DKA ALN AMP ENT EOA APN AWA A#.
           02  FILLER                 PICTURE IS X(40).  VALUE IS
         #RP C(V I$V E N I/P ISL ATP EULPAVR AWL A#.
           02  FILLER                 PICTURE IS X(8).  VALUE IS
         #XR AYR A#.
       01  F-S-ARRAY  REDEFINES FUND-SOURCE-ARRAY.
           02  SUB-PRO  OCCURS 42 TIMES.
               03  F-S  OCCURS 4 TIMES,
                                       PICTURE IS X.
THE ABOVE VARIABLES ARE USED IN COMPUTING THE FUND-CODE FROM THE VARIABLES
CUSTOMER, PROGRAM, AND SUBPROGRAM.
SEE FIGURE 6.


       01  GA-RATE-TABLE.
           02  GA-RATE-T  OCCURS 7 TIMES.
               03  GA-R-T  OCCURS 5 TIMES,
                                       PICTURE IS 9V9(4).
GA-RATE-TABLE IS USED TO STORE THE CURRENT G A RATE CARDS.
THESE CARDS REPRESENT THE MOST CURRENT G A RATES AVAILABLE FOR THE NARFS.


00804  PROCEDURE DIVISION.


00806      OPEN INPUT MASTER-FILE, CAP-DISTRI, RATE-FILE.
           OPEN INPUT GA-RATE-FILE.
           OPEN OUTPUT  DATA-BASE-FILE,  PRINTFILE.
           OPEN OUTPUT  DICTIONARY-FILE.
           MOVE QUOTE TO F-S (34, 1).
          MOVE ZEROS TO R-BUFFER, D-BUFFER, RSW, EOFD, EOFR.
           MOVE SPACES TO DFILLER1, DFILLER2.
           MOVE SPACES TO P-ID-NAME2.
           MOVE SPACES TO PRINTALL.
           MOVE 1 TO L, K, II, JJ, M, N.
           MOVE 1 TO I.
STEP 1.  INITIALIZATION STEP.


       MOVE-GA-RATE.
```

```
           READ GA-RATE-FILE.  AT END GO TO READ-DISTR.
           MOVE GA-INFO TO GA-RATE-T (I).
           IF I <     7,  ADD 1 TO I,  GO TO MOVE-GA-RATE.
STEP 2.  THE CURRENT G A RATE CARDS ARE READ INTO THE GA-RATE-TABLE.


00901  READ-DISTR.
00902      READ CAP-DISTRI, AT END GO TO END-PROGRAM.
           IF DTEC =     SPACES MOVE DWORK-CODE TO DTWORK-CODE (N),
                                MOVE DNARF TO DTNARF (N),
                                MOVE DFIELDA TO DTFIELDA (N),
               ADD 1 TO N,  GO TO READ-DISTR.
           MOVE DID-NAME2 TO D-B-ID2.
           MOVE DNARF TO D-B-N.
           MOVE DFIELDA TO D-B-DATA.
STEP 3.  THESE INSTRUCTIONS MOVE THE DISTRIBUTION RECORDS WITH THE SPECIAL
PROGRAM VALUES OF F, H, OR L INTO DIS-TABLE.
ALSO, THE IDENTIFICATION OF THE FIRST RECORD ON THE DISTRIBUTION FILE
WITH TEC NOT EQUAL TO SPACES IS STORED IN D-BUFFER.
N IS THE NUMBER OF SUCH DISTRIBUTION RECORDS IN DIS-TABLE.


01017  READ-RATE.
01018      READ RATE-FILE, AT END  GO TO END-PROGRAM.
           IF RTEC =     SPACES MOVE RWORK-CODE TO CWORK-CODE (M),
                                MOVE RFS TO CFS (M),
                                MOVE RNARF TO CNARF (M),
                                MOVE RFY TO CFY (M),
                                MOVE RFIELDC TO CFIELDC (M),
               ADD 1 TO M,  GO TO READ-RATE.
           MOVE RATE-ID TO R-B-ID.
           MOVE RFY TO R-B-FY.
           MOVE RFIELDC TO R-B-FIELDC.
STEP 3.  THE RATE RECORDS WITH THE SPECIAL PROGRAM VALUES OF F, H, L, P,
R, V, OR Y ARE READ INTO COST-TABLE.  ALSO, THE IDENTIFICATION OF THE FIRST
RECORD ON THE RATE FILE WITH TEC NOT EQUAL TO SPACES IS STORED IN R-BUFFER.
M IS THE NUMBER OF SUCH RATE RECORDS IN COST-TABLE.


00809  READ-MASTER-FILE.
00810      READ MASTER-FILE, AT END GO TO END-PROGRAM.
           IF MTEC =     ZEROS,   GO TO READ-MASTER-FILE.
           IF MTEC =     SPACES,  GO TO READ-MASTER-FILE.
00811      MOVE MTEC       TO  MTEC-SK.
00812      MOVE MWORK-CODE TO  MWORK-CODE-SK.
00813      MOVE MNARF      TO  MNARF-SK.
           MOVE MWORK-CODE TO IDATA-WC.
           MOVE MCUST TO IDATA-C.
           MOVE MNARF TO IDATA-N.
           EXAMINE MASTER REPLACING ALL SPACES BY ZEROS.
           MOVE 1 TO KK, K.
STEP 4.  ONE MASTER RECORD IS READ OFF THE MASTER FILE.
THE DISTRIBUTION RECORD IDENTIFIER IS STORED IN MSORT-KEY.


       CHECK-NARF.
           IF MNARF =     N-C (KK),  GO TO MOVE-FIELDS,
              ELSE
           IF KK <     35,  ADD 1 TO KK,  GO TO CHECK-NARF.
```

```
CONVERTS THE NARF CODE FROM  A LETTER (A=G< TO A  NUMBER (1=7<.


00816  MOVE-FIELDS.
           MOVE MTQS (K) TO ITQS (K).
           MOVE MTME (K) TO ITME (K).
00818      MOVE MFIELDB (K)  TO  IFIELDB (K).
           IF MPRO  = ≠F≠ OR ≠H≠  OR MPRO  =  ≠L≠
               COMPUTE  IREQ (K) = (MQTYMEFT (K) * MTME (K)) +
              (MQTYNMFT (K) * (MTQS (K) - MTME (K))),
                   ELSE
               COMPUTE  IREQ (K) = (MQTYME (K) + MQTYNM (K)).
00819      IF K  <      5, ADD 1 TO K,
00820                   GO TO   MOVE-FIELDS.
CERTAIN FIELDS ARE MOVED DIRECTLY FROM THE MASTER RECORD TO CORRESPONDING
FIELDS IN  INPUT-RECORD.
THEY WILL LATER BE MOVED TO EACH DATA BASE RECORD.
ALSO, THE REQUIREMENT VARIABLE IS COMPUTED.


       COMPUTE-FUND-CODE.
           IF MCUST =      ≠0≠,  MOVE ≠N≠ TO IDATA-FS,
               GO TO END-F-S.
           IF MCUST =      ≠N≠,  MOVE ≠0≠ TO  IDATA-FS,
               GO TO END-F-S.
           IF MCUSTN IS NUMERIC AND MCUSTN NOT < 1
               GO TO COMPUTE-F-C.
           IF MCUST IS ALPHABETIC AND MCUST > ≠A≠ OR = ≠A≠ AND MCUST IS
               < ≠I≠ OR = ≠I≠
               GO TO COMPUTE-F-C,
                   ELSE
               GO TO SET-FS-E-U.
       COMPUTE-F-C.
           MOVE 1 TO I, J.
       CHECK-SUB-PROGRAM.
           IF MSUB =      F-S (I, 1),
               GO TO CHECK-PROGRAM.
           IF I <      42,  ADD 1 TO I,  GO TO CHECK-SUB-PROGRAM
               ELSE
               GO TO SET-FS-E-U.
       CHECK-PROGRAM.
           ADD 1 TO J.
           IF F-S (I, J) =      ≠ ≠,  GO TO FINAL-CHECK.
           IF MPRO =      F-S (I, J),  MOVE F-S (I, 4) TO IDATA-FS,
               GO TO END-F-S.
                   ELSE
               IF J <      3,  GO TO CHECK-PROGRAM.
       FINAL-CHECK.
           IF MSUB =      ≠L≠ AND MPRO =      ≠A≠,
               MOVE ≠E≠ TO IDATA-FS,  GO TO END-F-S.
       SET-FS-E-U.
           MOVE ≠U≠ TO IDATA-FS.
       END-F-S.
           MOVE IDATA-FS TO MCUST.
STEP 5.  COMPUTATION OF THE FUND-CODE USING COMBINATIONS OF THE CUSTOMER,
PROGRAM, SUBPROGRAM FROM THE MASTER RECORD.
THESE COMBINATIONS ARE REPRESENTED IN FUGURE 6.
```

```
            IF MPRO =      ≠F≠ OR ≠H≠ OR MPRO =      ≠L≠
                GO TO INT-BS-DIS.
            IF MID-NAME2 =      P-ID-NAME2,  GO TO SEARCH-D-TABLE.
```

STEP 6. CHECKS THE MASTER RECORD PROGRAM VALUE FOR F, H, OR L.
IF EQUAL TO ONE OF THE ABOVE VALUES THE DISTRIBUTION TABLE, DIS-TABLE,
IF NOT EQUAL TO ONE OF THE ABOVE VALUES, THE DISTRIBUTION TAPE
IS SEARCHED BY FIRST FETCHING ALL DISTRIBUTION RECORDS FROM THE TAPE FILE
WITH TEC, PROGRAM, AND SUBPROGRAM EQUAL TO THOSE VALUES ON THE CURRENT
MASTER RECORD AND STORING THEM IN D-TABLE.
IF THE VALUES OF TEC, PROGRAM, AND SUBPROGRAM FROM THE PREVIOUS MASTER
RECORD ARE EQUAL TO THOSE OF THE CURRENT MASTER RECORD THEN IT IS ASSUMED
THAT IF THE APPROPRIATE DISTRIBUTION RECORD FOR THE CURRENT MASTER RECORD
EXISTS IT IS ALREADY IN D-TABLE.
IF THE PREVIOUS MASTER RECORD VALUES ARE NOT EQUAL TO THOSE ON THE
CURRENT MASTER RECORD AND THE CURRENT MASTER RECORD PROGRAM VALUE
IS NOT F, H, OR L THEN D-TABLE MUST BE INITIALIZED TO ZERO AND REFILLED WITH
NEW DISTRIBUTION RECORDS CORRESPONDING TO THE CURRENT MASTER RECORD.

```
        LOAD-D-TABLE.
            MOVE ZERO TO NDT.
        EQUAL-ID.
            IF MID-NAME2 >      D-B-ID2,  GO TO READ-DIS.
            IF MID-NAME2 =      D-B-ID2,  ADD 1 TO NDT,
                MOVE D-B-ID2 TO D-T-ID2 (NDT),
                MOVE D-B-N TO D-T-N (NDT),
                MOVE D-B-DATA TO D-T-DATA (NDT),
                GO TO READ-DIS.
            GO TO LOAD-R-TABLE.
        READ-DIS.
            IF EOFD =  1,  GO TO LOAD-R-TABLE.
            READ CAP-DISTRI, AT END MOVE 1 TO EOFD,  GO TO LOAD-R-TABLE.
            MOVE DID-NAME2 TO D-B-ID2.
            MOVE DNARF TO D-B-N.
            MOVE DFIELDA TO D-B-DATA.
            GO TO EQUAL-ID.
```

STEP 6. THESE INSTRUCTIONS FILL D-TABLE WITH ALL DISTRIBUTION RECORDS WITH
TEC, PROGRAM, AND SUBPROGRAM EQUAL TO THOSE VALUES IN THE CURRENT MASTER
RECORD. THE RESULT OF THIS IS THAT IF THE WORKLOAD FOR A PARTICULAR
TEC, PROGRAM, SUBPROGRAM IS ASSIGNED TO MORE THAN ONE DRP THEN ALL
DISTRIBUTION RECORDS FOR THE DIFFERENT DRPS WILL BE IN D-TABLE AT
THE SAME TIME. THE COUNTER NDT REPRESENTS THE NUMBER OF DISTRIBUTION
RECORDS IN D-TABLE.

```
        LOAD-R-TABLE.
            MOVE ZERO TO NRT.
        EQUAL-RID.
            IF MID-NAME2 >      R-B-ID2,  GO TO READ-RATE2.
            IF MID-NAME2 =      R-B-ID2,  ADD 1 TO NRT,
                MOVE R-B-ID TO R-T-ID (NRT),
                MOVE R-B-FY TO R-T-FY (NRT),
                MOVE R-B-FIELDC TO R-T-FIELDC (NRT),
            GO TO READ-RATE2.
            GO TO SEARCH-D-TABLE.
        READ-RATE2.
            IF EOFR =      1,  GO TO SEARCH-D-TABLE.
            READ RATE-FILE, AT END MOVE 1 TO EOFR,  GO TO SEARCH-D-TABLE.
            MOVE RATE-ID TO R-B-ID.
```

```
            MOVE RFY TO R-B-FY.
            MOVE RFIELDC TO R-B-FIELDC.
            GO TO EQUAL-RID.
STEP 7.  THESE INSTRUCTIONS FILL R-TABLE WITH RATE RECORDS WITH RATE
VARIABLES OF TEC, PROGRAM, SUBPROGRAM EQUAL TO THOSE VALUES FROM THE
MASTER RECORD AND WITH RATE PROGRAM VARIABLE EQUAL TO A, N, OR T.
THE COUNTER NRT IS THE NUMBER OF RATE RECORDS IN R-TABLE.


        SEARCH-D-TABLE.
            MOVE 1 TO L.
            IF NDT =      ZERO,  GO TO NO-MATCH-D.
        CHECK-KEY-DIS.
            IF MSORT-KEY =      D-T-ID (L),
                MOVE D-T-DATA (L) TO DFIELD,  GO TO SET-I-D.
            IF L <      NDT,  ADD 1 TO L,  GO TO CHECK-KEY-DIS.
        NO-MATCH-D.
            MOVE 1 TO RSW,  GO TO FIND-RATE.
STEP 6.  SEARCH D-TABLE FOR THE DISTRIBUTION RECORD WITH THE SAME DRP
AS THE MASTER RECORD.
RSW IS SET TO 1 IF NO MATCH IS FOUND.


        SET-I-D.
00914       MOVE  1 TO  I.
        MOVE-SHOPS.
            MOVE DFIELD TO ISHOPS (I).
                IF I <     5, ADD 1 TO I,   GO TO MOVE-SHOPS,
01003                  ELSE
01004           GO TO FIND-RATE.
STEP 6.  WHEN A MATCH IS FOUND IN DIS-TABLE OR D-TABLE THE DISTRIBUTION
FACTORS FOR FIVE YEARS ARE MOVED FROM THE TABLE TO INPUT-RECORD.
LATER THEY WILL BE MOVED TO THE DATA BASE RECORDS.


        INT-BS-DIS.
            MOVE 1 TO SUB.
        BS-DIS.
            IF MS-KEY =      DT-ID (SUB),  MOVE DTFIELDA (SUB) TO DFIELD,
                GO TO SET-I-D.
            IF SUB <      N,  ADD 1 TO SUB,  GO TO BS-DIS.
                MOVE 1 TO RSW,   GO TO FIND-RATE.
SEARCH DIS-TABLE FOR THE APPROPRIATE DISTRIBUTION RECORD.
THE SEARCH VARIABLES ARE PROGRAM, SUBPROGRAM,AND NARF.


        PRINT-ERROR.
            IF RSW =      ZERO,   GO TO WRITE-INPUT-FILE.
            IF RSW =      1 OR 3,  GO TO PRINT-D,
                ELSE
            GO TO PRINT-R.
        PRINT-D.
            MOVE ≠NO DISTRIBUTION RECORD EXISTS FOR≠ TO P-MESSAGE.
            MOVE MID-NAME TO P-DATA.
            WRITE PRINTALL.
            IF RSW =      1,   GO TO WRITE-INPUT-FILE.
        PRINT-R.
            MOVE ≠NO RATE RECORD EXISTS FOR≠ TO P-MESSAGE.
            MOVE MID-NAME TO P-DATA.
```

```
            WRITE PRINTALL.
WRITE MESSAGES RELATING TO NO DISTRIBUTION OR NO RATE RECORDS EXISTING
FOR THE CURRENT MASTER RECORD ONTO  THE PRINTER.


01014  WRITE-INPUT-FILE.
           MOVE 1 TO J.
           IF P-TEC =      MTEC  GO TO MOVE-NEW-CODE.
               MOVE N-C (II) TO IDEN1,
               MOVE N-C (JJ) TO IDEN2,
               MOVE IDENT TO DICT-ID,
               MOVE IDATA-FS TO DICT-FS,
               MOVE MTEC TO DICT-TEC,
               EXAMINE MTMS REPLACING ALL ZEROS BY SPACES,
               MOVE MTMS TO DICT-TMS,
               WRITE DICTIONARY,
           IF JJ <      35,  ADD 1 TO JJ,
               ELSE  ADD 1 TO II,  MOVE 1 TO JJ.
       MOVE-NEW-CODE.
           MOVE IDENT TO IDATA-ID.
       SET-DATA-REC.
           MOVE IFIELD (J) TO IDATA.
           MOVE J TO IDATA-YR.
           WRITE DATE-BASE FROM IDATA-RECORD.
           IF J <      5,
                   ADD 1 TO J,  GO TO SET-DATA-REC.
           MOVE ZEROS TO INPUT-RECORD.
           MOVE MID-NAME2 TO P-ID-NAME2.
           MOVE ZERO TO RSW.
01016      GO TO  READ-MASTER-FILE.
STEP 9 AND 10.  MOVE THE FIVE DATA BASE RECORDS CONTAINED IN INPUT-RECORD
TO DATA-BASE-FILE.  THIS IS DONE AFTER THE DATA BASE CODE HAS REPLACED
THE TEC CODE.  THE PROGRAM THEN BRANCHES BACK TO READ ANOTHER MASTER
RECORD, WHICH BECOMES THE CURRENT MASTER RECORD.


       FIND-RATE.
           IF MPRO =      #A# OR #N# OR MPRO =      #T#
               GO TO SEARCH-R-TABLE,
                   ELSE
                           MOVE MWORK-CODE TO MWC,
                           MOVE IDATA-FS TO MFUSO,
                           MOVE MNARF TO MNAR,
               GO TO INT-BS-RATE.
       SEARCH-R-TABLE.
           MOVE 1 TO L.
           IF NRT =      ZERO,  GO TO NO-MATCH-R.
       CHECK-KEY-RATE.
           IF MID-NAME =      R-T-ID (L),  GO TO SET-I-R.
           IF L <      NRT,  ADD 1 TO L,  GO TO CHECK-KEY-RATE.
       NO-MATCH-R.
           IF RSW =      1,  MOVE 3 TO RSW
               ELSE
           MOVE 2 TO RSW.
           GO TO PRINT-ERROR.
       SET-I-R.
           MOVE 1 TO I.
       MOVE-R-FIELDS.
           MOVE R-T-RATES (L, I) TO I-RATES (I).
```

```
            MOVE RUM (L, I) TO I-UMRATE (I).
            IF KK <      8,  MOVE GA-R-T (KK, I) TO GARATE, IGA-RATE (I),
                ELSE
            MOVE RGAO (L, I) TO IGA-RATE (I), GARATE.
            IF MNORM (I) =      ZERO,  MOVE ZERO TO ITOT-COST (I),
                GO TO ALTER-I-J.
            COMPUTE ITOT-COST (I) ROUNDED =
            ((RDLC (L, I) + RPO (L, I) + GARATE)       * MNORM (I))
            + RUM (L, I).
        ALTER-I-J.
            IF I <      5,  ADD 1 TO I,      GO TO MOVE-R-FIELDS.
            GO TO PRINT-ERROR.
STEP 8.  IF THE MASTER PROGRAM IS EQUAL TO A, N, OR T THEN SEARCH R-TABLE
FOR THE APPROPRIATE RATE RECORD.  THE SEARCH VARIABLES ARE TEC, PROGRAM,
SUBPROGRAM, FUND-CODE, AND ORP.
IF A MATCH IS FOUND THE RATES FROM THE RATE RECORD ARE USED TO COMPUTE
THE TOTAL COST VARIABLE FOR EACH OF THE FIVE DATA BASE RECORDS,
WITH THE EXCEPTION THAT IF THE RATE RECORD REPRESENTS A NARF THE G A RATE
FOR THAT NARF AND YEAR ARE TAKEN FROM THE GA-RATE-TABLE CONTAINING
THE CURRENT G A RATES.


        INT-BS-RATE.
            MOVE 1 TO SUB.
        BS-RATE.
            IF MPSCNF  =      COST-ID (SUB),  GO TO SET-I-RATE.
            IF SUB <      M,  ADD 1 TO SUB,  GO TO BS-RATE.
                GO TO NO-MATCH-R.
        SET-I-RATE.
            MOVE 1 TO I.
01210   COM-COST-FJ.
            MOVE COST-DATA (SUB, I) TO I-RATES (I).
            MOVE CUNIMAT (SUB, I) TO I-UMRATE (I).
            IF KK <      8,  MOVE GA-R-T (KK, I) TO GARATE, IGA-RATE (I),
                ELSE
            MOVE CGAOVHD (SUB, I) TO GARATE, IGA-RATE (I).
            COMPUTE ITOT-COST (I) ROUNDED =
            (CDIRLABCT (SUB, I) + CDIRMATFJ (SUB, I) + CPROOVHD (SUB, I)
            + GARATE)            * MNORM (I).
            IF I <      5,  ADD 1 TO      I,
                GO TO COM-COST-FJ,
                ELSE
            GO TO PRINT-ERROR.
SEARCH COST-TABLE FOR THE RATE RECORD IF THE MASTER PROGRAM VARIABLE
IS EQUAL TO F, H, L, P, R, V, OR Y.  AS IN 12 ABOVE IF A MATCH
IS FOUND THE RATES ARE USED TO COMPUTE THE TOTAL-COST VARIABLE FOR
THE FIVE DATA BASE RECORDS.
SIMILARLY, IF THE RATE RECORD REPRESENTS A NARF THE G A RATE IS TAKEN
FROM GA-RATE-TABLE.


01214   END-PROGRAM.
01218       CLOSE MASTER-FILE, CAP-DISTRI, RATE-FILE,
            DATA-BASE-FILE,  PRINTFILE,  DICTIONARY-FILE.
            CLOSE GA-RATE-FILE.
CLOSE THE FILES.


            STOP RUN.
```

```
//GO.SYS001 DD DSN=PCNA.UMASTFI3,DISP=(OLD,KEEP),UNIT=(TAPE,,DEFER),
//     LABEL=(,SL),VOL=SER=990006
//GO.SYS002 DD DSN=PCNA.UCAPDIS3,DISP=(OLD,KEEP),UNIT=(TAPE,,DEFER),
//     LABEL=(,SL),VOL=SER=992410
//GO.SYS003 DD DSN=PCNA.UCOSTRA3,DISP=(OLD,KEEP),UNIT=(TAPE,,DEFER),
//     LABEL=(,SL),VOL=SER=990222
//GO.SYS004 DD DSN=PCNA.UDATABAS,DISP=(OLD,KEEP),UNIT=(TAPE,,DEFER),
//     LABEL=(,SL),VOL=SER=992299
//GO.SYSPRT DD  SYSOUT=A
//GO.SYSPCH DD  SYSOUT=B
//GO.SYSRED DD  *
THE CURRENT G A RATE CARDS GO FOLLOW.


448804699446054698746178
399304127339703398363858 9
444914366644052464134856 1
489564966646762438244294 6
523365566257317601455919 7
384653702134488345153343 1
483655267855066568265628 7
/*
```

ANNEX A-3

MATRIX GENERATOR

Move
NTOI
LJABF
Search
CONVRT
Main

F150CT70 2/23/73

```
LOC  OBJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT
000000                              1          START
                                    2          ENTRY MOVE,NTOI,LJABF
000000 47FF 000A             000CA  3 MOVE     BC    15,10(15)
000004 05                           4          DC    X'5'
000005 D4D6E5C540                   5          DC    CL5'MOVE'
00000A 90E7 D00C             000CC  6          STM   14,7,12(13)
00000E 0520                         7          BALR  2,0
000010                              8          USING *,2
000010 9837 1000             00000  9          LM    3,7,0(1)
000014 5844 0000             00000 10          L     4,0(4)
000018 5866 0000             00000 11          L     6,0(6)
00001C 5877 0000             00000 12          L     7,0(7)
000020 5840 2098             000A8 13          S     4,=F'1'
000024 5860 2098             000A8 14          S     6,=F'1'
000028 5870 2098             000A8 15          S     7,=F'1'
00002C 4240 202D             0002D 16          STC   4,45(0,2)
000030 4260 202B             0002B 17          STC   6,43(0,2)
000034 4270 2029             00029 18          STC   7,41(0,2)
000038 D200 5000 3000 00000  00000 19          MVC   0(0,5),0(3)
00003E 98E7 D00C             000CC 20          LM    14,7,12(13)
000042 92FF D00C       0000C      21          MVI   12(13),X'FF'
000046 07FE                        22          BCR   15,14
000048 47FF 000A             000CA 23 NTOI     BC    15,10(15)
00004C 05                          24          DC    X'5'
00004D D5E3D6C940                  25          DC    CL5'NTOI '
000052 90E3 D00C             000CC 26          STM   14,3,12(13)
000056 0520                        27          BALR  2,0
000058                             28          USING *,2
000058 5831 0000             00000 29          L     3,0(1)
00005C 5803 0000             00000 30          L     0,0(3)
000060 8400 0018             00018 31          SRL   0,24
000064 5400 2054             000AC 32          N     0,=X'0000000F'
000068 9823 D01C             0001C 33          LM    2,3,28(13)
00006C 92FF D00C       0000C      34          MVI   12(13),X'FF'
000070 07FF                        35          BCR   15,14
000072 47FF 000A             0000A 36 LJABF    BC    15,10(15)
000076 05                          37          DC    X'5'
000077 D3D1C1C2C6                  38          DC    CL5'LJABF'
00007C 90E4 D00C             000CC 39          STM   14,4,12(13)
000080 0520                        40          BALR  2,0
000082                             41          USING *,2
000082 9834 1000             00000 42          LM    3,4,0(1)
000086 5844 0000             00000 43          L     4,0(4)
00008A 5840 2026             000A8 44          S     4,=F'1'
00008E 4304 3000             00000 45          IC    0,0(4,3)
000092 8900 0018             00018 46          SLL   0,24
000096 5600 202E             000B0 47          O     0,=X'00404040'
00009A 9814 D018             00018 48          LM    1,4,24(13)
00009E 92FF D00C       0000C      49          MVI   12(13),X'FF'
0000A2 07FE                       50          BCR   15,14
                                  51          END
0000A8 00000001                   52                =F'1'
0000AC 0000000F                   53                =X'0000000F'
0000B0 00404040                   54                =X'00404040'
```

```
0001            SUBROUTINE SEARCH (IVAL,NN)
0002            COMMON/DICT/IDR,IDICT(2,400)
0003            NDR=IDR
0004            N=NN
0005            IF (N.EQ.2) GO TO 60
0006            ITEMP=0
0007            CALL MOVE (IVAL,1,ITEMP,3,2)
0008         60 LPOINT=0
0009            INDX=NDR+1
0010         61 INDX=(INDX+1)/2
0011            KK=LPOINT+INDX
0012            IF (KK.GT.NDR) GO TO 63
0013            IF (N.EQ.2) GO TO 68
0014            ITEMQ=0
0015            CALL MOVE (IDICT(1,KK),1,ITEMQ,3,2)
0016            IF (ITEMQ - ITEMP) 62,66,63
0017         68 IF (IDICT(2,KK) - IVAL) 62,66,63
0018         62 LPOINT=KK
0019         63 IF (INDX - 1) 64,64,61
0020         64 IF (N.EQ.2) GO TO 71
0021            PRINT 67,IVAL
0022         67 FORMAT (3X,'SOMETHING WRONG IN SEARCH.  NO TEC RELOCATED FOR CODE
                 '',A2)
0023         71 NN=0
0024            GO TO 70
0025         66 NN=KK
0026         70 RETURN
0027            END
```

```
FORTRAN IV G LEVEL  20                CONVRT          DATE = 73054        19/29/38          PAGE 0001

0001                SUBROUTINE CONVRT (ICD,VALU,ICHK)
0002                DIMENSION ICD(64), IOP(21),ICHK(21)
0003                REAL*8 VALU(21)
0004                DATA IBLANK/1H /
0005                DATA MINUS/1H-/
0006                DO 5 J=1,21
0007                ICHK(J)=0
0008                IOP(J)=0
0009              5 VALU(J)=0.
0010                DO 100 J=1,6
0011                IF (ICD(J+50).NE.IBLANK) ICHK(20)=1
0012                VALU(20)=VALU(20)*10.+NTOI(ICD(J+50))
0013                IF (J.GT.5) GO TO 100
0014                IF (ICD(J+11).NE.IBLANK) ICHK(13)=1
0015                VALU(13)=VALU(13)*10.+NTOI(ICD(J+11))
0016                IF (ICD(J+41).NE.IBLANK) ICHK(17)=1
0017                VALU(17)=VALU(17)*10.+NTOI(ICD(J+41))
0018                IF (J.GT.4) GO TO 100
0019                IF (ICD(J).NE.IBLANK) ICHK(10)=1
0020                VALU(10)=VALU(10)*10.+NTOI(ICD(J))
0021                IF (ICD(J+4).NE.IBLANK) ICHK(11)=1
0022                VALU(11)=VALU(11)*10.+NTOI(ICD(J+4))
0023                IF (ICD(J+37).NE.IBLANK) ICHK(16)=1
0024                VALU(16)=VALU(16)*10.+NTOI(ICD(J+37))
0025                IF (ICD(J+46).NE.IBLANK) ICHK(18)=1
0026                VALU(18)=VALU(18)*10.+NTOI(ICD(J+46))
0027                IF (J.GT.3) GO TO 100
0028                IF (ICD(J+8).NE.IBLANK) ICHK(12)=1
0029                VALU(12)=VALU(12)*10.+NTOI(ICD(J+8))
0030                IF (ICD(J+16).NE.IBLANK) ICHK(14)=1
0031                VALU(14)=VALU(14)*10.+NTOI(ICD(J+16))
0032                IF (J.GT.2) GO TO 100
0033                M=17+J
0034                DO 100 L=1,9
0035                M=M+2
0036                IF (ICD(M).NE.IBLANK) ICHK(L)=1
0037                VALU(L)=VALU(L)*10.+NTOI(ICD(M))
0038            100 CONTINUE
0039                RETURN
0040                ENTRY BNDRHS(ICD,VALU,ICHK)
0041                DO 120 J=1,5
0042                ICHK(J)=0
0043                IOP(J)=0
0044            120 VALU(J)=0.
0045                DO 130 J=1,8
0046                M=J-8
0047                DO 130 L=1,5
0048                M=M+8
0049                IF (ICD(M).NE.IBLANK) ICHK(L)=1
0050                IF (ICD(M).EQ.MINUS) IOP(L)=1
0051                VALU(L)=VALU(L)*10.+NTOI(ICD(M))
0052            130 CONTINUE
0053                DO 140 J=1,5
0054                IF (IOP(J).EQ.1) VALU(J)=-VALU(J)
0055            140 CONTINUE
FORTRAN IV G LEVEL  20                BNDRHS          DATE = 73054        19/29/38          PAGE 0002

0056                RETURN
0057                END
```

```
0001              COMMON/DICT/NDR,IDICT(2,400)
0002              INTEGER ICLRW(2),IBNDX(2,5),IRIGHT(2,26,5),IDN(2,2000),
                 1NRIGHT(5),IBNDA(2,100),IBNDC(100),IDNA(2,5,12),IRECA(2,5,12),
                 2ITMP(2,5),LASTID(2),KASTID(2),IDSAVE(2,1000),IBOUND(2,2000),
                 3NMATCH(13),JDICT(4,400)
0003              INTEGER NBASE(13),NBAD(13)
0004              INTEGER MIXUP(3,100),ICD(64),ICHK(21)
0005              INTEGER JDENT(40)
                 */3HION,4HBNDS,3HRHS,3HEND,1HL,1HU,
                 *ZFFFFFFFF,1HF,1HS,1HT,1HL,1HG,1HM,1HN,1HU,1HV,1HW,
                 *1HH,1HL,1H-,1H ,1HE,1HL,1HL,1HM,1HN,1HP,1HH,1HL,1HG,1HK,
                 *1H-,1H ,1H-,1H ,2HLO,2HUP,1HB,4HGARC,
                 *1HD/
0006              INTEGER IOTF (5)/1H1,1H2,1H3,1H4,1H5/
0007              INTEGER IPROGS(10)/1HA,1HN,1HF,1HH,1HL,1HP,1HR,1HT,1HV,1HY/
0008              INTEGER IFUNDS(8)/1HA,1HC,1HD,1HE,1HI,1HN,1HO,1HU/
0009              INTEGER NAREX(12)/1HA,1HB,1HC,1HD,1HE,1HF,1HG,1HH,1HI,1HJ,1HK,1HL/
0010              INTEGER NAREY(12,2)/1HM,1HN,1HO,1HP,1HQ,1HR,1HS,1HT,1HU,1HV,1HW,
                 *1HX,1HY,1HZ,1HC,1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9/
0011              INTEGER IOBREN(22)
                 *                  /3HS 1,3HS 2,3HS 3,3HS 4,3HS 5,3HS 6,
                 *3HS 7,3HS 8,3HS 9,3HTQS,3HTME,3HM/E,4HNORM,3HM/N,
                 *3HREQ,3HDLR,3HDMR,3HPOR,3HGAR,3HUMC,4HCOST,
                 *3HDEL/
0012              INTEGER ITQS(5)
0013              REAL*8 TEM(15),PVAR(5),PCAP2(5),PCAP3(5),EF2(5),EF3(5),VALU(21),
                 *FRIGHT(5,25,5),BNDB(5,100),COSTB(5,9, 7,8),CAPB(5,9, 7),
                 2TOTCAP(5,9, 7),CML(9, 7,5),GAR( 7,5),
                 3RECB (21,5,12), TMP(21,5),SUMN(8,10,12,5),REQID (1000),
                 4BOUNDS (2,2000),        BOUNDA(2),BNDY(5,5),
                 5HCOST(4),SUMR(3),CMLA(9, 7,5)
0014              REAL*8 XLML,XUML,SUM,PROD,PERCAP,BVAL
0015              REAL*8 TEMP
0016              REAL*8 HPY(7,5)
0017              REAL*8 PHPY(5)
0018              DATA IBLANK/1H /
0019              REAL*8          FIXUP(21,100)
0020              EQUIVALENCE (JDICT(1,1),BOUNDS(1,1))
0021              ISTOP=0
0022              NBS=0
0023              ICODE=0
0024              LUI=1
0025              LUO=2
0026              LUQ=3
0027              LUU=4
0028              LUV=5
0029              LUD=5
0030              NRS=0
0031              JIDN=0
0032              JBNDS=0
0033              JRHS=0
0034              NFACIL=12
0035              NNARFS=7
0036              NSHOPS=9
0037              NPROGS=10
```

```
0038              NFUNDS=8
0039              NYEARS=5
0040              JID=0
0041              JBDS=0
0042              JDBC=0
0043              KG=19
0044              KR=15
0045              KC=21
0046              IWPR=21
0047              DO 46 J=1,5
0048              EF3(J)=0.
0049              FF2(J)=0.
0050              PHPY(J)=0.
0051              NRIGHT(J)=0
0052              DO 46 L=1,NNARFS
0053           46 HPY(L,J)=0.
0054              DO 47 M=1,NYEARS
0055              DO 47 L=1,NFACIL
0056              DO 47 K=1,NPROGS
0057              DO 47 J=1,NFUNDS
0058           47 SUMN(J,K,L,M)=0.
0059              DO 48 L=1,NNARFS
0060              DO 48 M=1,NYEARS
0061           48 GAR(L,M)=0.
             C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
             C---- READ FROM 2 TO 6 INFORMATION CARDS INTO CORE
0062              READ(5,91)IBOP,INOS,IAY,IYA,IYB,INC,IMCBB,XUML,XLML
0063           91 FORMAT (2I1,I2,4I1,2F5.0)
0064              IF(IYB.EQ.0)IYB=IYA
0065              IF(INC.GT.0)GO TO 93
0066              PRINT 92
0067              ISTOP=1
0068           92 FORMAT (3X,'FIFTH VARIABLE ON FIRST CARD NOT GIVEN. SET TO 1.')
0069              INC=1
0070           93 ISET=0
0071              DO 99 J=1,INC
0072              READ(5,95)IYC,TEM
0073           95 FORMAT (I1,4X,15F5.0)
0074              IF (IYC.GT.0.AND.IYC.LE.NYEARS) GO TO 96
0075              PRINT 94
0076           94 FORMAT (3X,'YEAR DATA CARD HAS INVALID YEAR')
0077              ISTOP=1
0078              GO TO 99
0079           96 IF(IYC.NE.IYA) GO TO 89
0080              ISET=1
0081              DO 86 L=1,13
0082              IF (IBOP.EQ.1.OR.L.NE.1) GO TO 81
0083              IF ( TEM(1).NE.0.) GO TO 86
0084              ISTOP=1
0085              PRINT 82
0086           82 FORMAT (3X,'PARAMETRIC PERCENT VARIATION IS NOT GIVEN FOR AT LEAST
                 * THE FIRST YEAR REQUESTED')
0087              GO TO 86
0088           81 IF (L.EQ.4.OR.L.EQ.5.OR.L.EQ.1)          GO TO 86
0089              IF (TEM(L).NE.0.) GO TO 86
```

```
0090            PRINT 83
0091         83 FORMAT (3X,'EITHER THE PERCENT CAPACITY OR THE NUMBER OF HOURS FOR
                * A NARF IS NOT GIVEN IN THE FIRST YEAR REQUESTED')
0092         86 CONTINUE
0093         89 PVAR(IYC)=TEM(1)/100.
0094            PCAP2(IYC)=TEM(2)/100.
0095            PCAP3(IYC)=TEM(3)/100.
0096            EF2(IYC)=TEM(4)/100.
0097            EF3(IYC)=TEM(5)/100.
0098            IF(EF2(IYC).EQ.0.)EF2(IYC)=1.
0099            IF(EF3(IYC).EQ.0.)EF3(IYC)=1.
0100            PHPY(IYC)=TEM(13)
0101            DO 87 L=1,NNARFS
0102         87 HPY(L,IYC)=TEM(L+5)
0103         99 CONTINUE
0104            IF(ISET.EQ.1)GO TO 97
0105            PRINT 90
0106         90 FORMAT (3X,'DATA FOR FIRST YEAR REQUESTED IS NOT GIVEN')
0107            ISTOP=1
0108         97 IF(INC.EQ.IYB-IYA+1)GO TO 108
0109            IF(INC.LT.IYB-IYA+1)GO TO 103
0110            PRINT 98
0111         98 FORMAT(3X,'TOO MANY CARDS FOLLOWING FIRST CARD')
0112            ISTOP=1
0113        103 IYD=IYB-1
0114            IF(IYD.LT.IYA)GO TO 108
0115            DO 106 J=IYA,IYD
0116            IF(EF2(J+1).NE.0.)GO TO 106
0117            EF3(J+1)=EF3(J)
0118            EF2(J+1)=EF2(J)
0119            PCAP3(J+1)=PCAP3(J)
0120            PCAP2(J+1)=PCAP2(J)
0121            PVAR(J+1)=PVAR(J)
0122            PHPY(J+1)=PHPY(J)
0123            DO 106 L=1,NNARFS
0124            HPY(L,J+1)=HPY(L,J)
0125        106 CONTINUE
0126        108 XUML=XUML/100.
0127            XLML=XLML/100.
          C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
          C---- ALL REMAINING CARDS IN THE FILE ARE READ HERE
0128         50 READ (5,52) IDENT,ICLRW,IYC,IUPLO,ICO
0129         52 FORMAT (A4,1X,2A4,I1,A1,1X,64A1)
          C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
          C---- CHECK FOR IDN IN COLUMNS 1 THRU 3
0130            IF(IDENT.NE.JDENT(1))GO TO 53
0131            ICODE=1
0132            IF (ICLRW(1).NE.IBLANK) GO TO 110
0133            GO TO 50
          C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
          C---- CHECK FOR BNDS IN COLUMNS 1 THRU 4. ARRAYS IBNDX AND BNDY HOLD
          C---- THE BOUND NAME AND THE STANDARD PERCENT VARIATION OVER FIVE YEARS.
0134         53 IF(IDENT.NE.JDENT(2))GO TO 54
0135            ICODE=5
0136         55 IF (ICLRW(1).EQ.IBLANK) GO TO 50
```

```
0137              ICODE=2
0138              NBS=NBS+1
0139              IF(NBS.LE.5)GO TO 61
0140              PRINT 62,ICLRW
0141           62 FORMAT(3X,'TOO MANY BOUND NAMES SPECIFIED ',
                  '2A4,' NOT INCLUDED')
0142              GO TO 50
0143           61 IBNDX(1,NBS)=      ICLRW(1)
0144              IBNDX(2,NBS)=ICLRW(2)
0145              CALL BNDRHS (ICD,VALU,ICHK)
0146              DO 63 J=1,NYEARS
0147           63 BNDY(J,NBS)=VALU(J)/100.
0148              GO TO 50
               C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
               C---- CHECK FOR RHS IN COLUMNS 1 THRU 3. ARRAY IRIGHT(1-2) HOLDS THE
               C---- NAME GIVEN TO THE CHANGES WHEN USING PARAMETRIC EQUATIONS
0149           54 IF(IDENT.NE.JDENT(3))GO TO 56
0150              ICODE=6
0151           57 IF (ICLRW(1).EQ.IBLANK) GO TO 50
0152              ISETR=0
0153              ICODE=3
0154              JRHS=1
0155              NRS=NRS+1
0156              IF(NRS.LE.5)GO TO 64
0157              PRINT 65,ICLRW
0158           65 FORMAT(3X,'TOO MANY RIGHT HAND SIDES SPECIFIED ',
                  '2A4,' NOT INCLUDED')
0159              GO TO 50
0160           64 IRIGHT(1,1,NRS)=      ICLRW(1)
0161              IRIGHT(2,1,NRS)=ICLRW(2)
0162              GO TO 50
0163           56 IF (IDENT.NE.JDENT(39)) GO TO 58
0164              ICODE=4
0165           69 IF (ICLRW(2).EQ.IBLANK) GO TO 50
0166              CALL BNDRHS (ICD,VALU,ICHK)
0167              ITEMP=LJABF(ICLRW(1),8)
0168              DO 59 J=1,NNARFS
0169              IF (ITEMP.EQ.NARFX(J)) GO TO 66
0170           59 CONTINUE
0171           66 DO 67 K=1,NYEARS
0172              IF (ICHK(K).EQ.0) GO TO 67
0173              GAR(J,K)=VALU(K)/10000.
0174              PRINT 70,NARFX(J),K,GAR(J,K)
0175           70 FORMAT (3X,'G+A RATE FOR NARF ',A1,' YEAR ',I1,' SET TO ',F9.5)
0176           67 CONTINUE
0177              GO TO 50
               C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
               C---- CHECK FOR END IN COLUMNS 1 THRU 3. IF NOT, GO TO LOCATION
               C---- ESTABLISHED PREVIOUSLY BY SETTING ICODE.
0178           58 IF (IDENT.EQ.JDENT(4)) GO TO 150
0179              IF (IDENT.EQ.IBLANK) GO TO 72
0180              PRINT 74,IDENT
0181           74 FORMAT (3X,'INVALID IDENT ',A4,' IN COLUMNS 1 THRU 4')
0182              ISTOP=1
0183              GO TO 50
```

```
0184          72 IGOTO=ICODE+1
0185             GO TO (101,110,120,130,69,55,57,101),IGOTO
0186         101 PRINT 102
0187         102 FORMAT(3X,'NO IDENT HAS BEEN ESTABLISHED IN COLUMNS 1 THRU 4')
0188             GO TO 50
               C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
               C---- ARRAY IDN HOLDS THE REQUESTS
0189         110 IF (IUPLO.EQ.JDENT(38)) GO TO 111
0190             IF (IUPLO.NE.JDENT(40)) GO TO 114
0191             IYC=IYC+NYEARS+1
0192             GO TO 111
0193         114 JIDN=JIDN+1
0194             IDN(1,JIDN)=ICLRW(1)
0195             IDN(2,JIDN)=ICLRW(2)
0196             IF (IUPLO.EQ.JDENT(21)) GO TO 50
0197         111 JDBC=JDBC+1
0198             CALL CONVRT (ICD,VALU,ICHK)
0199             MIXUP(1,JDBC)=ICLRW(1)
0200             MIXUP(2,JDBC)=ICLRW(2)
0201             MIXUP(3,JDBC)=IYC
0202             DO 112 J=1,21
0203             IF (ICHK(J).EQ.0) GO TO 113
0204             FIXUP(J,JDBC)=VALU(J)
0205             GO TO 112
0206         113 FIXUP(J,JDBC)=-1.
0207         112 CONTINUE
0208             GO TO 50
               C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
               C---- ARRAYS IBNDA AND BNDB HOLD THE BOUND
               C---- EXCEPTIONS GIVING THE IDN NAME AND FIVE YEARS OF
               C---- DIFFERENT PERCENT VARIATIONS
0209         120 IF(NBS.GT.5)GO TO 50
0210             JBNDS=JBNDS+1
0211             IBNDA(1,JBNDS)=      ICLRW(1)
0212             IBNDA(2,JBNDS)=ICLRW(2)
0213             CALL BNDRHS (ICD,VALU,ICHK)
0214             DO 68 J=1,5
0215             IF (ICHK(J).EQ.0) GO TO 71
0216             BNDB(J,JBNDS)=VALU(J)/100.
0217             GO TO 68
0218          71 BNDB(J,JBNDS)=BNDY(J,NBS)
0219          68 CONTINUE
0220             I=3
0221             IF(IUPLO.EQ.JDENT(5))I=1
0222             IF(IUPLO.EQ.JDENT(6))I=2
0223             IBNDC(JBNDS)=NBS*10+I
0224             GO TO 50
               C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
               C---- ARRAYS IRIGHT (3-4) AND FRIGHT HOLD THE ID NAME WHICH IS
               C---- TO BE ALTERED AND THE FIVE YEARS OF CHANGES.
0225         130 IF(NRS.GT.5)GO TO 50
0226             JRHS=JRHS+1
0227             IF(JRHS.LE.26)GO TO 77
0228             IF(ISETR.EQ.1)GO TO 50
0229             ISETR=1
```

```
0230            PRINT 73,(IRIGHT(J,1,NRS),J=1,2)
0231          73 FORMAT(3X,'MORE THAN 25 RIGHT HAND SIDES FOR ',2A4,
                 '.  EXTRAS IGNORED')
0232            GO TO 50
0233          77 NRIGHT(NRS)=NRIGHT(NRS)+1
0234            DO 78 J=1,2
0235          78 IRIGHT(J,JRHS,NRS)=ICLRW(J)
0236            CALL BNDRHS (ICD,VALU,ICHK)
0237            DO 79 J=1,NYEARS
0238          79 FRIGHT(J,JRHS-1,NRS)=VALU(J)
0239            GO TO 50
              C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
              C---- READ INTO ARRAY COSTB ALL INCREMENTAL COST DATA, INTO
              C---- CAPB THE CAPACITIES AND INTO CML THE CURRENT MANNING
              C---- LEVEL
0240        150 CONTINUE
0241        158 DO 156 K=1,NNARFS
0242            DO 156 J=1,NSHOPS
0243            READ(LUV,154)CML(J,K,1)
0244            CML(J,K,1)=CML(J,K,1)*HPY(K,IYA)
0245            CMLA(J,K,1)=CML(J,K,1)
0246            DO 156 L=2,5
0247            CML (J,K,L)=CML (J,K,1)
0248        156 CMLA(J,K,L)=CMLA(J,K,1)
0249            DO 155 K=1,NNARFS
0250            DO 155 J=1,NSHOPS
0251            READ (LUV,154)(CAPB(I,J,K),I=1,NYEARS)
0252            DO 157 I=1,5
0253        157 TOTCAP(I,J,K)=CAPB(I,J,K)
0254            DO 155 I=2,5
0255            IF(CAPB(I,J,K).NE.0.) GO TO 155
0256            CAPB(I,J,K)=CAPB(I-1,J,K)
0257            TOTCAP(I,J,K)=TOTCAP(I-1,J,K)
0258        155 CONTINUE
0259            DO 152 L=7,8
0260            DO 152 K=1,NNARFS
0261            DO 152 J=1,NSHOPS
0262            DO 152 I=1,NYEARS
0263        152 COSTB(I,J,K,L)=0.
0264            M=8
0265            IF (IMCBB.NE.2) M=6
0266            DO 153 L=1,M
0267            DO 153 K=1,NNARFS
0268            DO 153 J=1,NSHOPS
0269            READ(LUV,154)          (COSTB(I,J,K,L),I=1,NYEARS)
0270        154 FORMAT(16X,5F8.0)
0271            DO 153 I=2,5
0272            IF(COSTB(I,J,K,L).EQ.0.)COSTB(I,J,K,L)=COSTB(I-1,J,K,L)
0273        153 CONTINUE
0274        133 DO 151 L=5,6
0275            DO 151 K=1,NNARFS
0276            DO 151 J=1,NSHOPS
0277            DO 151 I=1,NYEARS
0278            IF (COSTB(I,J,K,L+2).LT.COSTB(I,J,K,L)) GO TO 151
0279            PRINT 132
```

```
0280          132 FORMAT(3X,'COSTS FOR HIRE/LAYOFF PAIRED VARIABLES MUST BE GREATER
                 *BEYOND THE BOUNDS')
0281                STOP
0282          151 CONTINUE
0283                DO 1506 K=1,NNARFS
0284                DO 1506 J=1,NSHOPS
0285                DO 1506 I=1,NYEARS
0286                COSTB(I,J,K,2)=(1.-EF2(I))*COSTB(I,J,K,1)+COSTB(I,J,K,2)
0287           1506 COSTB(I,J,K,3)=(1.-EF3(I))*COSTB(I,J,K,1)+COSTB(I,J,K,3)
              C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
              C---- READ THE DICTIONARY INTO CORE AND WRITE THE DICTIONARY ON
              C---- THE FILE TO BE USED IN THE REPORT GENERATOR. T/M/S IS
              C---- USED ONLY IN THE REPORT GENERATOR.
0288                NDR=0
0289          134 NDR=NDR+1
0290                READ (LUD,135,END=136) (IDICT(I,NDR),I=1,2),(JDICT(I,NDR),I=1,4)
0291                WRITE (LUQ,135)        (IDICT(I,NDR),I=1,2),(JDICT(I,NDR),I=1,4)
0292          135 FORMAT (6A4)
0293                GO TO 134
0294          136 IDICT(1,NDR)=IBLANK
0295                IDICT(2,NDR)=IBLANK
0296                NDR=NDR-1
0297                PRINT 137,(IDICT(I,1),I=1,2),(JDICT(I,1),I=1,4)
0298          137 FORMAT(3X,'THE FIRST DICTIONARY RECORD IS ',A3,1X,A4,1X,3A4,A3)
              C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
              C---- SORT THE IDN ARRAY AND ELIMINATE ANY DUPLICATE REQUESTS
0299          170 KIDN=JIDN+1
0300          171 KIDN=KIDN-1
0301                IF(KIDN.LT.2)GO TO 160
0302                J=1
0303          174 J=J+1
0304          177 IF(J.GT.KIDN)GO TO 171
0305                IF(IDN(1,J)-IDN(1,J-1))173,172,174
0306          172 IF(IDN(2,J)-IDN(2,J-1))173,175,174
0307          175 PRINT 176,(IDN(K,J),K=1,2)
0308          176 FORMAT (3X,'DUPLICATE IDN REQUEST ',2A4)
0309                DO 178 K=J,JIDN
0310                DO 178 I=1,2
0311          178 IDN(I,K-1)=IDN(I,K)
0312                JIDN=JIDN-1
0313                KIDN=KIDN-1
0314                GO TO 177
0315          173 DO 179 K=1,2
0316                ITEMP   =IDN(K,J)
0317                IDN(K,J)=IDN(K,J-1)
0318          179 IDN(K,J-1)=ITEMP
0319                GO TO 174
              C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
              C---- CHANGE ALL TEC CALLED FOR IN THE IDN ARRAY TO THE
              C---- ASSOCIATED CODE IN THE DICTIONARY. IF THE TEC CANNOT
              C---- BE FOUND, PRINT AN ERROR MESSAGE AND ELIMINATE THAT
              C---- RECORD. THE FINAL FILE IN ARRAY IDN SHOULD REPRESENT
              C---- A SUBSET OF THE RECORDS IN THE DATA BASE. ALL
              C---- DUPLICATE AND INVALID REQUESTS HAVE BEEN
              C---- ELIMINATED
```

```
0320          160 K=1
0321              J=0
0322          161 J=J+1
0323          166 IF(J.GT.JIDN)GO TO 140
0324          164 CONTINUE
0325              IF(K.GT.NDR)GO TO 167
0326              IF(IDN(1,J)-IDICT(2,K))162,163,165
0327          163 IDN(1,J)=IDICT(1,K)
0328              GO TO 161
0329          165 K=K+1
0330              GO TO 164
0331          162 PRINT 168,IDN(1,J)
0332              M=J+1
0333              DO 169 L=M,JIDN
0334              DO 169 I=1,2
0335          169 IDN(I,L-1)=IDN(I,L)
0336              JIDN=JIDN-1
0337              GO TO 166
0338          167 DO 159 L=J,JIDN
0339          159 PRINT 168,IDN(1,L)
0340              JIDN=J-1
0341          168 FORMAT (3X,'TEC ',A4,' IS INVALID')
            C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
            C----- THE IBNDA, IRIGHT AND MIXUP ARRAYS
            C-----          ARE CHANGED FROM THE TEC SUPPLIED BY THE
            C-----  USER TO THE CODE IN THE DICTIONARY.
0342          140 IF (JBNDS.EQ.0) GO TO 145
0343              DO 144 J=1,JBNDS
0344              NN=2
0345              CALL SEARCH (IBNDA(1,J),NN)
0346              IF (NN.NE.0) GO TO 143
0347              PRINT 142,IBNDA(1,J)
0348          142 FORMAT (3X,'BOUND EXCEPTION TEC ',A4,' IS INVALID')
0349              ISTOP=1
0350              GO TO 144
0351          143 IBNDA(1,J)=IDICT(1,NN)
0352          144 CONTINUE
0353          145 IF (NRS.EQ.0) GO TO 125
0354              DO 149 K=1,NRS
0355              KK=NRIGHT(K)+1
0356              DO 149 J=2,KK
0357              NN=2
0358              CALL SEARCH (IRIGHT(1,J,K),NN)
0359              IF (NN.NE.0) GO TO 147
0360              PRINT 146,IRIGHT(1,J,K)
0361          146 FORMAT (3X,'TEC ',A4,' FOR PARAMETRIC RIGHT HAND SIDE IS INVALID')
0362              ISTOP=1
0363              GO TO 149
0364          147 IRIGHT(1,J,K)=IDICT(1,NN)
0365          149 CONTINUE
0366          125 IF (JDBC.EQ.0) GO TO 180
0367              DO 129 J=1,JDBC
0368              NN=2
0369              CALL SEARCH(MIXUP(1,J),NN)
0370              IF (NN.NE.0) GO TO 128
```

```
0371            PRINT 168,MIXUP(1,J)
0372            ISTOP=1
0373            GO TO 129
0374          128 MIXUP(1,J)=IDICT(1,NN)
0375          129 CONTINUE
              C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
              C---- EVERY ROW NAME AND ITS TYPE ARE ESTABLISHED HERE
0376          180 CONTINUE
0377          187 WRITE(LUO,181)
0378          181 FORMAT(4HNAME,10X,5HMINUM)
0379              WRITE(LUO,182)
0380          182 FORMAT(4HROWS)
0381              WRITE(LUO,183)
0382          183 FORMAT(1X,1HN,2X,4HCOST)
0383              DO 189 L=IYA,IYB
0384              DO 189 J=1,NNARFS
0385              DO 189 K=1,NSHOPS
0386              DO 186 I=8,10
0387          186 WRITE(LUO,184)JDENT(5),JDENT(I),NARFX(J),K,L
0388          184 FORMAT(1X,A1,2X,2A1,2I1)
0389              M=K
0390              IF(INDS.EQ.2)GO TO 185
0391              IF(K.GT.1)GO TO 189
0392              M=0
0393          185 DO 188 I=22,24
0394          188 WRITE(LUO,184)JDENT(I),JDENT(I+3),NARFX(J),M,L
0395          189 CONTINUE
0396          190 LASTID(1)=JDENT(7)
0397              DO 199 J=1,JIDN
0398              DO 193 K=1,7
0399              ITEMQ=LJABF(LASTID(1),K)
0400              ITEMP=LJABF(IDN(1,J),K)
0401              IF(ITEMP.EQ.ITEMQ)GO TO 193
0402              DO 191 L=IYA,IYB
0403          191 WRITE(LUO,192)IDN(1,J),L,IDN(2,J)
0404          192 FORMAT(1X,1HE,2X,A3,I1,A3)
0405              DO 194 L=1,2
0406          194 LASTID(L)=IDN(L,J)
0407              GO TO 195
0408          193 CONTINUE
0409          195 IF(IBOP.EQ.1)GO TO 199
0410              ITEMQ=LJABF (IDN(1,J),8)
0411              DO 198 M=1,NFACIL
0412              IF(ITEMQ.NE.NARFX(M))GO TO 198
0413              DO 196 L=IYA,IYB
0414              DO 196 K=1,2
0415          196 WRITE(LUO,197)JDENT(K+10),IDN(1,J),L,IDN(2,J),NARFY(M,K)
0416          197 FORMAT(1X,A1,2X,A3,I1,A3,A1)
0417              GO TO 199
0418          198 CONTINUE
0419          199 CONTINUE
              C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
              C---- INITIALIZATION JUST PRIOR TO DATA BASE PROCESSING IS DONE HERE
0420              WRITE(LUO,200)
0421          200 FORMAT(7HCOLUMNS)
```

```
0422            DO 202 J=1,2
0423          202 IDN(J,JIDN+1)=JDENT(7)
0424            IMOV=2
0425            IEND=0
0426            KIDN=1
0427            ISET=1
0428            IF (JDBC.GT.0) PRINT 201
0429          201 FORMAT (1H0,32X,'DATA BASE UPDATES'//,2X,'***  ID NARE  ',
                *           '***  ELEMENT  YEAR  ***********  OLD VALUE  *********
               ****  NEW VALUE  ***'/)
              C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
              C---- CONSECUTIVE REQUESTS WITH THE SAME ID ARE PUT INTO
              C---- THE APPROPRIATE POSITION IN ARRAY IDNA. THE ID IS
              C---- NOTED IN ARRAY LASTID
0430          220 IF (IMOV.GT.2) GO TO 230
0431            DO 221 L=1,NFACIL
0432            DO 221 K=1,NYEARS
0433            DO 221 J=1,2
0434          221 IDNA (J,K,L)=0
0435            DO 228 J=1,2
0436          228 LASTID(J)=IDN(J,KIDN)
0437            IF (KIDN.GT.JIDN) GO TO 230
0438            GO TO 224
0439          222 DO 223 J=1,7
0440            ITEMQ=LJABF (IDN(1,KIDN),J)
0441            ITEMP=LJABF (LASTID(1),J)
0442            IF (ITEMP.NE.ITEMQ) GO TO 230
0443          223 CONTINUE
0444          224 ITEMP=LJABF (IDN(1,KIDN),8)
0445            DO 229 L=1,NFACIL
0446            IF(ITEMP.NE.NAREX(L))GO TO 229
0447            DO 227 K=IYA,IYB
0448            DO 225 J=1,2
0449          225 IDNA (J,K,L)=IDN(J,KIDN)
0450            CALL MOVE (IOTF(K),1,IDNA(1,K,L),4,1)
0451          227 CONTINUE
0452            GO TO 219
0453          229 CONTINUE
0454          219 KIDN=KIDN+1
0455            GO TO 222
              C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
              C---- CONSECUTIVE RECORDS IN THE DATA BASE HAVING THE SAME ID ARE PUT
              C---- INTO THE APPROPRIATE POSITION IN ARRAYS IRECA AND RECB.
              C---- THE ID IS NOTED IN ARRAY KASTID
0456          230 IF (IMOV.LT.2) GO TO 248
0457            DO 231 L=1,NFACIL
0458            DO 231 K=1,NYEARS
0459            DO 217 J=1,2
0460          217 IRECA (J,K,L)=0
0461            DO 231 J=1,IWPR
0462          231 RECB(J,K,L)=0.
0463            IF (ISET.LT.1) GO TO 240
0464          232 DO 235 K=1,NYEARS
0465          235 READ (LUI,233,END=210)(ITMP(J,K),J=1,2),(TMP(J,K),J=1,IWPR)
0466          233 FORMAT (2A4,9F3.0,2F4.0,F3.0,F5.0,F3.0,
```

```
                *F6.0,F4.0,F5.0,F4.0,F5.0,F6.0,F9.0)
0467              GO TO 218
0468          210 IEND=1
0469              ISET=0
0470              DO 211 J=1,2
0471          211 ITMP(J,IYA)=JDENT(7)
0472              GO TO 250
0473          218 ITEMP=LJABF (ITMP(1,IYA),8)
0474              DO 234 L=1,NFACIL
0475              IF(ITEMP.EQ.NAREX(L))GO TO 237
0476          234 CONTINUE
0477          237 IF (L.GT.NNARES) GO TO 243
0478              DO 226 K=1,NYEARS
0479              IF (GAR(L,K).EQ.0.) GAR(L,K)=TMP(KG,K)/10000.
0480          226 CONTINUE
0481          243 LN=L
0482              IF (ISET.EQ.2) GO TO 239
0483          240 DO 238 J=1,2
0484          238 KASTID (J)=ITMP(J,IYA)
0485              IF (IEND.EQ.1) GO TO 248
0486              GO TO 236
0487          239 DO 245 J=1,7
0488              IF (J.EQ.4) GO TO 245
0489              ITEMP=LJABF(KASTID(1),J)
0490              ITEMQ=LJABF (ITMP(1,IYA),J)
0491              IF (ITEMP.EQ.ITEMQ) GO TO 245
0492              ISET=0
0493              GO TO 250
0494          245 CONTINUE
0495          236 ISET=2
0496              DO 242 K=IYA,IYB
0497              DO 241 J=1,2
0498          241 IRECA(J,K,LN)=ITMP(J,K)
0499              DO 242 J=1,IWPR
0500          242 RECB(J,K,LN)=TMP(J,K)
0501              GO TO 232
            C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
            C---- THE DATA BASE UPDATE ARRAY MIXUP IS CHECKED FOR ANY ID MATCHES
0502          250 IF (JDBC.EQ.0) GO TO 2520
0503              NN=1
0504              CALL SEARCH (KASTID(1),NN)
0505              IF (NN.EQ.0) NN=NDR+1
0506              DO 2500 J=1,NYEARS
0507         2500 ITQS(J)=0
0508              DO 2510 J=1,JDBC
0509              DO 2504 I=1,7
0510              IF (I.EQ.4) GO TO 2504
0511              ITEMP=LJABF(MIXUP(1,J),I)
0512              ITEMQ=LJABF(KASTID(1),I)
0513              IF (ITEMP.NE.ITEMQ) GO TO 2510
0514         2504 CONTINUE
0515              MXX=IYA
0516              NXX=IYB
0517              ISETR=0
0518              IF (MIXUP(3,J).GT.NYEARS) ISETR=1
```

```
0519              ITEMP=MOD(MIXUP(3,J),NYEARS+1)
0520              IF (ITEMP.EQ.0) GO TO 2502
0521              MXX=ITEMP
0522              NXX=ITEMP
0523         2502 ITEMP=LJABF(MIXUP(1,J),8)
0524              DO 2505 L=1,NFACIL
0525              IF (ITEMP.EQ.NARFX(L)) GO TO 2506
0526         2505 CONTINUE
0527         2506 DO 2509 K=MXX,NXX
0528              IF (ISETR.EQ.0) GO TO 2503
0529              PRINT 2501,IDICT(2,NN),KASTID(2),NARFX(L),IDBREN(22),K
0530              DO 2515 M=1,2
0531         2515 IRECA(M,K,L)=0
0532              GO TO 2505
0533         2503 IF (IRECA(1,K,L).NE.0) GO TO 2517
0534              DO 2516 M=1,2
0535         2516 IRECA(M,K,L)=MIXUP(M,J)
0536              CALL MOVE (IOTF(K),1,IRECA(1,K,L),4,1)
0537         2517 DO 2507 I=1,20
0538         2507 ICHK(I)=0
0539              DO 2508 I=1,20
0540              IF (FIXUP(I,J).EQ.-1.) GO TO 2508
0541              IF (I.EQ.KR.OR.I.EQ.KG) GO TO 2508
0542              IF (I.EQ.10) GO TO 2521
0543              PRINT 2501,IDICT(2,NN),KASTID(2),NARFX(L),
                 *IDBREN(I),K,RECB(I,K,L ),FIXUP(I,J)
0544         2501 FORMAT (6X,A4,A3,A1,9X,A4,5X,I1,2(5X,F20.0))
0545              RECB(I,K,L)=FIXUP(I,J)
0546              GO TO 2523
0547         2521 IF (ITQS(K).EQ.1) GO TO 2508
0548              ITQS(K)=1
0549              DO 2522 LL=1,NFACIL
0550              PRINT 2501,IDICT(2,NN),KASTID(2),NARFX(LL),
                 *IDBREN(I),K,RECB(I,K,LL),FIXUP(I,J)
0551              RECB(I,K,LL)=FIXUP(I,J)
0552         2522 CONTINUE
0553         2523 ICHK(I)=1
0554         2508 CONTINUE
0555              ITEMR=LJABF(MIXUP(1,J),5)
0556              DO 2513 I=10,14
0557              IF (I.EQ.13) GO TO 2513
0558              IF (ICHK(I).EQ.1) GO TO 2514
0559         2513 CONTINUE
0560              GO TO 2509
0561         2514 TEMP=RECB(KR,K,L)
0562              IF (ITEMR.LT.IPROGS(3).OR.ITEMR.GT.IPROGS(5)) GO TO 2512
0563              RECB(KR,K,L)=(RECB(11,K,L)*RECB(12,K,L)+RECB(14,K,L)
                 **(RECB(10,K,L)-RECB(11,K,L)))
0564              GO TO 2511
0565         2512 RECB(KR,K,L)=(RECB(12,K,L)+RECB(14,K,L))*100.
0566         2511 PRINT 2501,IDICT(2,NN),KASTID(2),NARFX(L),
                 *IDBREN(KR),K,TEMP,RECB(KR,K,L)
0567         2509 CONTINUE
0568         2510 CONTINUE
            C** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** **
```

```
C---- THE ASSUMED DECIMAL DATA IS RECONVERTED
0569    2520 ITEMR=LJABF(KASTID(1),5)
0570         ISETR=0
0571         IF (ITEMR.EQ.IPROGS(1).OR.ITEMR.EQ.IPROGS(2).OR.
            *ITEMR.EQ.IPROGS(8)) ISETR=1
0572         DO 252 L=1,NFACIL
0573         DO 252 K=IYA,IYB
0574         DO 251 J=1,NSHOPS
0575     251 RECB(J,K,L)=RECB(J,K,L)*RECB(13,K,L)/100.
0576         RECB(KR,K,L)=RECB(KR,K,L)/100.
0577         PROD=0.
0578         IF (RECB(13,K,L).EQ.0.) GO TO 252
0579         SUM=RECB(16,K,L)+RECB(18,K,L)
0580         IF (L.LE.NNARFS) SUM=SUM+GAR(L,K)*100.
0581         IF (ISETR.EQ.1) GO TO 2518
0582         PROD=RECB(13,K,L)*(SUM+RECB(17,K,L))/100.
0583         GO TO 252
0584    2518 PROD=RECB(20,K,L)+RECB(13,K,L)*SUM/100.
0585     252 RECB(KC,K,L)=PROD
0586         IF (ITEMR.GE.IPROGS(3).AND.ITEMR.LE.IPROGS(5)) GO TO 248
0587         DO 247 K=1,NYEARS
0588         DO 247 L=1,NFACIL
0589         ITEMP=RECB(KR,K,L)+.5
0590     247 RECB(KR,K,L)=ITEMP
C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
C---- THE ARRAYS LASTID AND KASTID ARE COMPARED FOR
C---- ID EQUALITY. IF LASTID IS LESS THAN KASTID, AN
C---- ERROR MESSAGE IS PRINTED. (THIS SITUATION SHOULD
C---- ONLY OCCUR IF AN MISTAKE
C---- WAS MADE WHEN REQUESTING THE PROGRAM (5), SUBPROGRAM (6)
C---- AND/OR CUSTOMER (7). A INVALID TEC (1-4) WOULD HAVE
C---- BEEN ELIMINATED PREVIOUSLY)
0591     248 DO 255 J=1,7
0592         IF (J.EQ.4) GO TO 255
0593         ITEMP=LJABF(LASTID(1),J)
0594         ITEMQ=LJABF(KASTID(1),J)
0595         IF (ITEMP - ITEMQ) 254,255,253
0596     255 CONTINUE
0597         IF (JDENT(7).EQ.LASTID(1)) GO TO 400
0598         IMOV=2
0599         GO TO 246
0600     254 NN=1
0601         CALL SEARCH (LASTID(1),NN)
0602         IF (NN.EQ.0) GO TO 258
0603         PRINT 259,IDICT(2,NN),LASTID(2)
0604     259 FORMAT (3X,'ALL REQUESTS FOR ID ',A4,A3,   ' ARE INVALID')
0605     258 IMOV=1
0606         GO TO 220
0607     253 IMOV=3
C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
C---- PROCESSING IS BY YEAR AND ONLY FOR THE YEARS REQUESTED
0608     246 DO 360 K=IYA,IYB
0609         IF (IMOV.EQ.2) GO TO 286
0610         NBAD(13)=0
0611         NMATCH(13)=0
```

```
0612              II=0
0613              DO 256 L=1,NFACIL
0614              IF (IRECA(1,K,L).EQ.0) GO TO 256
0615              II=II+1
0616              NBASE(II)=L
0617            256 CONTINUE
0618              NBASE(13)=II
0619              GO TO 272
                C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
                C---- IF A MATCH OCCURRED (LASTID=KASTID), CONSECUTIVE LOCATIONS
                C---- OF ARRAYS NMATCH, NBASE AND NBAD ARE USED TO HOLD THE
                C---- NUMBER OF THE NARF (1-12 FOR A THRU L) ON WHICH
                C---- 1) A MATCH OCCURRED, 2) A DATA BASE RECORE EXISTS
                C---- FOR WHICH THERE IS NO REQUEST AND 3) A REQUEST
                C---- EXISTS WHEN THERE IS NO CORRESPONDING DATA BASE RECORD.
                C---- A MATCH MEANS THAT ALL EIGHT CHARACTERS ARE
                C---- EQUAL.
                C---- FOLLOWING THIS SECTION THE NUMBERS IN NMATCH, NBASE AND
                C---- NBAD ARE USED TO DETERMINE WHICH RECORDS WILL BECOME
                C---- PART OF THE MATRIX (NMATCH), WHICH RECORDS ARE FOR
                C---- BASE INFORMATION (NBASE) AND WHICH ARE ERRORS (NBAD)
0620            286 DO 297 L=1,13
0621              NMATCH(L)=0
0622              NBASE(L)=0
0623            297 NBAD(L)=0
0624              I=0
0625              II=0
0626              III=0
0627              DO 293 L=1,NFACIL
0628              IF (IDNA(1,K,L).EQ.C.AND.IRECA(1,K,L).EQ.0) GO TO 293
0629              IF (IDNA(1,K,L).EQ.C) GO TO 294
0630              IF (IRECA(1,K,L).EQ.0) GO TO 295
0631              DO 292 J=1,2
0632              IF (IDNA(J,K,L).EQ.IRECA(J,K,L)) GO TO 292
0633              PRINT 296,(IDNA(N,K,L),N=1,2),(IRECA(N,K,L),N=1,2)
0634            296 FORMAT (3X,'ID - MATCH.   IDN - NO MATCH.   ',2(2X,2A4))
0635              GO TO 291
0636            292 CONTINUE
0637            291 I=I+1
0638              NMATCH(I)=L
0639              GO TO 293
0640            294 II=II+1
0641              NBASE(II)=L
0642              GO TO 293
0643            295 III=III+1
0644              NBAD(III)=L
0645            293 CONTINUE
0646              NMATCH(13)=I
0647              NBASE(13)=II
0648              NBAD(13)=III
                C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
                C---- THIS SECTION COMPUTES THE SUM OF ALL REQUIREMENTS FOR THE GIVEN
                C---- YEAR IN WHICH MATCHES OCCURRED. IF THE SUM IS ZERO,
                C---- NO DATA EXISTS AND NO MATCHES WILL BECOME PART
                C---- OF THE MATRIX (SET NMATCH (13)=0).
```

```
0649          265 IF (NMATCH(13).EQ.0) GO TO 249
0650              I=1
0651              SUM=0.
0652              DO 257 L=1,NFACIL
0653              IF (NMATCH(I).NE.L) GO TO 257
0654              SUM=SUM+RECB(KR,K,L)
0655              I=I+1
0656          257 CONTINUE
0657              IF (SUM.NE.0.) GO TO 268
0658              NMATCH(13)=0
0659              GO TO 249
0660          268 JID=JID+1
0661              CALL MOVE (IOTF(K),1,LASTID(1),4,1)
0662              DO 264 J=1,2
0663          264 IDSAVE(J,JID)=LASTID(J)
0664              REQID(JID)=SUM
             C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
             C---- PRINT REQUEST FOR WHICH THERE IS NO
             C---- DATA BASE RECORD
0665          249 MMM=NBAD(13)
0666              IF (MMM.EQ.0) GO TO 272
0667              DO 269 MM=1,MMM
0668              M=NBAD(MM)
0669              NN=1
0670              CALL SEARCH (IDNA(1,K,M),NN)
0671              IF (NN.EQ.0) GO TO 269
0672              PRINT 271,IDICT(2,NN),IDNA(2,K,M),K
0673          271 FORMAT (3X,'REQUESTED IDN ',2A4,' IN YEAR ',I1,
                 *' IS NOT IN THE DATA BASE')
0674          269 CONTINUE
             C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
             C---- ACCOUNT FOR ALL BASE DATA. THAT IS, ADJUST
             C---- ARRAYS CMLA AND CAPB AND ACCUMMULATE IN ARRAY SUMN
0675          272 MMM=NBASE(13)
0676              IF (MMM.EQ.0) GO TO 300
0677              DO 298 MM=1,MMM
0678              M=NBASE(MM)
0679              IF (M.GT.NNARFS) GO TO 274
0680              DO 273 J=1,NSHOPS
0681              PROD=RECB(J,K,M)*RECB(KR,K,M)
0682              CAPB (K,J,M)=CAPB(K,J,M)-PROD
0683              CMLA(J,M,K)=CMLA(J,M,K)-PROD
0684          273 CONTINUE
0685          274 ITEMP=LJABF (IRECA(1,K , M),5)
0686              DO 277 J=1,NPROGS
0687              IF (ITEMP.EQ.IPROGS(J)) GO TO 278
0688          277 CONTINUE
0689          278 NP=J
0690              ITEMP=LJABF (IRECA(1,K,M),3)
0691              DO 279 J=1,NFUNDS
0692              IF (ITEMP.EQ.IFUNDS(J)) GO TO 283
0693          279 CONTINUE
0694          283 NF=J
0695          284 SUMN(NF,NP,M,K)=SUMN(NF,NP,M,K)+RECB(KR,K,M)*RECB(KC,K,M)
0696          298 CONTINUE
```

```
C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
C---- THE MATCHES ARE PROCESSED HERE. VARICUS RECORDS ARE
C---- WRITTEN INTO THE COLUMN SECTION, ONE COLUMN AT A TIME
C---- THE TOTAL NUMBER OF COLUMNS PROCESSED FOR A GIVEN ID
C---- IS THE PRODUCT OF THE NUMBER OF YEARS AND THE
C---- NUMBER OF MATCHES. IF BOUNDS ARE USED, REQUIREMENTS
C---- AND THE SUM OF REQUIREMENTS ARE SAVED IN ARRAYS
C---- IBOUND AND BOUNDS. OTHERWISE PARAMETRIC EQUATIONS
C---- ARE USED AND VALUES ARE INCLUDED UNDER THE
C---- APPROPRIATE ROW NAMES. THE VALUES ARE COMPUTED AND
C---- PUT INTO ARRAY BOUNDA
0697        300 MMM=NMATCH(13)
0698            IF (MMM.EQ.0) GO TO 360
0699            DO 340 MM=1,MMM
0700            M=NMATCH(MM)
0701            IF (RECB(KC,K,M).EQ.0.) GO TO 340
0702            IF (M.GT.NNARFS) GO TO 305
0703            DO 304 J=1, NSHOPS
0704            IF (RECB (J,K,M).NE.0.) GO TO 305
0705        304 CONTINUE
0706            GO TO 340
0707        305 WRITE (LUO,301) (IDNA(J,K,M),J=1,2),(IDNA(J,K,M),J=1,2)
0708        301 FORMAT (4X,2A4,2X,A4,A3,14X,1H1)
0709            WRITE (LUO,302) (IDNA(J,K,M),J=1,2), RECB (KC,K,M)
0710        302 FORMAT (4X,2A4,2X,4HCOST,6X,F12.2)
0711            IF (M.GT.NNARFS) GO TO 320
0712            DO 307 J=1,NSHOPS
0713            IF (RECB(J,K,M).EQ.0.) GO TO 307
0714            WRITE(LUO,306)(IDNA(L,K,M),L=1,2),JDENT(8),NARFX(M),J,K,RECB
               *(J,K,M)
0715        306 FORMAT(4X,2A4,2X,2A1,2I1,6X,F12.2)
0716        307 CONTINUE
0717            DO 315 J=1,NSHOPS
0718            IF (INOS.EQ.1) GO TO 310
0719            PROD=RECB(J,K,M)
0720            JJ=J
0721            GO TO 312
0722        310 PROD=0.
0723            JJ=0
0724            DO 311 L=1,NSHOPS
0725        311 PROD=PROD+RECB(L,K,M)
0726        312 DO 314 I=13,13
0727            IF (PROD.EQ.0.) GO TO 314
0728            WRITE(LUO,313)(IDNA(L,K,M),L=1,2),JDENT(I),NARFX(M),JJ,K,PROD
0729        313 FORMAT(4X,2A4,2X,2A1,2I1,6X,F12.2)
0730        314 CONTINUE
0731            IF (INOS.EQ.1) GO TO 320
0732        315 CONTINUE
0733        320 IF (IBOP. NE. 1) GO TO 324
0734            JBDS=JBDS+1
0735            DO 322 J=1,2
0736        322 IBOUND (J,JBDS)=IDNA(J,K,M)
0737            BOUNDS (1,JBDS)=RECB (KR,K,M)
0738            BOUNDS (2,JBDS)=SUM
0739            GO TO 340
```

```
0740          324 DO 333 L=1,MMM
0741              LL=NMATCH (L)
0742              BOUNDA(2)=-((RECB(KR,K,LL)/SUM)-PVAR(K))
0743              BOUNDA(1)=-((RECB(KR,K,LL)/SUM)+PVAR(K))
0744              IF (M.NE.LL) GO TO 328
0745              BOUNDA (1)=1+ BOUNDA(1)
0746              BOUNDA (2)=1 + BOUNDA (2)
0747          328 DO 330 J=1,2
0748          330 WRITE(LUO,329)(IDNA(I,K,M),I=1,2),(IDNA(I,K,M),I=1,2),NARFY(LL,J),
                 *BOUNDA(J)
0749          329 FORMAT (4X, 2A4,2X,A4,A3,A1,2X,F12.2)
0750          333 CONTINUE
0751          340 CONTINUE
0752          360 CONTINUE
0753              GO TO 220
              C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
              C---- AFTER PROCESSING THE DATA BASE, THE VALUE WHICH
              C---- CONTROL SHIFT OVERFLOWS AND HIRING FIRING ARE
              C---- WRITTEN INTO THE MATRIX.
0754          400 DO 415 K=IYA,IYB
0755              DO 415 L=1, NNARFS
0756              DO 415 J=1,NSHOPS
0757              WRITE(LUO,404)NARFX(L),J,K,COSTB(K,J,L,2)
0758          404 FORMAT(4X,1HU,A1,2I1,6X,4HCOST,6X,F12.2)
0759              WRITE(LUO,405)NARFX(L),J,K,NARFX(L),J,K,EF2(K)
0760          405 FORMAT(4X,1HU,A1,2I1,6X,1HF,A1,2I1,13X,1H-,F4.2)
0761              WRITE(LUO,406)NARFX(L),J,K,NARFX(L),J,K
0762          406 FORMAT(4X,1HU,A1,2I1,6X,1HS,A1,2I1,17X,1H1)
0763              M=0
0764              IF (INOS.EQ.2) M=J
0765          402 BVAL=1-EF2(K)
0766              WRITE (LUO,401) NARFX(L),J,K,NARFX(L),M,K,BVAL
0767          401 FORMAT (4X,1HU,A1,2I1,6X,1HM,A1,2I1,6X,F12.2)
0768              BVAL=1-EF3(K)
0769              WRITE(LUO,412)NARFX(L),J,K,NARFX(L),M,K,BVAL
0770          412 FORMAT (4X,1HV,A1,2I1,6X,1HM,A1,2I1,6X,F12.2)
0771          403 WRITE(LUO,407)NARFX(L),J,K,NARFX(L),J,K,EF3(K)
0772          407 FORMAT(4X,1HV,A1,2I1,6X,1HF,A1,2I1,13X,1H-,F4.2)
0773              WRITE(LUO,408)NARFX(L),J,K,NARFX(L),J,K
0774          408 FORMAT(4X,1HV,A1,2I1,6X,1HT,A1,2I1,17X,1H1)
0775              WRITE(LUO,409)NARFX(L),J,K,COSTB(K,J,L,3)
0776          409 FORMAT(4X,1HV,A1,2I1,6X,4HCOST,6X,F12.2)
0777              WRITE(LUO,410)NARFX(L),J,K,NARFX(L),J,K
0778          410 FORMAT(4X,1HW,A1,2I1,6X,1HF,A1,2I1,16X,2H-1)
0779              WRITE(LUO,411)NARFX(L),J,K,COSTB(K,J,L,4)
0780          411 FORMAT(4X,1HW,A1,2I1,6X,4HCOST,6X,F12.2)
0781          415 CONTINUE
0782              DO 440 L=1,NNARFS
0783              DO 439 K=IYA, IYB
0784              DO 438 J=1,NSHOPS
0785              DO 422 I=1,4
0786          422 HCOST(I)=COSTB(K,J,L,I+4)
0787              M=J
0788              IF (INOS.EQ.2) GO TO 430
0789              IF (J.GT.1) GO TO 439
```

```
0790              M=0
0791              DO 424 I=1,4
0792              HCOST (I)=0.
0793              DO 423 N=1,NSHOPS
0794            423 HCOST(I)=HCOST(I)+COSTB(K,N,L,I+4)
0795            424 HCOST(I)=HCOST(I)/NSHOPS
0796            430 DO 437 I=1,4
0797                IF (HCOST(I).EQ.0.) GO TO 436
0798                WRITE (LUO,435) JDENT(I+27),NAREX(L),M,K,HCOST(I)
0799            435 FORMAT (4X,2A1,2I1,6X,4HCOST,6X,F12.2)
0800            436 IF (I.LT.3) GO TO 434
0801                WRITE (LUO,432) JDENT(I+27),NAREX(L),M,K,
                   *JDENT(I+23),NAREX(L),M,K
0802            432 FORMAT (4X,2A1,2I1,6X,2A1,2I1,16X,2H+1)
0803            434 DO 437 KKK=K,IYB
0804                WRITE (LUO,431) JDENT(I+27),NAREX(L),M,K,
                   *NAREX(L),M,KKK,JDENT(I+31)
0805            431 FORMAT (4X,2A1,2I1,6X,1HM,A1,2I1,16X,A1,1H1)
0806            437 CONTINUE
0807            438 CONTINUE
0808            439 CONTINUE
0809            440 CONTINUE
              C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
              C---- THE RIGHT HAND SIDE IS WRITTEN IN THIS SECTION
0810            500 WRITE (LUO,501)
0811            501 FORMAT (3HRHS)
0812                DO 508 K=1,JID
0813            508 WRITE (LUO, 507) (IDSAVE(J,K),J=1,2),REQID( K )
0814            507 FORMAT (4X, 5HRIGHT,5X,A4,A3,3X,F12.2)
0815                DO 518 M=IYA,IYB
0816                DO 518 L=1,NNAREAS
0817                DO 518 K=1,NSHOPS
0818                WRITE(LUO,511)NAREX(L),K,M,CAPB(M,K,L)
0819            511 FORMAT(4X,5HRIGHT,5X,1HF,A1,2I1,6X,F12.2)
0820                PERCAP=PCAP2(M)*TOTCAP(M,K,L)
0821                WRITE(LUO,512)NAREX(L),K,M,PERCAP
0822            512 FORMAT(4X,5HRIGHT,5X,1HS,A1,2I1,6X,F12.2)
0823                PERCAP=PCAP3(M)*TOTCAP(M,K,L)
0824                WRITE(LUO,513)NAREX(L),K,M,PERCAP
0825            513 FORMAT(4X,5HRIGHT,5X,1HT,A1,2I1,6X,F12.2)
0826            518 CONTINUE
0827                DO 535 M=IYA,IYB
0828                DO 535 L=1, NNAREAS
0829                DO 530 K=1,NSHOPS
0830                SUMR(1)=CMLA(K,L,M)
0831                SUMR(2)=CML (K,L,M)*XUML
0832                SUMR(3)=CML (K,L,M)*XLML
0833                KK=K
0834                IF (INOS.EQ.2) GO TO 526
0835                KK=0
0836                DO 521 J=1,3
0837            521 SUMR(J)=0.
0838                DO 522  J=1,NSHOPS
0839                SUMR(1)=SUMR(1)+CMLA(J,L,M)
0840                SUMR(2)=SUMR(2)+CML (J,L,M)*XUML
```

```
0841          522 SUMR(3)=SUMR(3)+CML (J,L,M)*XLML
0842          526 DO 528 J=1,3
0843          528 WRITE(LUO,527)JDENT(J+24),NAREX(L),KK,M,SUMR(J)
0844          527 FORMAT(4X,5HRIGHT,5X,2A1,2I1,6X,F12.2)
0845              IF (INOS.EQ.1) GO TO 535
0846          530 CONTINUE
0847          535 CONTINUE
0848              IF (IBOP.EQ.1) GO TO 542
0849              DO 539 J=1,NRS
0850              K=NRIGHT (J)+1
0851              IF (K.EQ.1) GO TO 539
0852              DO 538 L=2,K
0853              DO 538 I=IYA,IYB
0854          538 WRITE(LUO,537)(IRIGHT(M,1,J),M=1,2),IRIGHT(1,L,J),I,IRIGHT(2,L,J),
                 *FRIGHT(I,L-1,J)
0855          537 FORMAT (4X,2A4,2X,A3,I1,A4,2X,F12.2)
0856          539 CONTINUE
0857              GO TO 800
              C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
              C---- BOUNDS, IF ANY, ARE GENERATED HERE
0858          542 IF (JBDS.EQ.0) GO TO 800
0859              IF (NBS.EQ.0) GO TO 800
0860              WRITE (LUO, 543)
0861          543 FORMAT (6HBOUNDS)
0862              DO 590 L=1,NBS
0863              DO 580 J=1,JBDS
0864              ITEMR=LJABF(IBOUND(1,J),5)
0865              ITEMP=LJABF(IBOUND(1,J),4)
0866              DO 546 I=IYA,IYB
0867              IF (ITEMP.EQ.IOTF(I)) GO TO 548
0868          546 CONTINUE
0869          548 DO 571 K=1,2
0870              IF (JBNDS.EQ.0) GO TO 558
0871              DO 560 M=1,JBNDS
0872              ITEMP=IBNDC (M)/10
0873              IF (ITEMP-L) 560,547,558
0874          547 DO 550 N=1,8
0875              IF (N.EQ.4) GO TO 550
0876              ITEMP = LJABF (IBOUND(1,J),N)
0877              ITEMQ=LJABF (IBNDA(1,M),N)
0878              IF (ITEMP.NE.ITEMQ) GO TO 560
0879          550 CONTINUE
0880              ITEMP=MOD(IBNDC(M),10)
0881              IF (K.EQ.2) GO TO 555
0882              IF (ITEMP.EQ.2) GO TO 558
0883              BVAL=BOUNDS(1,J)-BOUNDS(2,J)*BNDB(I,M)
0884              GO TO 553
0885          555 IF (ITEMP.EQ.1) GO TO 558
0886              BVAL=BOUNDS(1,J)+BOUNDS(2,J)*BNDB(I,M)
0887          553 IF (BNDB(I,M).LT.0.) BVAL=0.
0888              GO TO 566
0889          560 CONTINUE
0890          558 IF (K.EQ.2) GO TO 564
0891              BVAL=BOUNDS(1,J)-BOUNDS(2,J)*BNDY(I,L)
0892              GO TO 566
```

```
0893          564 BVAL=BOUNDS(1,J)+BOUNDS(2,J)*BNDY(I,L)
0894          566 IF (BVAL.LT.0.) BVAL=0.
0895              IF (ITEMR.GE.IPROGS(3).AND.ITEMR.LE.IPROGS(5)) GO TO 569
0896              ITEMP=BVAL
0897              BVAL=ITEMP
0898          569 WRITE(LUO,567)JDENT(K+35),((BNDX(N,L),N=1,2),(IBOUND(N,J),N=1,2),
                 1BVAL
0899          567 FORMAT (1X,A2,1X,2A4,2X,2A4,2X,F12.2)
0900          571 CONTINUE
0901          580 CONTINUE
0902          590 CONTINUE
0903          800 WRITE (LUO, 801)
0904          801 FORMAT (6HENDATA)
            C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
            C---- THE BASE INFORMATION IS WRITTEN ONTO THE FILE FOR
            C---- USE IN THE REPORT GENERATOT.
0905              WRITE (LUU,808) IYA,IYB,IBOP,INOS,XLML,XUML,IAY
0906          808 FORMAT (4I5,2F6.3,I5)
0907              WRITE (LUU,851) EF2,EF3
0908              DO 803 J=1,NYEARS
0909          803 WRITE(LUU,811)(HPY(L,J),L=1,NNARFS),PHPY(J)
0910              DO 810 M=IYA,IYB
0911              DO 810 L=1,NFUNDS
0912              DO 810 J=1,NPROGS
0913          810 WRITE (LUU, 811) (SUMN(L,J,K,M),K=1,NFACIL)
0914          811 FORMAT (12F11.0)
0915              DO 820 K=1,NSHOPS
0916              DO 820 J=1,NYEARS
0917          820 WRITE (LUU,811) (CAPB(J,K,L),L=1,NNARFS)
0918              DO 830 K=1,NSHOPS
0919              DO 830 J=1,NYEARS
0920          830 WRITE (LUU,811) (TOTCAP(J,K,L),L=1,NNARFS)
0921              DO 840 J=1,NSHOPS
0922          840 WRITE (LUU,811) (CML(J,K,I),K=1,NNARFS)
0923              DO 850 K=1,NYEARS
0924          850 WRITE (LUU,851) (GAR(J,K),J=1,NNARFS)
0925          851 FORMAT (10F11.4)
0926              STOP
0927              END
```

ANNEX A-4

REPORT GENERATOR

Move
IBYTE
LJABF
Search
CVADJ
Main

```
 LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                    F15OCT70  2/23/73

000000                             1          START
                                   2          ENTRY MOVE,IBYTE,LJABF
000000 47FF 000A           0000A   3 MOVE     BC    15,10(15)
000004 05                          4          DC    X'5'
000005 D4D6E5C540                  5          DC    CL5'MOVE'
00000A 90EC D00C           0000C   6          STM   14,7,12(13)
00000E 0520                        7          BALR  2,0
000010                             8          USING *,2
000010 9837 1000           00000   9          LM    3,7,0(1)
000014 5844 C000           C0000  10          L     4,0(4)
000018 5866 0000           00000  11          L     6,0(6)
00001C 5877 0000           00000  12          L     7,0(7)
000020 5840 209A           000A8  13          S     4,=F'1'
000024 5860 209E           000A8  14          S     6,=F'1'
000028 5870 2098           000A8  15          S     7,=F'1'
00002C 4240 202D           0002D  16          STC   4,45(0,2)
000030 4260 202B           0002B  17          STC   6,43(0,2)
000034 4270 2029           00029  18          STC   7,41(0,2)
000038 D200 5000 3000 00000 00000 19          MVC   0(0,5),0(3)
00003E 98EC D00C           0000C  20          LM    14,7,12(13)
000042 92FF D00C      0000C        21          MVI   12(13),X'FF'
000046 07FF                        22          BCR   15,14
000048 47FF 000A           0000A   23 IBYTE    BC    15,10(15)
00004C 05                          24          DC    X'5'
00004D C9C2E8E3C5                  25          DC    CL5'IBYTE'
000052 90EC D00C           0000C   26          STM   14,4,12(13)
000056 0520                        27          BALR  2,0
000058                             28          USING *,2
000058 1B00                        29          SR    0,0
00005A 9834 1000           00000   30          LM    3,4,0(1)
00005E 5844 0000           00000   31          L     4,0(4)
000062 5840 2050           000A8   32          S     4,=F'1'
000066 4304 3000           00000   33          IC    0,0(4,3)
00006A 9824 001C           0001C   34          LM    2,4,28(13)
00006E 92FF D00C      0000C        35          MVI   12(13),X'FF'
000072 07FF                        36          BCR   15,14
000074 47FF 000A           0000A   37 LJABF    BC    15,10(15)
000078 05                          38          DC    X'5'
000079 D3D1C1C2C6                  39          DC    CL5'LJABF'
00007E 90EC D00C           0000C   40          STM   14,4,12(13)
000082 0520                        41          BALR  2,0
000084                             42          USING *,2
000084 9834 1000           00000   43          LM    3,4,0(1)
000088 5844 0000           C0000   44          L     4,0(4)
00008C 5840 2024           000A8   45          S     4,=F'1'
000090 4304 3000           00000   46          IC    0,0(4,3)
000094 8900 0018           00018   47          SLL   0,24
000098 5600 202A           000AC   48          O     0,=X'00404040'
00009C 9814 0018           00018   49          LM    1,4,24(13)
0000A0 92FF D00C      0000C        50          MVI   12(13),X'FF'
0000A4 07FF                        51          BCR   15,14
                                   52          END
0000A8 00000001                    53                =F'1'
0000AC 00404040                    54                =X'00404040'
```

```
FORTRAN IV G LEVEL  20                SEARCH          DATE = 73054      19/05/53          PAGE 0001

0001                  SUBROUTINE SEARCH (ITEC,NN)
0002                  DIMENSION ITEC(2)
0003                  COMMON/DICT/NDR,IDICT(2,400)
0004                  LPOINT=0
0005                  INDX=NDR+1
0006                1 INDX=(INDX+1)/2
0007                  KK=LPOINT + INDX
0008                  IF(KK.GT.NDR)GO TO 3
0009                  ITEMP=LJABF(IDICT(1,KK),1)
0010                  IF(ITEMP-ITEC(1))2,5,3
0011                5 ITEMP=LJABF(IDICT(1,KK),2)
0012                  IF(ITEMP-ITEC(2))2,6,3
0013                2 LPOINT=KK
0014                3 IF(INDX-1)4,4,1
0015                4 NN=0
0016                  RETURN
0017                6 NN=KK
0018                  RETURN
0019                  END
```

```
0001          SUBROUTINE CVADJ
0002          REAL*8 CVA(7),DOPCS( 7,5),GAR( 7,5),GART(2,8)
0003          REAL*8 TABLE(2,8,7),CARD(2,7)
0004          REAL*8 ZMANH,GARATE
0005          DO 5 L=1,5
0006        5 READ (2,21) (GAR(J,L),J=1,7)
0007       21 FORMAT (7F11.4)
0008          DO 40 J=1,2
0009       40 READ(5,41) (CARD(J,K),K=1,7)
0010          DO 43 L=1,7
0011          DO 43 J=1,2
0012       43 READ (5,41) (TABLE(J,K,L),K=1,8)
0013       41 FORMAT (8F10.0)
0014          RETURN
0015          ENTRY CVADJ1 (JJJ,LLL,DOPCS,CVA)
0016          J=JJJ
0017          L=LLL
0018          IF (CARD(1,J).LT.TABLE(1,1,J)) CARD (1,J)=TABLE(1,1,J)
0019          IF (CARD(1,J).GT.TABLE(1,8,J)) CARD(1,J)=TABLE(1,8,J)
0020          DO 100 K=1,8
0021          IF (TABLE(1,K,J)-CARD(1,J)) 100,110,120
0022      110 GART(2,1)=CARD(2,J)-TABLE(2,K,J)
0023          GO TO 130
0024      120 GART(2,1)=CARD(2,J)+
             *          ((TABLE(1,K,J)-CARD(1,J))*
             *          (TABLE(2,K,J)-TABLE(2,K-1,J))/
             *          (TABLE(1,K,J)-TABLE(1,K-1,J)))-
             *          TABLE(2,K,J)
0025          GO TO 130
0026      100 CONTINUE
0027      130 DO 150 K=1,8
0028          GART(1,K)=TABLE(1,K,J)
0029      150 GART(2,K)=GART(2,1)+TABLE(2,K,J)
0030          ZMANH=DOPCS(J,L)/1000.
0031          IF (ZMANH.GE.GART(1,1)) GO TO 170
0032          GARATE=GART(2,1)
0033          GO TO 230
0034      170 IF (ZMANH.LE.GART(1,8)) GO TO 180
0035          GARATE=GART(2,8)
0036          GO TO 230
0037      180 DO 200 K=1,8
0038          IF (ZMANH - GART(1,K)) 220,210,200
0039      210 GARATE=GART(2,K)
0040          GO TO 230
0041      220 GARATE=GART(2,K)-
             *        ((GART(1,K)-ZMANH)*
             *        (GART(2,K)-GART(2,K-1))/
             *        (GART(1,K)-GART(1,K-1)))
0042          GO TO 230
0043      200 CONTINUE
0044      230 CVA(J)=DOPCS(J,L)*(GARATE-GAR(J,L))
0045          RETURN
0046          END
```

```
0001              COMMON/DICT/NDR,IDICT(2,400)
0002              INTEGER JDENT(26)/1HF,1HS,1HT,1HN,1HP,1HM,1HU,1HV,1HW,
                 *1HH,1HL,1HG,1HK,1HI,1H2,1H3,2H3+,1H ,1H ,1H ,1H ,4HNARF,1H ,
                 *4HSHOP,4HCATE,4HGORY/
0003              INTEGER ISHOPX(10)/1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9,1H0/,
                 *INAREX(12)/1HA,1HB,1HC,1HD,1HE,1HF,1HG,1HH,1HI,1HJ,1HK,1HL/,
                 *IPROGX(10)/1HA,1HN,1HF,1HH,1HL,1HP,1HR,1HT,1HV,1HY/,
                 *IFUNDX(8)/1HA,1HC,1HD,1HE,1HI,1HN,1HO,1HU/,
                 *IFY(5)/1H ,1H ,1H ,1H ,1H /
0004              INTEGER JCTU(3)/4HCONT,4HINUE,1HD/,IBLANK/1H /,XXXX/4HXXXX/
0005              INTEGER ITEC(7),ITED(8),ICTU(3),JDICT(4,400 ),IRNN(13),IDATE(5)
0006              REAL*8 DOP(13)/8H      ALA,8H      NOR,8H      N I,8H      Q P,
                 *8H      JAX,8H      C P,8H      PNS,8H    OTHER,8H    XCNUS,
                 *8H     ARMY,8H     USAF,8H     COMM,8H    TOTAL/
0007              REAL*8 PROGN(10)/8HAIRCRAFT,8H  ENGINE,8HACC/COMP,8HELE/COMM,
                 *8HOTH SUPP,8HSPL SUPP,8H O/MAINT,8HARMAMENT,8H  MANF'G,
                 *8HCONT RES/
0008              REAL*8 TOTAL/8H    TOTAL/,AIRCFT/8HAIRCRAFT/,BLANKD/1H /
0009              REAL*8 PROID(13),VALUES(12),FPN(9,11,14),SUMN(9,11,14,5),
                 *CAPB(9,7,5),TOTCAP(10,8,5),CML(9,7),USUM(14),
                 *VSUM(14),WSUM(14),SLACK(9, 7),UBND(9, 7),UBND3(9, 7),
                 *U(9,7),V(9,7),W(9,7),HSUM(14),LSUM(14),
                 *SUBT(12),BSWKLD(9, 7),S23BW(9),S1WKLD(9,7),TWKLD(10,8),
                 *LPWKLD(10,8),PUTIL(10,8),DOPCS( 7,5),DOPGT(12,5),CVA(14),
                 *         BAJ(2,9,7),HSN(9, 7),LSN(9, 7),HCOST(9, 7),
                 *LCOST(9, 7),CML(10,8),TOTL(9,11,14),
                 *GSN(9,12),GCOST(9,12),KSN(9,12),KCOST(9,12)
0010              REAL*8 EF2(5),EF3(5)
0011              REAL*8 FACTOR(5)
0012              REAL*8 FACT3(5)
0013              REAL*8 HPY(7,5),PHPY(5),MEN(10)
0014              REAL*8 SHRNPF(2)
0015              REAL*8 LDOL(9,7),KDOL(9,7),HDOL(9,7),GDOL(9,7),BASE(10,8)
0016              REAL*8 PB(9,200 ),RR(6,100 ),SP(9,7,5,4),RC(9,7,4,4)
0017              REAL*8 NAREN(2,14)/8HALAMEDA ,1H ,8HNORFOLK ,1H ,
                 *8HNORTH IS,4HLAND,8HQUONSET ,5HPOINT,8HJACKSONV,4HILLE,
                 *8HCHERRY P,4HOINT,8HPENSACOL,1HA ,5HOTHER,1H ,6HXCONUS,1H ,
                 *4HARMY,1H ,4HUSAF,1H ,8HCOMMERCI,2HAL,8HALL RWK ,8HFACILITY,
                 *8HALL NARF,1HS/
0018              REAL*8 SHOPN(9)/8HAIRFRAME,6HENGINE,8HACC/COMP,8HELE/COMM,
                 *8HARMAMENT,8HSUPP/EQU,8HMANF/REP,8HTEST/CAL,5HOTHER/
0019              REAL*8 FUNDN(9)/8HO+M A/C ,8HARM/CAL ,8HOTH O+M ,3HPAM,
                 *8HSTK FUND,4HUSAF,4HARMY,5HOTHER,8HALL FUND/
0020              REAL*8 AIIC,TAIIC,ACOST,TCOST,TEMP,XLML,XUML,ANAME,TSUM,
                 *TGPA,TNPA,TMA,TEMQ,TEMR
0021              DIMENSION IYEARX(5)
0022              EQUIVALENCE (ISHOPX(1),IYEARX(1))
C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
0023              LUQ=1
0024              LUU=2
0025              IFILE=4
0026              READ 5,IPQQ,IRNN,IDATE
0027            5 FORMAT (A1,9X,12A4,A2,5A4)
0028              N=1
0029            3 READ(LUQ,1,END=2)(IDICT(L,N),L=1,2),(JDICT(L,N),L=1,4)
```

```
0030          1 FORMAT(6A4)
0031            N=N+1
0032            GO TO 3
0033          2 DO 4 J=1,2
0034          4 IDICT(J,N)=XXXX
0035            DO 8 J=1,4
0036          8 JDICT(J,N)=XXXX
0037            NDR=N-1
0038            NFACIL=12
0039            NNARFS=7
0040            NSHOPS=9
0041            NPROGS=10
0042            NFUNDS=8
0043            NYEARS=5
0044            DO 7 K=1,NYEARS
0045            DO 7 N=1,14
0046            DO 7 M=1,11
0047            DO 7 L=1,9
0048          7 SUMN(L,M,N,K)=0.
0049            DO 25 J=1,10
0050            DO 25 L=1,8
0051            DO 25 K=1,NYEARS
0052         25 TOTCAP(J,L,K)=0.
0053            READ(LUU,13)IYA,IYB,IBOP,INOS,XLML,XUML,IAY
0054         13 FORMAT(4I5,2F6.3,I5)
0055            READ (LUU,14) FF2,FF3
0056            DO 29 J=1,NYEARS
0057         29 READ (LUU,14) (HPY(L,J),L=1,NNARFS),PHPY(J)
0058            DO 15 M=IYA,IYB
0059            DO 15 L=1,NFUNDS
0060            DO 15 J=1,NPROGS
0061         15 READ (LUU,14)(SUMN(L,J,K,M),K=1,NFACIL)
0062         14 FORMAT(12F11.0)
0063            DO 17 K=1,NSHOPS
0064            DO 17 J=1,NYEARS
0065         17 READ(LUU,14)(CAPB(K,L,J),L=1,NNARFS)
0066            DO 19 K=1,NSHOPS
0067            DO 19 J=1,NYEARS
0068         19 READ(LUU,14)(TOTCAP(K,L,J),L=1,NNARFS)
0069            DO 20 J=1,NSHOPS
0070         20 READ(LUU,14)(CML(J,K),K=1,NNARFS)
0071            CALL CVADJ
0072            DO 18 K=1,NNARFS
0073            DO 18 J=1,NSHOPS
0074         18 CML(J,K)=CML(J,K)/HPY(K,IYA)
0075            IF (INOS .EQ. 2) GO TO 11
0076            DO 10 K=1,NNARFS
0077            DO 10 J=2,NSHOPS
0078         10 CML(1,K)=CML(1,K)+CML(J,K)
0079         11 IAY=IAY-IYA
0080            DO 16 J=1,NYEARS
0081            IAY=IAY+1
0082            ITEMP=IAY/10
0083            IF (ITEMP.EQ.0) ITEMP=10
0084            ITEMQ=MOD(IAY,10)
```

```
0085              IF (ITEMQ.EQ.0) ITEMQ=10
0086              CALL MOVE (IYEARX(ITEMP),1,IFY(J),1,1)
0087              CALL MOVE (IYEARX(ITEMQ),1,IFY(J),2,1)
0088           16 CONTINUE
0089              DO 12 J=IYA,IYB
0090              FACTOR(J)=1./EF2(J)-1.
0091           12 FACT3(J)=1./EF3(J)-1.
                C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
0092              DO 999 K=IYA,IYB
                C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
0093              DO 6 L=1,14
0094              DO 6 J=1,11
0095              DO 6 I=1,9
0096              TOTL(I,J,L)=0.
0097            6 FPN(I,J,L)=0.
0098              DO 9 L=1,14
0099              CVA(L)=0.
0100              HSUM(L)=0.
0101              LSUM(L)=0.
0102              USUM(L)=0.
0103              VSUM(L)=0.
0104            9 WSUM(L)=0.
0105              DO 27 J=1,10
0106              MEN(J)=0.
0107              DO 27 L=1,8
0108              TWKLD(J,L)=0.
0109              PUTIL(J,L)=0.
0110              LPWKLD(J,L)=0.
0111              BASE(J,L)=0.
0112           27 DML(J,L)=0.
0113              IEND2=0
0114              NROWS=0
0115              NCOLS=0
0116              NOP=0
                C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
                C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
0117           21 CALL POSITN(IFILE,INDIC,2)
0118              CALL ARRAY(IFILE,INDIC,ANAME)
0119              IF(IEND2.EQ.1)GO TO 42
0120              NR=0
0121              NN=1
0122              NS=1
0123           22 CALL VECTOR (IFILE,INDIC,VALUES)
0124              DO 23 L=1,4
0125           23 ITEC(L)=LJABF(VALUES(1),L)
0126              IF(ITEC(4).NE.IYEARX(K))GO TO 22
0127              DO 26 L=4,6
0128              IF (ITEC(1).EQ.JDENT(L)) GO TO 22
0129           26 CONTINUE
0130              NR=NR+1
0131              GO TO (24,28,32),NR
0132           24 SLACK(NS,NN)=VALUES(4)
0133              GO TO 22
0134           28 UBND(NS,NN)=VALUES(6)
0135              GO TO 22
```

```
0136          32 UBND3(NS,NN)=VALUES(6)
0137          33 NP=0
0138             NS=NS+1
0139             IF(NS.LE.NSHOPS)GO TO 22
0140             NS=1
0141             NN=NN+1
0142             IF(NN.LE.NNAREAS)GO TO 22
0143             NN=1
0144             IEND2=1
0145             M=378*(IYB-K)
0146             IF (INOS.EQ.1) M=210*(IYB-K)
0147             IF (M.EQ.0) GO TO 37
0148             DO 36 J=1,M
0149             CALL VECTOR (IFILE,INDIC,VALUES)
0150          36 CONTINUE
0151          37 CALL VECTOR(IFILE,INDIC,VALUES)
0152             IF (INDIC.EQ.1) GO TO 42
0153             ITEMP=LJABF(VALUES(1),4)
0154             IF (ITEMP.NE.IYEARX(K)) GO TO 37
0155             ITEMP=LJABF(VALUES(1),8)
0156             IF (ITEMP.NE.IBLANK) GO TO 37
0157             NROWS=NROWS+1
0158             RB(1,NROWS)=VALUES(1)
0159             RB(2,NROWS)=VALUES(3)
0160             GO TO 37
               C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
0161          42 LCT=22
0162             NOP=NOP+1
0163             IFST=0
0164             IEND=0
0165             DO 43 J=1,3
0166          43 ICTU(J)=IBLANK
0167             CALL APPAY(IFILE,INDIC,ANAME)
0168             IF(NOP.GT.NPROGS)GO TO 90
0169             DO 44 J=1,12
0170          44 PROID(J)=0.
0171          45 CALL VECTOR(IFILE,INDIC,VALUES)
0172             DO 46 J=1,8
0173          46 ITED(J)=LJABF(VALUES(1),J)
0174             IF(ITED(3).NE.ISHOPX(1))GO TO 48
0175             IEND=1
0176             GO TO 62
0177          48 IF(ITED(4).NE.IYEARX(K))GO TO 45
0178             IF (NOP.GT.1) GO TO 52
0179             NCOLS=NCOLS+1
0180             PB(1,NCOLS)=VALUES(1)
0181             DO 51 J=2,5
0182          51 PB(J,NCOLS)=VALUES(J+1)
0183          52 IF(ITED(5).NE.IPROGX(NOP))GO TO 45
0184             DO 54 J=1,NEACIL
0185             IF(ITED(8).EQ.INAREX(J))GO TO 56
0186          54 CONTINUE
0187          56 DO 53 I=1,NFUNDS
0188             IF(ITED(3).EQ.IFUNDX(I))GO TO 58
0189          53 CONTINUE
```

```
0190          58 FPN(I,NOP,J)=FPN(I,NOP,J)+VALUES(3)*VALUES(4)
0191             IF(IFST.NE.0)GO TO 59
0192             IFST=1
0193          47 DO 57 L=1,7
0194          57 ITEC(L)=ITED(L)
0195             GO TO 55
0196          59 DO 60 L=1,7
0197             IF(ITEC(L).NE.ITED(L))GO TO 62
0198          60 CONTINUE
0199          55 PROID(J)=VALUES(3)
0200             GO TO 45
              C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
0201          62 IF(IFST.EQ.0) GO TO 21
0202             IF(LCT.LT.22) GO TO 70
0203             LCT=0
0204             PRINT 69
0205          69 FORMAT(1H1,55X,'WORKLOAD ASSIGNMENT'//)
0206             PRINT 66,IRNN,IDATE
0207          66 FORMAT(10X,'RUN NO./NAME ',12A4,A2,26X,'DATE ',5A4//)
0208             PRINT 68,PROGN(NOP),(ICTU(L),L=1,3),IFY(K)
0209          68 FORMAT(10X,'PROGRAM  ',A8,3X,3A4,57X,'YEAR  19',A2//)
0210             DO 65 L=1,3
0211          65 ICTU(L)=JCTU(L)
0212             DOP(13)=TOTAL
0213             IF(NOP.LT.3.OR. NOP.GT.5)GO TO 74
0214             DOP(13)=AIRCFT
0215             PRINT 63
0216          63 FORMAT(57X,'PERCENT ASSIGNED',/,123X,'SUPPORTED')
0217          74 PRINT 67,DOP
0218          67 FORMAT(1X,'T/M/S',10X,'TEC SP C F',12A8,2X,A8//)
              C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
0219          70 LCT=LCT+1
0220             PROID(13)=0.
0221             DO 71 L=1,NFACIL
0222          71 PROID(13)=PROID(13)+PROID(L)
0223             IF (NOP.LT.3.OR. NOP.GT.5) GO TO 64
0224             DO 61 L=1,NFACIL
0225          61 PROID(L) = PROID(L)/PRCID(13)*100.
0226             GO TO 82
0227          64 DO 72 L=1,13
0228             ITEMP=PROID(L)+0.5
0229             TEMP=ITEMP
0230             IF (DABS(PROID(L) - TEMP).LT.0.1) PROID(L)=TEMP
0231          72 CONTINUE
0232          82 CALL SEARCH(ITEC,NN)
0233             IF(NN.EQ.0)NN=NDR+1
0234             PRINT 73,(JDICT(L,NN),L=1,4),IDICT(2,NN),(ITEC(L),L=6,7),ITEC(3),
                *PROID
0235          73 FORMAT (1X,3A4,A3,A4,1X,A1,1X,A1,1X,A1,12F8.1,F10.0/)
0236             IF(IEND.EQ.1)GO TO 21
0237             DO 83 L=1,NFACIL
0238          83 PRCID(L)=0.
0239             GO TO 47
              C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
              C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
```

```
0240          90 NR=0
0241             NS=1
0242             NN=1
0243             IGOBY=0
0244          92 CALL VECTOR (IFILE,INDIC,VALUES)
0245             IF (IGOBY.EQ.1) GO TO 93
0246             ITEMP=LJABF(VALUES(1),3)
0247             ITEMQ=LJABF(VALUES(1),4)
0248             IF (ITEMP.NE.ISHOPX(1).OR.ITEMQ.NE.IYEARX(K)) GO TO 92
0249             IGOBY=1
0250          93 NR=NR+1
0251             GO TO (95,96,97),NR
0252          95 U(NS,NN)=VALUES(3)
0253             USUM(NN)=USUM(NN)+VALUES(3)*VALUES(4)
0254             GO TO 92
0255          96 V(NS,NN)=VALUES(3)
0256             VSUM(NN)=VSUM(NN)+VALUES(3)*VALUES(4)
0257             GO TO 92
0258          97 NR=0
0259             W(NS,NN)=VALUES(3)
0260             WSUM(NN)=WSUM(NN)+VALUES(3)*VALUES(4)
0261             NS=NS+1
0262             IF (NS.LE.NSHOPS) GO TO 92
0263             NS=1
0264             NN=NN+1
0265             IF (NN.LE.NNARFS) GO TO 92
0266             NN=1
0267             M=1899*(IYR-K)
0268             IF (M.EQ.0) GO TO 125
0269             DO 84 L=1,M
0270             CALL VECTOR (IFILE,INDIC,VALUES)
0271          84 CONTINUE
           C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
0272         125 JS=NSHOPS
0273             IF (INOS.EQ.1) JS=1
0274             IGOBY=0
0275             DO 127 I=1,NNARFS
0276             DO 127 J=1,NSHOPS
0277             DO 127 N=1,2
0278         127 BAJ(N,J,I)=0.
0279          91 DO 98 NS=1,JS
0280             DO 98 I=1,4
0281             CALL VECTOR (IFILE,INDIC,VALUES)
0282             IF (IGOBY.EQ.1) GO TO 123
0283             ITEMP=LJABF(VALUES(1),4)
0284             IF (ITEMP.NE.IYEARX(K)) GO TO 94
0285             IGOBY=1
0286             GO TO 123
0287          94 IF (I.EQ.1.OR.I.EQ.3) GO TO 121
0288             BAJ(1,NS,NN)=BAJ(1,NS,NN)-VALUES(3)
0289             GO TO 98
0290         121 BAJ(1,NS,NN)=BAJ(1,NS,NN)+VALUES(3)
0291             GO TO 98
0292         123 TEMP=VALUES(3)*VALUES(4)
0293             GO TO (85,86,87,88),I
```

```
0294          85 HSUM(NN)=HSUM(NN)+TEMP
0295             HSN(NS,NN)=VALUES(3)/HPY(NN,K)
0296             HCOST(NS,NN)=VALUES(4)*HPY(NN,K)
0297             HDOL(NS,NN)=TEMP
0298             GO TO 98
0299          86 LSUM(NN)=LSUM(NN)+TEMP
0300             LSN(NS,NN)=VALUES(3)/HPY(NN,K)
0301             LCOST(NS,NN)=VALUES(4)*HPY(NN,K)
0302             LDOL(NS,NN)=TEMP
0303             GO TO 98
0304          87 HSUM(NN)=HSUM(NN)+TEMP
0305             GSN(NS,NN)=VALUES(3)/HPY(NN,K)
0306             GCOST(NS,NN)=VALUES(4)*HPY(NN,K)
0307             GDOL(NS,NN)=TEMP
0308             GO TO 98
0309          88 KSN(NS,NN)=VALUES(3)/HPY(NN,K)
0310             KCOST(NS,NN)=VALUES(4)*HPY(NN,K)
0311             LSUM(NN)=LSUM(NN)+TEMP
0312             KDOL(NS,NN)=TEMP
0313          98 CONTINUE
0314             NN=NN+1
0315             IF (NN.LE.NNARES) GO TO 91
0316             NN=1
0317             IF (ITEMP.NE.IYEARX(K)) GO TO 91
               C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
               C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
0318             DO 150 J=1,NNARES
0319             PRINT 102
0320          102 FORMAT (1H1,55X,'WORKLOAD VARIANCE REPORT'/)
0321             PRINT 103
0322          103 FORMAT (64X,'MANHOURS'//)
0323             PRINT 66,IRNN,IDATE
0324             PRINT 104,(NAREN (L,J),L=1,2),IFY(K)
0325          104 FORMAT (10X,'DOP    ',2A9,66X,'YEAR  19',A2,//)
0326             PRINT 105
0327             PRINT 106
0328             PRINT 107
0329          105 FORMAT (23X,'INCREMENT',4X,'L.P.',76X,'POST')
0330          106 FORMAT (1X,'SHOP',11X,'BASE  + BASEWKLD +ASSIGNED =  TOTAL',
                  '1X,2(3X,'SHIFT 1'),2(3X,'SHIFT 2'),3(3X,'SHIFT 3'),
                  '3X,'PERCENT')
0331          107 FORMAT (1X,'CATEGORY',5X,'WORKLOAD SHIFT 2+3',3(2X,'WORKLOAD'),
                  '3(2X,'CAPACITY',2X,'WORKLOAD'),2X,'UTILIZED'//)
               C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
0332             DO 114 I=1,NSHOPS
0333             BSWKLD(I,J)=TOTCAP(I,J,K)-CAPB(I,J,K)
0334             S23BW(I)=0.
0335             IF (CAPB(I,J,K).GE.0.) GO TO 109
0336             IF (BSWKLD(I,J).LT.TOTCAP(I,J,K)+UBND(I,J)*EF2(K)) GO TO 161
0337             S23BW(I)=UBND(I,J)*(1.-EF2(K))
0338             GO TO 162
0339          161 S23BW(I)=(BSWKLD(I,J)-TOTCAP(I,J,K))*FACTOR(K)
0340             GO TO 109
0341          162 IF (BSWKLD(I,J).LT.TOTCAP(I,J,K)+UBND(I,J)*EF2(K)
                  *+UBND3(I,J)*EF3(K)) GO TO 163
```

```
0342            S23BW(I)=S23BW(I)+UBND3(I,J)*(1.-EF3(K))
0343            GO TO 109
0344          163 S23BW(I)=S23BW(I)+(BSWKLD(I,J)-TOTCAP(I,J,K)
                 *-UBND(I,J)*EF2(K))*FACT3(K)
0345          109 S1WKLD(I,J)=TOTCAP(I,J,K)-SLACK(I,J)
0346            TWKLD(I,J)=S1WKLD(I,J)+U(I,J)+V(I,J)+W(I,J)
0347            LPWKLD(I,J)=TWKLD(I,J)-BSWKLD(I,J)-S23BW(I)
0348            IF (TOTCAP(I,J,K).NE.0.) GO TO 113
0349            PUTIL(I,J)=0.
0350            GO TO 114
0351          113 PUTIL(I,J)=TWKLD(I,J)/TOTCAP(I,J,K)*100.
0352          114 CONTINUE
              C * * *
```

```
0391            PRINT 104,(NAREN(L,14),L=1,2),IFY(K)
0392            PRINT 173,(SHOPN(L),L=1,LSHOPS)
0393          173 FORMAT (1X,'NARE',18X,A8,6X,A6,6(4X,A8),11X,'TOTAL'/)
0394            DO 178 L=1,NNARES
0395            DO 178 I=1,LSHOPS
0396            TEMP=TOTCAP(I,L,K)
0397            TEMQ=TWKLD(I,L)
0398            TOTCAP(10,9,K)=TOTCAP(10,8,K)+TEMP
0399            TOTCAP(10,L,K)=TOTCAP(10,L,K)+TEMP
0400            TOTCAP(I,8,K)=TOTCAP(I,8,K)+TEMP
0401            TWKLD(10,8)=TWKLD(10,8)+TEMQ
0402            TWKLD(10,L)=TWKLD(10,L)+TEMQ
0403            TWKLD(I,8)=TWKLD(I,8)+TEMQ
0404          178 CONTINUE
0405            DO 179 L=1,8
0406          179 PUTIL(10,L)=TWKLD(10,L)/TOTCAP(10,L,K)*100.
0407            DO 180 I=1,LSHOPS
0408          180 PUTIL(I,8)=TWKLD(I,8)/TOTCAP(I,8,K)*100.
0409            DO 190 J=1,8
0410            IF (J.EQ.8) GO TO 183
0411            PRINT 182,(NAREN(L,J),L=1,2)
0412          182 FORMAT (1HO,2A8)
0413            GO TO 185
0414          183 PRINT 184
0415          184 FORMAT (1HO,'TOTAL')
0416          185 PRINT 187,(TWKLD(L,J),L=1,LSHOPS),TWKLD(10,J)
0417          187 FORMAT (3X,'TOTAL WORKLOAD',2X,8F12.0,F16.0)
0418            PRINT 188,(TOTCAP(L,J,K),L=1,LSHOPS),TOTCAP(10,J,K)
0419          188 FORMAT (3X,'CAPACITY',8X,8F12.0,F16.0)
0420            PRINT 189,(PUTIL(L,J),L=1,LSHOPS),PUTIL(10,J)
0421          189 FORMAT (3X,'% UTILIZED',6X,8F12.0,F16.0)
0422          190 CONTINUE
0423            PRINT 402
0424          402 FORMAT (1H1,55X,'REQUIRED MANNING LEVEL'/)
0425            PRINT 403
0426          403 FORMAT (60X,'DIRECT LABOR'//)
0427            PRINT 66,IRNN,IDATE
0428            PRINT 104,(NAREN(L,14),L=1,2),IFY(K)
0429            PRINT 405,(SHOPN(L),L=1,NSHOPS)
0430          405 FORMAT (2X,'NARE',13X,A8,5X,A6,6(3X,A8),6X,A5,10X,'TOTAL'/)
0431            DO 407 L=1,NNARES
0432          407 TWKLD(9,8)=TWKLD(9,8)+TWKLD(9,L)
0433            DO 409 L=1,8
0434          409 TWKLD(10,L)=TWKLD(10,L)+TWKLD(9,L)
0435            DO 430 J=1,8
0436            IF (J.EQ.8) GO TO 421
0437            PRINT 182,(NAREN(L,J),L=1,2)
0438            GO TO 422
0439          421 PRINT 184
0440          422 PRINT 424,(TWKLD(L,J),L=1,10)
0441          424 FORMAT (4X,'MANHOURS',4X,9F11.0,F15.0)
0442            IF (J.EQ.8) GO TO 425
0443            DO 426 L=1,10
0444            TWKLD(L,J)=TWKLD(L,J)/HPY(J,K)
0445          426 MEN(J)=MEN(J)+TWKLD(L,J)
```

```
0446              PRINT 427,(TWKLD(L,J),L=1,10)
0447              GO TO 430
0448          425 PRINT 427,(MEN(L),L=1,10)
0449          427 FORMAT (4X,'WORKERS',5X,9F11.0,F15.0//)
0450          430 CONTINUE
                C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
0451              DO 209 N=1,NFACIL
0452              DO 209 NN=13,14
0453              VSUM(NN)=VSUM(NN)+VSUM(N)
0454              USUM(NN)=USUM(NN)+USUM(N)
0455              WSUM(NN)=WSUM(NN)+WSUM(N)
0456              HSUM(NN)=HSUM(NN)+HSUM(N)
0457              LSUM(NN)=LSUM(NN)+LSUM(N)
0458          209 CONTINUE
0459              DO 210 N=1,NFACIL
0460              DO 210 M=1,NPROGS
0461              DO 210 L=1,NFUNDS
0462          210 TOTL(L,M,N)=SUMN(L,M,N,K)+FPN(L,M,N)
0463              NNN=14
0464              DO 220 N=1,NFACIL
0465              IF (N.GT.7) NNN=13
0466              DO 220 M=1,NPROGS
0467              DO 220 L=1,NFUNDS
0468              TEMP=SUMN(L,M,N,K)
0469              TEMQ=FPN(L,M,N)
0470              TEMR=TOTL(L,M,N)
0471              SUMN(9,M,N,K)=SUMN(9,M,N,K)+TEMP
0472              SUMN(L,11,N,K)=SUMN(L,11,N,K)+TEMP
0473              SUMN(9,11,N,K)=SUMN(9,11,N,K)+TEMP
0474              FPN(9,M,N)=FPN(9,M,N)+TEMQ
0475              FPN(L,11,N)=FPN(L,11,N)+TEMQ
0476              FPN(9,11,N)=FPN(9,11,N)+TEMQ
0477              TOTL(9,M,N)=TOTL(9,M,N)+TEMR
0478              TOTL(L,11,N)=TOTL(L,11,N)+TEMR
0479              TOTL(9,11,N)=TOTL(9,11,N)+TEMR
0480              DO 220 NN=13,NNN
0481              SUMN(L,M,NN,K)=SUMN(L,M,NN,K)+TEMP
0482              SUMN(9,M,NN,K)=SUMN(9,M,NN,K)+TEMP
0483              SUMN(L,11,NN,K)=SUMN(L,11,NN,K)+TEMP
0484              SUMN(9,11,NN,K)=SUMN(9,11,NN,K)+TEMP
0485              FPN(L,M,NN)=FPN(L,M,NN)+TEMQ
0486              FPN(9,M,NN)=FPN(9,M,NN)+TEMQ
0487              FPN(L,11,NN)=FPN(L,11,NN)+TEMQ
0488              FPN(9,11,NN)=FPN(9,11,NN)+TEMQ
0489              TOTL(L,M,NN)=TOTL(L,M,NN)+TEMR
0490              TOTL(9,M,NN)=TOTL(9,M,NN)+TEMR
0491              TOTL(L,11,NN)=TOTL(L,11,NN)+TEMR
0492              TOTL(9,11,NN)=TOTL(9,11,NN)+TEMR
0493          220 CONTINUE
                C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
0494              DO 249 J=1,14
0495              IF (J.GT.NFACIL) GO TO 223
0496              PRINT 222
0497          222 FORMAT (1H1,55X,'DOP COST REPORT'//)
0498              GO TO 226
```

```
0499          223 PRINT 224
0500          224 FORMAT (1H1,55X,'PROGRAM COST REPORT'//)
0501          226 PRINT 66,IRNN,IDATE
0502              PRINT 104,(NAREN(L,J),L=1,2),IFY(K)
0503              PRINT 227,PROGN
0504          227 FORMAT (1X,'FUNDS',3X,10(3X,A8),7X,'TOTALS'/)
             C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
0505              DO 235 I=1,9
0506              PRINT 229,FUNDN(I)
0507          229 FORMAT (1X,A8)
0508              PRINT 230,(SUMN(I,L,J,K),L=1,11)
0509          230 FORMAT (4X,'BASE',1X,10F11.0,F13.0)
0510              PRINT 231,(EPN(I,L,J),L=1,11)
0511          231 FORMAT (4X,'L.P.',1X,10F11.0,F13.0)
0512              PRINT 232,(TOTL(I,L,J),L=1,11)
0513          232 FORMAT (4X,'TOTAL',10F11.0,F13.0)
0514          235 CONTINUE
             C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
0515              TSUM=TOTL(9,11,J)
0516              IF (J.GT.NNARFS) GO TO 240
0517              CALL CVADJ1 (J,K,DOPCS,CVA)
0518              CVA(14)=CVA(14)+CVA(J)
0519              CVA(13)=CVA(13)+CVA(J)
0520          240 IF (J.GT.NNARFS.AND.J.LE.NFACIL) GO TO 248
0521              PRINT 241,CVA(J)
0522          241 FORMAT (1H0,'COST/VOLUME ADJUSTMENT',96X,F13.0/)
0523          239 TSUM=TSUM+CVA(J)
0524              PRINT 242,TSUM
0525          242 FORMAT (1X,'SUB TOTAL',109X,F13.0)
0526              PRINT 236,USUM(J)
0527          236 FORMAT (1H0,'INCREMENTAL 2ND SHIFT',97X,F13.0)
0528              PRINT 237,VSUM(J)
0529          237 FORMAT (1X,'INCREMENTAL 3RD SHIFT',97X,F13.0)
0530              PRINT 238,WSUM(J)
0531          238 FORMAT (1X,'INCREMENTAL POST 3RD SHIFT',92X,F13.0/)
0532              PRINT 243,HSUM(J)
0533          243 FORMAT (1X,'INCREASE TO MANNING LEVEL',93X,F13.0)
0534              PRINT 244,LSUM(J)
0535          244 FORMAT (1X,'DECREASE TO MANNING LEVEL',93X,F13.0/)
0536              TSUM=TSUM+HSUM(J)+LSUM(J)+USUM(J)+VSUM(J)+WSUM(J)
0537              PRINT 245,TSUM
0538          245 FORMAT (1X,'GRAND TOTAL',107X,F13.0)
0539          248 IF (J.GT.NFACIL) GO TO 249
0540              DOPGT(J,K)=TSUM
0541          249 CONTINUE
             C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
             C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
0542              DO 251 J=1,NNARFS
0543              DO 251 I=1,NSHOPS
0544              BASE(I,J)=BASE(I,J)/HPY(J,K)
0545              LPWKLD(I,J)=LPWKLD(I,J)/HPY(J,K)
0546          251 BAJ(1,I,J)=BAJ(1,I,J)/HPY(J,K)
0547              DO 253 J=1,NNARFS
0548              DO 253 I=1,NSHOPS
0549              TEMP=BASE(I,J)
```

```
0550            TEMQ=LPWKLD(I,J)
0551            IF (I.GT.1.AND.INOS.EQ.1) GO TO 257
0552            TEMR=CML(I,J)+BAJ(1,I,J)
0553            DML(I,J)=TEMR
0554            DML(I,9)=DML(I,8)+TEMR
0555            DML(10,J)=DML(10,J)+TEMR
0556            DML(10,8)=DML(10,8)+TEMR
0557            BAJ(2,I,J)=CML(I,J)*(1.+XUML)+BAJ(1,I,J)
0558            BAJ(1,I,J)=CML(I,J)*(1.-XLML)+BAJ(1,I,J)
0559          257 BASE(I,9)=BASE(I,8)+TEMP
0560            BASE(10,J)=BASE(10,J)+TEMP
0561            BASE(10,8)=BASE(10,8)+TEMP
0562            LPWKLD(I,9)=LPWKLD(I,8)+TEMQ
0563            LPWKLD(10,J)=LPWKLD(10,J)+TEMQ
0564            LPWKLD(10,8)=LPWKLD(10,8)+TEMQ
0565          253 CONTINUE
0566            IF (INOS.EQ.2) GO TO 261
0567            NXX=1
0568            MXX=NNARES
0569            M=3
0570            GO TO 265
0571          261 NXX=NNARES
0572            MXX=NSHOPS
0573            M=6
0574          265 DO 258 N=18,20
0575          258 JDENT(N)=JDENT(N+M)
0576            SHPNPE(2)=BLANKD
0577            DO 300 L=1,NXX
0578            PRINT 262
0579          262 FORMAT (1H1,55X,'MANNING LEVEL VARIANCE',//,64X,'MEN'//)
0580            PRINT 66,IPNN,IDATE
0581            IF (INOS.EQ.1) GO TO 263
0582            PRINT 807,(NAREN(N,L),N=1,2),IFY(K)
0583            LL=L
0584            LX=L
0585            GO TO 268
0586          263 PRINT 805,IFY(K)
0587            JJ=10
0588            JX=1
0589            LX=8
0590          268 PRINT 264
0591          264 FORMAT (71X,'CURRENT',17X,'TOTAL    MANNING')
0592            PRINT 256,JDENT(18)
0593          256 FORMAT (10X,A4,16X,'BASE',7X,'L.P.',6X,'TOTAL',6X,'LOWER',4X,
                 *'MANNING',6X,'UPPER',4X,'MANNING',3X,'ADJ INCL',5X,'ADJUSTMENT')
0594            PRINT 266,JDENT(19),JDENT(20)
0595          266 FORMAT (10X,2A4,8X,'WORKLOAD + WORKLOAD = WORKLOAD',6X,'BOUND',
                 *6X,'LEVEL',6X,'BOUND',3X,'ADJSTMNT',4X,'IN COST',11X,'COST',///)
0596            TGPA=0.
0597            TNPA=0.
0598            TAIIC=0.
0599            TCOST=0.
0600            DO 295 J=1,MXX
0601            IF (INOS.EQ.2) GO TO 252
0602            LL=J
```

```
0603            DO 254 N=1,2
0604          254 SHPNRF(N)=NAREN(N,LL)
0605            GO TO 259
0606          252 JJ=J
0607            JX=J
0608            SHPNRF(1)=SHOPN(JJ)
0609          259 IF (JJ.EQ.9) GO TO 285
0610            TMA=TWKLD(JJ,LL)-DML(JJ,LL)
0611            IF (KSN(JX,LL).GT.0.) GO TO 275
0612            IF (GCOST(JX,LL).EQ.0.) GO TO 274
0613            AIIC=HSN(JX,LL)+GSN(JX,LL)
0614            GO TO 273
0615          274 AIIC=HSN(JX,LL)
0616          273 ACOST=HSN(JX,LL)*HCOST(JX,LL)+GSN(JX,LL)*GCOST(JX,LL)
0617            TAIIC=TAIIC+AIIC
0618            GO TO 280
0619          275 IF (KCOST(JX,LL).EQ.0.) GO TO 276
0620            AIIC=-(LSN(JX,LL)+KSN(JX,LL))
0621            GO TO 278
0622          276 AIIC=-LSN(JX,LL)
0623          278 ACOST=LSN(JX,LL)*LCOST(JX,LL)+KSN(JX,LL)*KCOST(JX,LL)
0624            TAIIC=TAIIC-AIIC
0625          280 TCOST=TCOST+ACOST
0626            TGPA=TGPA+CABS(TMA)
0627            TNPA=TNPA+TMA
0628            PRINT 282,SHPNRF,BASE(JJ,LL),LPWKLD(JJ,LL),TWKLD(JJ,LL),
                *BAJ(1,JX,LL),DML(JJ,LL),BAJ(2,JX,LL),TMA,AIIC,ACOST
0629          282 FORMAT (10X,A8,A5,8F11.1,F15.0//)
0630            GO TO 295
0631          285 PRINT 286,SHPNRF,BASE(9,LL),LPWKLD(9,LL),TWKLD(9,LL)
0632          286 FORMAT (10X,A8,A5,3F11.1,5(7X,'N.A.'),11X,'N.A.'//)
0633          295 CONTINUE
0634            PRINT 296,BASE(10,LX),LPWKLD(10,LX),TWKLD(10,LX),DML(10,LX),
                *TAIIC,TCOST
0635          296 FORMAT (10X,'TOTAL',8X,3F11.1,11X,F11.1,22X,F11.1,F15.0//)
0636            PRINT 298,TNPA
0637          298 FORMAT (1HO,55X,'NET PERSONNEL ADJUSTMENT',9X,F11.1)
0638            PRINT 297,TGPA
0639          297 FORMAT (//54X,'TOTAL PERSONNEL ADJUSTMENT',9X,F11.1)
0640          300 CONTINUE
0641            IF (IRQQ.EQ.IBLANK) GO TO 999
0642            NARY=4
0643            DO 550 IQR=1,2
0644            NARY=NARY+IQR
0645            IROW=0
0646            CALL POSITN(IFILE,INDIC,NARY)
0647            CALL ARRAY (IFILE,INDIC,ANAME)
0648          513 CALL VECTOR (IFILE,INDIC,VALUES)
0649            IF (INDIC.EQ.1) GO TO 550
0650            ITEMQ=LJABF(VALUES(1),3)
0651            IF (ITEMQ.LT.ISHOPX(10)) GO TO 543
0652            ITEMP=LJABF(VALUES(1),4)
0653            IF (ITEMP.NE.IYEARX(K)) GO TO 513
0654            ITEMP=LJABF(VALUES(1),1)
0655            DO 525 L=1,5
```

```
0656            IF (ITEMP.EQ.JDENT(L)) GO TO 531
0657          525 CONTINUE
0658            GO TO 513
0659          531 DO 517 I=1,10
0660            IF (ITEMQ.EQ.ISHOPX(I)) GO TO 518
0661          517 CONTINUE
0662          518 IF (I.EQ.10) I=1
0663            ITEMP=LJABF(VALUES(1),2)
0664            DO 522 J=1,NNARFS
0665            IF (ITEMP.EQ.INAREX(J)) GO TO 523
0666          522 CONTINUE
0667          523 M=1
0668            DO 528 N=1,3,2
0669            SP(I,J,L,N)=VALUES(N+M)
0670            SP(I,J,L,N+1)=VALUES(N+M+1)
0671            M=3
0672          528 CONTINUE
0673            GO TO 513
0674          537 CALL VECTOR (IFILE,INDIC,VALUES)
0675          540 IF (INDIC.EQ.1) GO TO 550
0676          543 ITEMP=LJABF(VALUES(1),4)
0677            IF (ITEMP.NE.IYEARX(K)) GO TO 537
0678          541 IROW=IROW+1
0679            IF (RR(1,IROW).NE.VALUES(1)) GO TO 541
0680            DO 547 J=3,4
0681          547 RR(J,IROW)=VALUES(J-1)
0682            DO 548 J=5,6
0683          548 RR(J,IROW)=VALUES(J+1)
0684            GO TO 537
0685          550 CONTINUE
0686          560 NARY=5
0687            DO 630 IQC=1,2
0688            ICOL=0
0689            NARY=NARY+IQC
0690            CALL POSITN(IFILE,INDIC,NARY)
0691            CALL ARRAY (IFILE,INDIC,ANAME)
0692          573 CALL VECTOR(IFILE,INDIC,VALUES)
0693            IF (INDIC.EQ.1) GO TO 630
0694            ITEMQ=LJABF(VALUES(1),3)
0695            IF (ITEMQ.GE.ISHOPX(10)) GO TO 601
0696            ITEMP=LJABF(VALUES(1),4)
0697            IF (ITEMP.NE.IYEARX(K)) GO TO 573
0698          582 ICOL=ICOL+1
0699            IF (PP(1,ICOL).NE.VALUES(1)) GO TO 582
0700            DO 584 J=6,7
0701          584   PB(J,ICOL)=VALUES(J-4)
0702            DO 585 J=8,9
0703          585   PB(J,ICOL)=VALUES(J-1)
0704            GO TO 573
0705          579 CALL VECTOR(IFILE,INDIC,VALUES)
0706          600 IF (INDIC.EQ.1) GO TO 630
0707          601 ITEMP=LJABF(VALUES(1),4)
0708            IF (ITEMP.NE.IYEARX(K)) GO TO 579
0709            ITEMQ=LJABF(VALUES(1),3)
0710            ITEMP=LJABF(VALUES(1),1)
```

```
0711            DO 615 L=1,4
0712            IF(ITEMP.EQ.JDENT(L+9)) GO TO 621
0713          615 CONTINUE
0714            GO TO 579
0715          621 DO 607 I=1,10
0716            IF (ITEMQ.EQ.ISHOPX(I)) GO TO 608
0717          607 CONTINUE
0719          608 IF (I.EQ.10) I=1
0719            ITEMP=LJABF(VALUES(1),2)
0720            DO 612 J=1,NNARFS
0721            IF (ITEMP.EQ.INARFX(J)) GO TO 613
0722          612 CONTINUE
0723          613 M=1
0724            DO 618 N=1,3,2
0725            RC(I,J,L,N)=VALUES(N+M)
0726            RC(I,J,L,N+1)=VALUES(N+M+1)
0727            M=4
0728          618 CONTINUE
0729            GO TO 579
0730          630 CONTINUE
0731          650 DO 680 MM=1,2
0732            NXX=NCOLS
0733            IF(MM.EQ.2)NXX=NROWS
0734            DO 680 IJ=1,NPROGS
0735            LCT=22
0736            DO 651 N=1,3
0737          651 ICTU(N)=IBLANK
0738            DO 680 I=1,NXX
0739            DO 654 N=1,8
0740            IF(MM.EQ.2)GO TO 655
0741            ITED(N)=LJABF(PR(1,I),N)
0742            GO TO 654
0743          655 ITED(N)=LJABF(RR(1,I),N)
0744          654 CONTINUE
0745            IF(ITED(5).NE.IPROGX(IJ))GO TO 680
0746            IF(LCT.LT.22)GO TO 670
0747            LCT=0
0748            IF(MM.EQ.2)GO TO 656
0749            PRINT 652
0750          652 FORMAT(1H1,56X,'PRODUCTION BOUNDS')
0751            PRINT 653
0752          653 FORMAT(1HO,59X,'REDUCED COST'/)
0753            GO TO 659
0754          656 PRINT 657
0755          657 FORMAT(1H1,55X,'REWORK REQUIREMENT')
0756            PRINT 658
0757          658 FORMAT(1HO,58X,'SHADOW PRICES'/)
0758          659 PRINT 66,IPNN,IDATE
0759            PRINT 68, PROGN(IJ),ICTU,IFY(K)
0760            IF(MM.EQ.2)GO TO 662
0761            PRINT 660
0762          660 FORMAT(43X,'QUANTITY',7X,'UNIT',6X,'LOWER',6X,'UPPER',3X,'LOWER RC
                 ''  ,6X,'LOWER',3X,'UPPER RC',6X,'UPPER')
0763            PRINT 661
0764          661 FORMAT(4X,'T/M/S',11X,'TEC  F S C     NARF',4X,'ASSIGNED',
```

```
     *7X,'COST',
     '2(6X,'P. B.'),2(6X,'LIMIT',6X,'R. C.')//)
0765      GO TO 665
0766  662 PRINT 663
0767  663 FORMAT(52X,'REWORK',8X,'LOWER SP',11X,'LOWER',8X,'UPPER SP',
     '11X,'UPPER')
0769      PRINT 664
0769  664 FORMAT(10X,'T/M/S',11X,'TEC  F S C',11X,'REQUIREMENT',11X,
     ''LIMIT',11X,'S. P.',11X,'LIMIT',11X,'S. P.'//)
0770  665 DO 666 N=1,3
0771  666 ICTU(N)=JCTU(N)
0772  670 LCT=LCT+1
0773      IF (MM.EQ.2) GO TO 689
0774      DO 683 J=1,NFACIL
0775      IF (ITED(8).EQ.INAREX(J)) GO TO 688
0776  683 CONTINUE
0777  688 DO 687 N=1,7
0778      IF (ITEC(N).NE.ITEC(N)) GO TO 689
0779  687 CONTINUE
0780      IF (LCT.NE.1) GO TO 675
0781  689 DO 671 N=1,7
0782  671 ITEC(N)=ITED(N)
0783      CALL SEARCH(ITEC,NN)
0784      IF(NN.EQ.0)NN=NDR+1
0785      IF(MM.EQ.2)GO TO 673
0786      PRINT 672,(JDICT(N,NN),N=1,4),IDICT(2,NN),ITED(3),(ITED(N),N=6,7),
     *DOP(J),(PB(N,I),N=2,9)
0788  672 FORMAT(4X,3A4,A3,1X,A4,3(1X,A1),1X,A8,1X,4F11.0,2(F11.0,F11.2)/)
0788      GO TO 680
0789  673 PRINT 674,(JDICT(N,NN),N=1,4),IDICT(2,NN),ITED(3),(ITED(N),N=6,7),
     *(PB(N,I),N=2,6)
0790  674 FORMAT(10X,3A4,A3,1X,A4,3(1X,A1),6X,F16.0,2(F16.0,F16.2)/)
0791      GO TO 680
0792  675 PRINT 676,ITED(3),(ITED(N),N=6,7),DOP(J),(PB(N,I),N=2,9)
0793  676 FORMAT(24X,3(1X,A1),1X,A8,1X,4F11.0,2(F11.0,F11.2)/)
0794  680 CONTINUE
0795  700 DO 715 L=1,NNARFS
0796      PRINT 701
0797  701 FORMAT(1H1,52X,'SHOP CATEGORY CONSTRAINTS')
0798      PRINT 658
0799      PRINT 66,IRNN,IDATE
0800      PRINT 703,(NAREN(N,L),N=1,2),IFY(K)
0801  703 FORMAT(10X,'NARE   ',2A8,66X,'YEAR  19',A2,//)
0802      PRINT 704
0803  704 FORMAT(10X,'SHOP',52X,'LOWER SP',11X,'LOWER',8X,'UPPER SP',11X,
     ''UPPER')
0804      PRINT 705
0805  705 FORMAT(10X,'CATEGORY',4X,'SHIFT',7X,'WORKLOAD',8X,'CAPACITY',
     '2(11X,'LIMIT',11X,'S. P.'))
0806      DO 715 J=1,LSHOPS
0807      PRINT 711,SHOPN(J),JDENT(14),SIWKLD(J,L),TOTCAP(J,L,K),
     *(SP(J,L,1,N),N=1,4)
0809  711 FORMAT(1H0,9X,A8,6X,A2,2F16.0,2(F16.0,F16.2))
0809      PRINT 712,JDENT(15),U(J,L),UBND(J,L),(SP(J,L,2,N),N=1,4)
0810  712 FORMAT(24X,A2,2F16.0,2(F16.0,F16.2))
```

```
0811            PRINT 712,JDENT(16),V(J,L),UBND3(J,L),(SP(J,L,3,N),N=1,4)
0812            PRINT 713,JDENT(17),W(J,L)
0813          713 FORMAT(24X,A2,F16.0,5(12X,'N.A.'))
0814          715 CONTINUE
0815          750 IF(INDS.EQ.2)GO TO 756
0816            NXX=1
0817            MXX=NNARFS
0818            DO 753 N=18,20
0819          753 JDENT(N)=JDENT(N+3)
0820            GO TO 761
0821          756 NXX=NNARFS
0822            MXX=NSHOPS
0823            SHPNRF(2)=BLANKD
0824            DO 758 N=18,20
0825          758 JDENT(N)=JDENT(N+6)
0826          761 DO 795 L=1,NXX
0827            PRINT 763
0828          763 FORMAT(1H1,56X,'MANPOWER VARIANCE')
0829            PRINT 653
0830            PRINT 66,IRNN,IDATE
0831            IF (INDS.EQ.2) GO TO 766
0832            PRINT 805,IFY(K)
0833            GO TO 769
0834          766 PRINT 807,(NARFN(N,L),N=1,2),IFY(K)
0835          769 PRINT 767,JDENT(18)
0836          767 FORMAT(10X,A4,32X,'UNIT',19X,'TOTAL',4X,'LOWER RC',7X,
                 *'LOWER',4X,'UPPER RC',7X,'UPPER')
0837            PRINT 769,JDENT(19),JDENT(20)
0838          768 FORMAT(10X,2A4,11X,'VARIABLE',9X,'COST',4X,'MANHOURS',8X,'COST',
                 *2(7X,'LIMIT',7X,'R. C.')/)
0839            TCOST=0.
0840            DO 790 J=1,MXX
0841            IF(INDS.EQ.2)GO TO 773
0842            JJ=1
0843            LL=J
0844            DO 774 N=1,2
0845          774 SHPNRF(N)=NARFN(N,LL)
0846            GO TO 775
0847          773 JJ=J
0848            LL=L
0849            SHPNRF(1)=SHOPN(JJ)
0850          775 IF(KSN(JJ,LL).NE.0.)GO TO 780
0851            PRINT 777,SHPNRF,GCOST(JJ,LL),GSN(JJ,LL),GDOL(JJ,LL),
                *(RC(JJ,LL,3,N),N=1,4)
0852          777 FORMAT(10X,2A8,5X,'HIRE 1 ',F12.2,2F12.0,2(F12.0,F12.2))
0853            PRINT 778,HCOST(JJ,LL),HSN(JJ,LL),HDOL(JJ,LL),(RC(JJ,LL,1,N),
                *N=1,4)
0854          778 FORMAT(31X,'HIRE 2 ',F12.2,2F12.0,2(F12.0,F12.2)/)
0855            TCOST=TCOST+GDOL(JJ,LL)+HDOL(JJ,LL)
0856            GO TO 790
0857          780 PRINT 781,SHPNRF,KCOST(JJ,LL),KSN(JJ,LL),KDOL(JJ,LL),
                *(RC(JJ,LL,4,N),N=1,4)
0858          781 FORMAT(10X,2A8,3X,'LAYOFF 1 ',F12.2,2F12.0,2(F12.0,F12.2))
0859            PRINT 782,LCOST(JJ,LL),LSN(JJ,LL),LDOL(JJ,LL),(RC(JJ,LL,2,N),
                *N=1,4)
```

```
0860          782 FORMAT(29X,'LAYOFF 2 ',F12.2,2F12.0,2(F12.0,F12.2)/)
0861              TCOST=TCOST+KDOL(JJ,LL)+LDOL(JJ,LL)
0862          790 CONTINUE
0863              PRINT 791,TCOST
0864          791 FORMAT(//1HO,49X,'TOTAL COST  ',F12.0)
0865          795 CONTINUE
0866          800 DO 840 L=1,NXX
0867              PRINT 803
0868          803 FORMAT(1H1,52X,'MANNING LEVEL CONSTRAINTS')
0869              PRINT 658
0870              PRINT 66,IRNN,IDATE
0871              IF(INOS.EQ.2)GO TO 806
0872              PRINT 805,IFY(K)
0873          805 FORMAT(10X,'ALL NAREAS',80X,'YEAR  19',A2//)
0874              GO TO 810
0875          806 PRINT 807,(NAREN(N,L),N=1,2),IFY(K)
0876          807 FORMAT(10X,'NAREA',2X,2A8,67X,'YEAR  19',A2,//)
0877          810 PRINT 811,JDENT(18)
0878          811 FORMAT(10X,A4,33X,'LOWER BOUND',8X,'LOWER SP',11X,'LOWER',
                 '8X,'UPPER SP',11X,'UPPER')
0879              PRINT 812,JDENT(19),JDENT(20)
0880          812 FORMAT(10X,2A4,16X,'MANHOURS',5X,'UPPER BOUND',
                '2(11X,'LIMIT',11X,'S. P.')/)
0881              DO 840 J=1,MXX
0882              IF(INOS.EQ.2)GO TO 817
0883              JJ=1
0884              LL=J
0885              DO 815 N=1,2
0886          815 SHPNRE(N)=NAREN(N,LL)
0887              DO 816 N=2,NSHOPS
0888          816 TWKLD(1,LL)=TWKLD(1,LL)+TWKLD(N,LL)
0889              GO TO 820
0890          817 JJ=J
0891              LL=L
0892              SHPNRE(1)=SHOPN(JJ)
0893          820 PRINT 821,SHPNRE,TWKLD(JJ,LL),BAJ(1,JJ,LL),(SP(JJ,LL,4,N),N=1,4)
0894          821 FORMAT(1HO,9X,2A8,2F16.0,2(F16.0,F16.2))
0895              PRINT 822,BAJ(2,JJ,LL),(SP(JJ,LL,5,N),N=1,4)
0896          822 FORMAT(42X,2F16.0,2(F16.0,F16.2))
0897          840 CONTINUE
0898          999 CONTINUE
             C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
             C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
0899              PRINT 361
0900          361 FORMAT (1H1,48X,'DOP WORKLOAD - COST SUMMARY REPORT',//)
0901              PRINT 66,IRNN,IDATE
0902              PRINT 363,(DOP(L),L=1,7)
0903          363 FORMAT(21X,7(A8,6X),7X,'TOTALS')
0904              PRINT 364,(DOP(L),L=8,12)
0905          364 FORMAT (28X,5(A8,6X),2X,'STOTAL'//)
0906              DO 390 K=IYA,IYB
0907              DO 366 I=1,4
0908          366 SUBT(I)=0.
0909              DO 367 I=1,NFACIL
0910              IF (I.GT.NNARFS) GO TO 365
```

```
0911              SUBT(1)=SUBT(1)+DOPCS(I,K)
0912              SUBT(2)=SUBT(2)+DOPGT(I,K)
0913              GO TO 367
0914          365 SUBT(3)=SUBT(3)+DOPGT(I,K)
0915          367 CONTINUE
0916              SUBT(4)=SUBT(2)+SUBT(3)
                C * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *  * *
0917              PRINT 370,IFY(K)
0918          370 FORMAT (///,64X,'19',A2//)
0919              PRINT 371,(DOPCS(I,K),I=1,7),SUBT(1)
0920          371 FORMAT (1X,'MANHOURS',6X,7F14.0,3X,F16.0)
0921              PRINT 373,(DOPGT(I,K),I=1,7),SUBT(2)
0922          373 FORMAT (1X,'DOLLARS',7X,7F14.0,3X,F16.0/)
0923              PRINT 372,(DOPGT(I,K),I=8,12),(SUBT(I),I=3,4)
0924          372 FORMAT (1HO,'DOLLARS',14X,6F14.0,10X,F16.0)
0925          390 CONTINUE
0926              STOP
0927              END
```

APPENDIX B

FORMAT OF RECORDS IN INPUT AND DATA BASE FILES

TABLE B-1

MASTER FILE

Card column	Variable	Length
1-4	TEC	4
5	Program	1
6	Subprogram	1
7	Customer	1
8	NARF	1
9-10	Filler	2
11-25	Type Model Series (TMS)	15
25-42	Type	17
43-57	Eng #1 TMS	15
58-61	Filler	4
62-76	Eng #2 TMS	15
77-80	Filler	4
81-84	Total Quantity Service (TQS) Year 1	4
85-87	TQR-AFL Year 1	3
88-91	Total Mission Essential (TME) Year 1	4
92-94	MER-OME Year 1	3
95-98	TQS Year 2	4
99-101	TQR-AFL Year 2	3
102-105	TME Year 2	4
106-108	MER-OME Year 2	3
109-112	TQS Year 3	4
113-115	TQR-AFL Year 3	3
116-119	TME Year 3	4
120-122	MER-OME Year 3	3
123-126	TQS Year 4	4
127-129	TQR-AFL Year 4	3
130-133	TME Year 4	4
134-136	MER-OME Year 4	3
137-140	TQS Year 5	4
141-143	TQR-AFL Year 5	3
144-147	TME Year 5	4
148-150	MER-OME Year 5	3
151-153	Quantity Mission Essential (M/E) Year 1	3
154-158	Norm-Factor Year 1	5
159-161	Quantity Mission Non-Essential (M/N) Year 1	3
162-164	M/E Year 2	3
165-169	Norm-Factor Year 2	5
170-172	M/N Year 2	3
173-175	M/E Year 3	3
176-180	Norm-Factor Year 3	5
181-183	M/N Year 3	3
184-186	M/E Year 4	3
187-191	Norm-Factor Year 4	5
192-194	M/N Year 4	3
195-197	M/E Year 5	3
198-202	Norm-Factor Year 5	5
203-205	M/N Year 5	3
206-300	Filler	95

TABLE B-2

CAPACITY AND DISTRIBUTION FILE

Card column	Variable	Length
	Capacity Record	
1-5	Filler	5
6-7	Fiscal Year	2
8	NARF	1
9	Card Type (=5 for capacity record)	1
10	Filler	1
11-17	Airframe (shop category 1)	7
18-24	Engine (shop category 2)	7
25-31	Accessories and Components (shop category 3)	7
32-38	Elect/Comm/Armament (shop category 4)	7
39-45	Armament (shop category 5)	7
46-52	Support Equipment	7
53-59	Manufacture and Repair (shop category 6)	7
60-66	Test and Calibration (shop category 7)	7
67-74	Covered Area (sq. feet)	8
78-100	Filler	23
	Distribution Record	
1-4	TEC	4
5	Program	1
6	Subprogram	1
7	Filler	1
8	NARF	1
9	Card Type (=4 for distribution record)	1
10	Filler	1
11-13	Shop distribution percent shop 1	3
14-16	Shop distribution percent shop 2	3
17-19	Shop distribution percent shop 3	3
20-22	Shop distribution percent shop 4	3
23-25	Shop distribution percent shop 5	3
26-28	Shop distribution percent shop 6	3
29-31	Shop distribution percent shop 7	3
32-34	Shop distribution percent shop 8	3
35-37	Shop distribution percent other	3
38-100	Filler	63

TABLE B-3

COST RATE FILE

Card column	Variable	Length
1	NARF	1
2	Program	1
3	Subprogram	1
4-7	TEC	4
8	Fund Source	1
9-10	Fiscal Year	2
11	Quarter	1
12-21	TMS	10
22-25	Latest Quarter Total Units	4
26-34	Latest Quarter Total Hours	9
35-38	All Quarters Total Units	4
39-47	All Quarters Total Hours	9
48-50	Filler	3
51-54	Direct Labor Rate (DLR) Year 1	4
55-59	Direct Material Rate (DMR) Year 1	5
60-63	Production Overhead Rate (POR) Year 1	4
64-67	General and Administrative Overhead Rate (GAR) Year 1	4
68-73	Unit Material Rate (UMR) Year 1	6
74-79	GFM Year 1	6
80-83	DLR Year 2	4
84-88	DMR Year 2	5
89-92	POR Year 2	4
93-96	GAR Year 2	4
97-102	UMR Year 2	6
103-108	GFM Year 2	6
109-112	DLR Year 3	4
113-117	DMR Year 3	5
118-121	POR Year 3	4
122-125	GAR Year 3	4
126-131	UMR Year 3	6
132-137	GFM Year 3	6
138-141	DLR Year 4	4
142-146	DMR Year 4	5
147-150	POR Year 4	4
151-154	GAR Year 4	4
155-160	UMR Year 4	6
161-166	GFM Year 4	6
167-170	DLR Year 5	4
171-175	DMR Year 5	5
176-179	POR Year 5	4
180-183	GAR Year 5	4
184-189	UMR Year 5	6
190-195	GFM Year 5	6
196-200	Filler	5

TABLE B-4

DATA BASE FILE

Card column	Variable	Length	
1-2	Data Base Code for TEC	2	
3	Fund Source	1	
4	Year	1	
5	Program	1	
6	Subprogram	1	
7	Customer	1	
8	NARF	1	
9-35	Distribution Factors	27	*(9 three-character fields)*
36-39	Total Quantity in Service (TQS)	4	
40-43	Total Mission Essential (TME)	4	
44-46	Mission Essential (M/E)	3	
47-51	Norm	5	
52-54	Mission Non-Essential (M/N)	3	
55-60	Requirements	6	
61-64	Direct Labor Rate (DLR)	4	
65-69	Direct Material Rate (DMR)	5	
70-73	Production Overhead Rate (POR)	4	
74-78	General and Administrative Rate (GAR)	5	
79-84	Unit Material Rate (UMR)	6	
85-93	Total Cost	9	
94-96	Filler	3	

TABLE B-5

MATRIX ASSIGNING FUND CODE FOR CUSTOMER 1 THROUGH 9 AND A THROUGH I BY SUBPROGRAM AND PROGRAM

Subprogram	Program	Fund code	Subprogram	Program	Fund code
1	A N	A	J	Y	D
2	A N	A	K	A	A
3	A	A	L	N	A
4	F P	A	M	P	E
5	H	A	N	T	E
6	L	A	O	A	A
7	T	C	P	N	A
8	T	C	Q	A	A
9	T	C	R	P	C
#	T	C	[(	V	I
@	A	A	$	V	E
&	A	A	" '	N	I
A	A	A	/	P	I
B	A	A	S	L	A
C	A	A	T	P	E
D	A	A	U	L P	A
E	A	A	V	R	A
F	A	A	W	L	A
G	A N	A	X	R	A
H	A	A	Y	R	A
+	Y	D	L	A	E
–	Y	D			

Unclassified

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)	2a. REPORT SECURITY CLASSIFICATION
Center for Naval Analyses	For Official Use Only
	2b. GROUP

3. REPORT TITLE

Programmer's Guide to the NARF Workload Planning and Budgeting Model

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

5. AUTHOR(S) (First name, middle initial, last name)

Jeffrey B. Birch, Ralph D. Halford

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
August 1973	137	6

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
N00014-68-A-0091	CRC 213
b. PROJECT NO.	
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
d.	---

10. DISTRIBUTION STATEMENT

---

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
---	Office of Naval Research Department of the Navy Washington, D.C. 20350

13. ABSTRACT

This guide presents a detailed description of the computer programs constituting the Naval Aircraft Rework Facility (NARF) Workload Planning and Budgeting Model. As the guide is intended for use by programmers in making detailed changes to program coding, coding receives especial attention in the form of lines-by-lines description of main program listings. A general description of each program, the program listings, and flow charts are included.

The description of the model is contained in the Center for Naval Analyses' INS Study 38, "Naval Aircraft Rework Facility Study." A discussion of the model's uses is contained in CNA Research Contribution 212, the "User's Guide to the NARF Workload Planning and Budgeting Model."

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
NARF (Naval Aircraft Rework Facility) aircraft maintenance computer programs programming linear programming						