TECHNICAL REPORT STANDARD TITLE PAGE

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.

4. Title and Subtitle	5. Report Date
PROGRAM DOCUMENTATION MANUAL FOR THE TEXAS TRIP DISTRIBUTION PACKAGE	November 1971
	6. Performing Organization Code
7. Author(s)	8. Performing Organization Report No.
J. D. Benson, David F. Pearson, Charles E. Bell and Vergil G. Stover	Research Report 167-2
9. Performing Organization Name and Address	10. Work Unit No.
Texas Transportation Institute Texas A&M University College Station, Texas 77843	11. Contract or Grant No. Research Study 2-10-71-167
	13. Type of Report and Period Covered
12. Sponsoring Agency Name and Address	Interim - September 1971 November 1971
Texas Highway Department 11th & Brazos Austin, Texas 78701	14. Sponsoring Agency Code

15. Supplementary Notes

Research performed in cooperation with DOT, FHWA.
Research Study Title: Urban Travel Forecasting

16. Abstract

The Texas Trip Distribution Package is a collection of computer programs designed to perform trip distributions featuring the application of a constrained interactance model. Other programs, available in the package, provide full support. The purpose of this manual is to provide data processing personnel with a link between the Operating Manual for the Texas Trip Distribution Package (Research Report 167-1) and the programs contained in the package. The manual describes the operation of the package and provides flowcharts of the programs in the package. Cross references for significant variables and arrays used in the package and formats for all data sets and data cards associated with the package are provided.

17. Key Words	18. Distribution Statement
Urban Travel Forecasting, Program Documentation Manual, Texas Trip Distribution Package, Computer Flow Charts.	

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified		

Form DOT F 1700.7 (8-69)

PROGRAM DOCUMENTATION MANUAL

for

THE TEXAS TRIP DISTRIBURION PACKAGE

by

J. D. Benson
Assistant Research Planner

David F. Pearson
Research Assistant

Charles E. Bell
Data Processing Programmer

and

Vergil G. Stover
Study Supervisor

Research Report 167-2

Urban Travel Forecasting
Research Study Number 2-10-71-167

Sponsored by the
Texas Highway Department
in cooperation with
U. S. Department of Transportation
Federal Highway Administration

Texas Transportation Institute
Texas A&M University
College Station, Texas
November 1971

(Updated to the status of March, 1974)

DISCLAIMER

The contents of this report reflect the views of the authors who are responsible for the facts and the accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the Federal Highway Administration. This report does not constitute a standard, specification or regulation.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

ABSTRACT

The Texas Trip Distribution Package is a collection of computer programs designed to perform trip distributions featuring the application of a constrained interactance model. Other programs, available in the package, provide full support. The purpose of this manual is to provide data processing personnel with a link between the Operating Manual for the Texas Trip Distribution Package (Research Report 167-1) and the programs contained in the package. The manual describes the operation of the package and provides flowcharts of the programs in the package. Cross references for significant variables and arrays used in the package and formats for all data sets and data cards associated with the package are provided.

SUMMARY

The Texas Trip Distribution Package is a complete collection of computer programs having the capability of performing several different types of trip distributions. The methods range from directionally expanding existing trip matrices to new totals, to performing synthetic distributions using a constrained interactance model.

The basic interactance model applies trip lengths directly in the distribution process and, consequently, needs no calibration. Other properties of the interactance model are similar to a gravity model, without 'F-factors'. By activating a constraint based upon interchange propensity only selected zone pairs enter in to the distribution rather than all possible zone pair combinations as with the gravity model. A sector structure may be imposed to permit a statistical analysis for, and correction of, sector interchange bias created by socio-economic-topographical travel barriers. Movements having external terminals may be processed simultaneously with the synthetic distribution of internal trips.

The Texas Trip Distribution Package is designed to interface with the Texas Small and Large Network Traffic Assignment Packages. It has been prepared for and implemented on IBM 360/50, IBM 360/65, and IBM 370/155 computers. Although it is programmed largely in the FORTRAN IV language for these computers it does take advantage of many of the options available under these operating systems and may, therefore, be somewhat sensitive to peculiarities between installations. For benefit of the user, simplicity and ease of operation have been emphasized in the development

of the package. A number of options are available to the user which provide the flexibility needed for unusual situations.

The package is capable of accommodating up to 4800 zones using a computer having 512,000 bytes of core storage. By making one minor program modification, the capacity can be varied to conform to the amount of core storage available; the minimum amount of core storage that would be required by the package is about 120,000 bytes.

IMPLEMENTATION STATEMENT

The Texas Trip Distribution Package has been operational on the IBM 360 computer installation of the Texas Highway Department since September 1970. It has been used in conjunction with urban studies performed in El Paso, Victoria, Sherman-Denison, Brownsville, and Big Spring.

Several additions, revisions and improvements in the package have been implemented since transmittal of the original version of the program package. Research results from the continuing cooperative research program between the Texas Highway Department and the Texas Transportation Institute will undoubtedly lead to additional refinements. Revisions will be made in this manual as future revisions are implemented in the Texas Trip Distribution Package. The format and binding of this manual are designed to facilitate the inclusion of supplementary pages and the substitution of revised pages as necessary; a revision date will be indicated in the bottom margin of such pages.

INTRODUCTION

The purpose of this manual is to provide data processing personnel with a link between the Operating Manual for the Texas Trip Distribution Package and the programs contained in the package. This manual, therefore, assumes the working knowledge and understanding of the operating manual, and general familiarity with traffic assignment, trip distribution, and computer science.

Organization of Package

The Texas Trip Distribution Package is comprised of 50 control sections (a main program and 49 subprograms). These control sections are listed in Table 1 along with the date of their latest revision. The diagram shown in Figure 1 illustrates the overlay structure in which these control sections operate.

The control sections associated with each of the 24 routines available in the Texas Trip Distribution Package are listed in Table 2. As can be seen from this table, many of the control sections are used by more than one of the routines in the package. In fact, the control sections MAIN, DIRECT, and NAME are used by all the routines.

The Texas Trip Distribution Package basically operates in two phases: the initialization phase and the execution phase. During the initialization phase, the CONTROL cards are read and interpreted by the package and the array which specifies the routines to be executed and their order of execution is initialized. During the execution phase, the routines specified by the CONTROL cards are executed.

Table 1: CONTROL SECTIONS AND THE DATE OF THEIR LATEST REVISION

Control Section	Revision Date	Control Section	Revision Date
ACCEPT	6/18/73	MODEL	7/6/73
ADJUST	6/18/73	NAME	8/4/71
ALTER	9/27/72	ORDER	8/4/71
ASSESS	6/18/73	PACK	8/4/71
BLOCK DATA	7/6/73	PRE	6/18/73
BUILD	6/18/73	PREVUE	8/4/71
CHECK	8/4/71	PRINT	8/4/71
DIRECT	6/18/73	PUNCH	8/4/71
DIVIDE	9/21/71	PUT	6/18/73
DRAW	10/27/72	RANDOM	8/4/71
EDIT	6/18/73	RANK	8/4/71
EQUATE	6/18/73	REFINE	6/18/73
EXPAND	8/4/71	SAVE	6/18/73
FIT (ENTER, REGRET, FORM, REGRES)	8/4/71	SCAN	6/18/73
GRAPH	10/27/72	SCREEN	8/4/71
IMPOSE	6/18/73	SET (GET)	6/18/73
INSERT	8/4/71	SORT	8/4/71
INVOKE	8/4/71	SUM	8/4/71
KEY	8/4/71	SWITCH (MERGE)	8/4/71
LENGTH	8/4/71	TEST	8/4/71
LIST	6/18/73	TIME	8/4/71
MAIN	8/4/71	UNPACK	8/4/71
MATCH	6/18/73	VERIFY	8/4/71
		WRITE	6/18/73

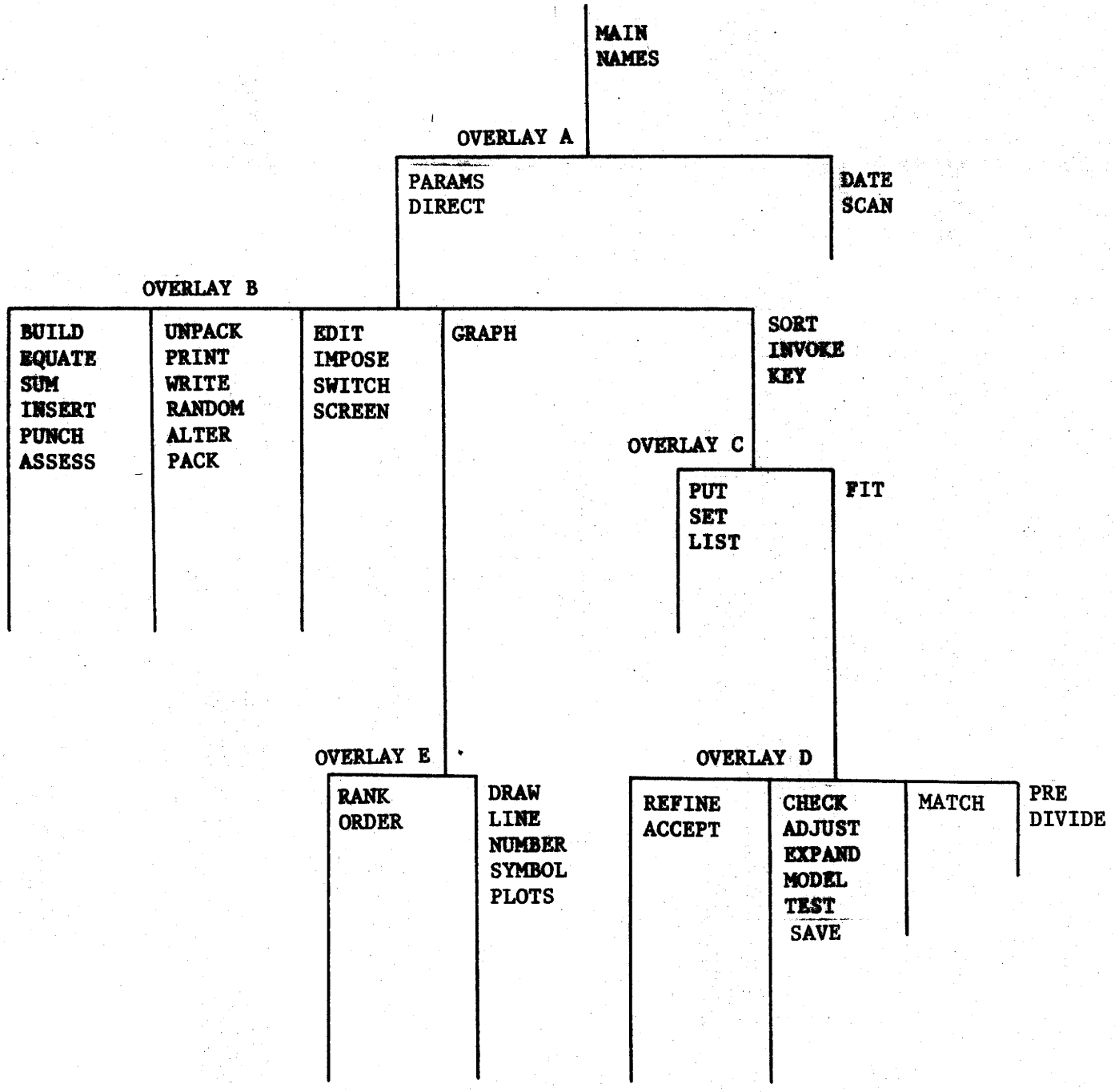


FIGURE 1: OVERLAY STRUCTURE

TABLE 2: THE CONTROL SECTIONS ASSOCIATED WITH EACH OF THE ROUTINES IN THE TEXAS TRIP DISTRIBUTION PACKAGE

ROUTINE	CONTROL SECTIONS
ACCEPT	ACCEPT, DIRECT, INSERT, MAIN, NAME, REFINE, VERIFY
ALTER	ALTER, DIRECT, MAIN, NAME, VERIFY
BUILD	BUILD, DIRECT, MAIN, NAME, VERIFY
EDIT	DIRECT, EDIT, MAIN, NAME, VERIFY
EQUATE	DIRECT, EQUATE, MAIN, NAME, VERIFY
EXPAND	ADJUST, CHECK, DIRECT, ENTER, EXPAND, FIT, FORM, MAIN, NAME
GET	DIRECT, DRAW, EQUATE, GET, GRAPH, MAIN, NAME, PUT, VERIFY
IMPOSE	DIRECT, IMPOSE, MAIN, NAME, VERIFY
LIST	DIRECT, LIST, MAIN, NAME, PUT, VERIFY
MATCH	DIRECT, INVOKE, MAIN, MATCH, NAME, REGRES, SORT, VERIFY
MODEL	ADJUST, CHECK, DIRECT, DIVIDE, ENTER, FIT, FORM, MAIN, MODEL, NAME, PRE, SAVE, TEST
MODIFY	DIRECT, MAIN, NAME, PREVUE
PACK	DIRECT, MAIN, NAME, PACK, VERIFY
PERUSE	DIRECT, MAIN, NAME
PRINT	DIRECT, MAIN, NAME, PRINT, VERIFY
REFINE	ADJUST, CHECK, DIRECT, DRAW, ENTER, FIT, FORM, GRAPH, INSERT, MAIN, NAME, PUNCH, REFINE, TEST
RESTART	ADJUST, CHECK, DIRECT, ENTER, FIT, FORM, MAIN, MODEL, NAME, RESTOR, SAVE, TEST
SCREEN	DIRECT, INVOKE, MAIN, NAME, SCREEN, SORT
SET	DIRECT, EQUATE, MAIN, NAME, PUT, SET, VERIFY
SUM	DIRECT, MAIN, NAME, SUM, VERIFY
SWITCH	DIRECT, INVOKE, MAIN, MERGE, NAME, SORT, SWITCH, VERIFY
UNPACK	DIRECT, MAIN, NAME, UNPACK, VERIFY
WRITE	DIRECT, MAIN, NAME, VERIFY, WRITE

Organization of Manual

Both the operating manual and the programs (with their own internal documentation) are each a form of documentation. As previously mentioned, the objective of this manual is to provide an intermediate level of documentation between the operating manual and the actual program listings. This manual consists of seven sections containing the following information:

- SECTION I - GENERAL OPERATION OF THE PACKAGE: This section describes the operations performed during the initialization phase and the operation of the control programs during the execution phase thereby providing the programmer with an overview of the general operation of the package.

- SECTION II - INDIVIDUAL ROUTINES: This section describes operations performed by each of the 24 routines available in the package. The program calling sequence and a brief description of the operations performed by each of the subprograms is included. For convenience, the information contained in the "Descriptions of Individual Routines" section of the Operating Manual for the Texas Trip Distribution Package (Research Report 167-1) has also been included in this section thereby lessening the need for cross referencing between the manuals. This section should provide the programmer with sufficient information so that he may identify the particular subprogram(s) of interest.

- SECTION III - FLOWCHARTS: This section contains flowcharts for each of the 50 control sections contained in the package. The objective of the flowcharts is to provide the programmer with an overview of the operation of each of the control sections.

- SECTION III - FLOWCHARTS: This section contains flowcharts for each of the 50 control sections contained in the package. The objective of the flowcharts is to provide the programmer with an overview of the operation of each of the control sections. It should be noted that these flowcharts are intended to be used in conjunction with information contained in Sections IV, V, and, possibly, VI when reviewing a particular program listing.

- SECTION IV - SIGNIFICANT VARIABLES AND ARRAYS: This section provides cross references for both the significant variables and significant arrays which are passed between routines. It identifies the routines in which the variable or array may be defined and which routines assume they have been defined and uses the information they contain.

- SECTION V - DATA SET AND DATA CARD FORMATS: The format for each data set and data card used by the package is described in this section.

- SECTION VI - OTHER INFORMATION: This section contains additional general information which is felt to be pertinent to the understanding of some of the routines in the package.

- SECTION VII - RECENT CHANGES AND MODIFICATIONS: This section is provided for information relative to changes which have been implemented since the original documentation and, therefore, serves an "update" function for this manual.

G E N E R A L O P E R A T I O N

INTRODUCTION

INITIALIZATION PHASE

EXECUTION PHASE

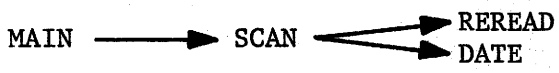
GENERAL OVERVIEW

INTRODUCTION

The Texas Trip Distribution Package basically operates in two phases: the initialization phase and the execution phase. The primary function of the initialization phase is to read and interpret the CONTROL card(s) which identify the routines to be executed. During the execution phase, the specified routines are executed to accomplish the desired task(s). The purpose of this section is, therefore, to describe the program operation of the initialization phase and the operation of the control programs during the execution phase. The functions performed by the individual routines are described in Section II (INDIVIDUAL ROUTINES) of this manual.

INITIALIZATION PHASE

The initialization phase determines the routines to be executed and their order of execution. The execution sequence during this phase is as follows:



The program MAIN simply calls the subroutine SCAN. The subroutine SCAN serves three primary functions:

- It issues the initial call to the system subroutine REREAD. This call is required to initialize the REREAD subroutine thereby allowing the remainder of the subprograms in the package the option of using the REREAD option. This option is used extensively in the package.
- It calls the system subroutine DATE to obtain the date contained in the operating system. This is the date used on the output from the package to specify the date of each run made with the package.
- It then reads and interprets the CONTROL card(s). It also handles any HEADING cards which are encountered while searching for the CONTROL card(s). The first card encountered which is neither a HEADING card nor a CONTROL card terminates the initialization phase and control is returned to program MAIN. Each CONTROL card is interpreted as follows:
  - The card is scanned from left to right for entries

- As each entry is encountered, it is matched against a table of possible valid entries contained in array TABLE (which is defined in the BLOCK DATA subprogram). If a match is found, the integer index pointing to the valid entry in array TABLE is saved in array LIST. If a match is not found, the appropriate error message is printed, an error flag is set, and the scan is continued to identify any other invalid entries.

After the CONTROL cards have been interpreted, array LIST will contain the integer codes which identify the routines to be executed. These integer codes are simply the array index which points to the routine name in the TABLE array. The order in which the integer codes are entered in the LIST array specifies the order the routines are to be executed. It is array LIST which is passed to subroutine DIRECT for use during the execution phase.

If SCAN has encountered an invalid entry in a CONTROL card (i.e., the error flag is set) the job is terminated with a Stop Code 1 rather than control being returned to MAIN for further processing.

EXECUTION PHASE CONTROL PROGRAMS

During the execution phase, the specified routines are executed to accomplish the desired task(s). The two control programs which control the execution phase are MAIN and DIRECT. For each routine to be executed, the program execution sequence is as follows:

After calling SCAN for the initialization phase, the program MAIN simply begins calling DIRECT. It calls DIRECT for each entry in the LIST array (defined in SCAN) and passes to DIRECT the integer code from the LIST array which specifies the routine to be executed. If all the routines specified in the LIST array have been executed and a STOP command was not encountered then the program MAIN terminates the execution of the job with a Stop Code of 10.

The control program DIRECT largely consists of 24 small control sections (one for each of the 24 routines available in the package) which controls the execution of the desired routine. Each control section within DIRECT contains a call to the subroutine NAME which prints the name of the routine being executed. The control program DIRECT basically operates as follows:

- On the first call to DIRECT from MAIN, it issues a call to ASSESS which calculates and prints the package capacities.
- DIRECT then handles any HEADING card(s) which have been encountered in the card input stream.

- Control is then passed to the appropriate control section within DIRECT by the use of a FORTRAN Computed GO TO statement. The Computed GO TO statement uses as its "key" variable the integer variable which contains the integer code from LIST array specifying the routine to be executed.

GENERAL OVERVIEW

All routines in the Texas Trip Distribution Package are referenced by name. The names merely need to be entered on the CONTROL cards in the sequence in which the routines are to be executed. The CONTROL cards must be the first records in the input data stream entered from unit 5. HEADING cards may be intermingled with CONTROL cards in any manner, but no other cards should be encountered before the last CONTROL card. Each CONTROL card that is encountered is scanned for valid control entries. Any improper entries or invalid coding will result in program termination immediately after the first card not identified as either a CONTROL or HEADING card is encountered. Such a termination will produce a STOP code of 1.

Regardless of how many CONTROL cards are used, only a total of 40 routines may be processed at one time. If more than 40 routines are specified, the first 40 entries will be executed and the program will then terminate with a STOP code of 10. The entry STOP does not actually reference a routine but is a command used to terminate the execution of the Texas Trip Distribution Package. If subsequent entries are listed on a CONTROL card following the STOP command, they will be processed and checked for validity, but will never be executed because the program will terminate when the STOP command is encountered. When the Texas Trip Distribution Package encounters the STOP entry in its control sequence, the package terminates with a STOP code of 0. At some computer installations, a STOP code of 0 is considered as the normal termination and does not appear on the processing job log.

It is not essential to enter the STOP command in the CONTROL card sequence. If the STOP command is omitted, a stop entry will be furnished by the subroutine which interprets the CONTROL card so the Texas Trip Distribution Package may terminate with a STOP code of 0, if processing has progressed properly.

The HEADING card may be entered at any point in the data card input stream except in the middle of a contiguous set of data. Normally, a HEADING card may appear as the first card in the input data stream to serve as identification of the cards following it. Occasionally, if a HEADING card is placed as the last card in the data card input stream, an error message will result which implies that the program attempted to read more input data than was provided. This error message may simply be disregarded, or the practice of feeding a HEADING card last may be avoided, or an extra card such as a blank card might be fed in as the last card.

Each time control is passed from one routine to another, the data card input stream is checked for the existence of a HEADING card appearing as the next record. This feature permits changing the heading between routine execution. In normal operation, this feature is not frequently needed.

If successive HEADING cards are entered amid the data card input stream, it may not be desired that the second HEADING card change the heading immediately prior to the execution of the second routine in the sequence. In this instance, it may be desirable to utilize the MODIFY routine to space the second HEADING card as desired. In other words, a &VALUES card which contains no parameters, but is closed by

an &END, may be inserted between the two consecutive HEADING cards. At the point at which it is desired to change the heading, the MODIFY entry can be placed in the control sequence.

The HEADING card may also be used as a separator to distinguish between two different data sets which might be used sequentially. As an example, two different sector structures may be utilized in the analysis of travel patterns within a large urban area. It might be planned to execute the GET routine using one sector structure and then redefine the sector structure by executing the EQUATE routine and re-execute GET to summarize the movements with regard to the second sector structure which might contain more or less detail than the first structure. Clearly, there must be some way for the program to distinguish where the first set of EQUALS cards ends, and the second set begins. A HEADING card between the two sets of EQUALS cards will aptly fulfill this purpose. The message on the separator HEADING card could simply be duplicated from the original HEADING card, however, in the context of the particular example cited, it would very likely be beneficial to change the heading message.

It is fully acceptable to operate the Texas Trip Distribution Package without supplying any heading messages. If no HEADING card is encountered, or until the first HEADING card is encountered, headings will simply be blank.

I N D I V I D U A L R O U T I N E S

INTRODUCTION

OPERATION OF INDIVIDUAL ROUTINES

ACCEPT
ALTER
BUILD
EDIT
EQUATE
EXPAND
GET
IMPOSE
LIST
MATCH
MODEL
MODIFY
PACK
PERUSE
PRINT
REFINE
RESTART
SCREEN
SET
SUM
SWITCH
UNPACK
WRITE

INTRODUCTION

After having reviewed the general operation of the package, the next level of documentation of interest to data processing personnel is the operation of each of the routines available to the user of the package. This section, therefore, documents the operation of the package with regard to each of the twenty-four routines. For convenience, the information contained in the "Descriptions of Individual Routines" section of the Operating Manual for the Texas Trip Distribution Package (Research Report 167-1) has been included in this section thereby lessening the need for cross referencing between the manuals.

The documentation functions served by this section are:

- To provide a review concerning the use of each of the routines.
- To provide the calling sequence of programs used in conjunction with a particular routine.
- To provide sufficient information regarding the operation of each of the programs used by a particular routine so that the particular program(s) of interest may be identified.

After having identified the particular program(s) of interest, the flowcharts (contained in Section III) used in conjunction with the information concerning significant variables and arrays (Section IV) and data set and data card formats (Section V) should provide the next level of documentation.

OPERATION OF INDIVIDUAL ROUTINES

The description of the operation of each routine in the Texas Trip Distribution Package has been divided into ten sections. These sections state the routine's function, execution requirements, parameter references, data set references, data card references, operation, printed output, user considerations, sequence of programs called, and provide a brief description of each of the individual programs.

The first section, entitled "Function," contains a very brief statement describing the routine's function.

The second section is titled "Execution Requirements." The statements under this heading will indicate whether the program is an independent or dependent routine. This classification is based upon the arrays which are held in core. Several of the routines require that arrays be defined before the routines are executed; these are classified as dependent routines since they require the prior execution of another routine to define the arrays. The routines to be executed in advance of the dependent routines are noted. Due to reuse of much of the core storage, several of the independent routines can destroy key arrays. Therefore, the status of the key arrays is noted with regard to each routine. Additional information concerning these arrays is included in Section IV (significant variables and arrays) of this manual.

The third section is "Parameter References." Under this heading, either one or two subheadings may appear entitled "Required" and/or "Defined." The "Required" column refers to parameters which are

required for proper execution of the routine. The "Defined" column refers to parameters which are either evaluated or revised during the execution of the routine. All parameters referenced under either of these two subheadings appear in the VALUES namelist. Several of the parameters in the "Required" list are shown equal to a value which is enclosed in brackets. Values enclosed in brackets are default values. These are shown in instances when it is likely that prior routine executions have not affected the parameter. Parameters for which no default value is shown should have been defined by the user, or by a prior execution of another routine. Some of the "Defined" parameters are shown being equated to another parameter. These are pointers which are being redefined to point to a different data set. Additional information concerning parameters is included in Section IV (significant variables and arrays) of this manual.

The fourth section is "Data Set References." Any of three columns may appear under this heading labeled as "Input," "Scratch," and/or "Output." The entries appearing under any of these columns may be either symbolic data set references or pointers. Pointers merely refer to certain data sets, and these are changed following the execution of various routines. This feature usually relieves the user from having to define or change data set references if he exercises discretion in the sequence in which he executes the routines. It should be remembered that the user may redefine the pointer and symbolic data set references through the execution of MODIFY prior to executing any routine in question. Following most symbolic data set names is a value enclosed in brackets. This is the default value of the data set. A value, not enclosed in brackets, which follows a data set is the unit

number which that data set must have; and the user is provided no option to redefine such a data set reference. Additional information concerning the parameters associated with the data sets is contained in Section IV (significant variables and arrays) while additional information concerning the formats of the data sets is included in Section V (Data Set and Data Card formats) of this manual.

The fifth section concerns "Data Card References." Again, column headings marked "Input" and/or "Output" may be encountered. The input data cards must be placed in the data card input stream in the sequence in which they are listed. It should be noted that if stray data cards should appear in the input data stream, these cards will not be processed properly. All of the routines operate in the same manner with respect to card input data. When a program reaches the point where it is to process data cards, it checks the next entry in the data card input stream for the appropriate type of data card. If the data cards are the type expected, they are read until a data card is reached which is not the desired type. This last card then will be saved until needed through the use of the REREAD subroutine. With this procedure, no delimiter is necessary to indicate the end of a particular group of data cards. It should be remembered that HEADING cards are the only data cards which may be placed in the input data stream which do not require an explicit program reference, in the CONTROL entry sequence, to be read and processed properly.

The sixth section is "Operation." This section consists of a general discussion of how each routine actually operates without regard to particular programs. Errors leading to abnormal termination conditions are noted.

The seventh section describes the "Printed Output." All of the printed output bears page headings which describe the output. Once the user becomes acquainted with the package, it will not be necessary for him to continually refer to these discussions. In the execution of some of the routines, identical or similar output is received, and rather than repeat long discussions, a mere statement has been provided indicating where the particular type of output is discussed.

The eighth section is entitled "User Considerations." The discussions under this heading vary in nature from items of computational efficiency to the basic philosophy of the distribution procedure.

The ninth section is entitled "Sequence of Programs Called." This section contains a diagram illustrating the sequence of programs called during the execution of the routine. This diagram serves two important functions:

- It serves as a summary of the control sections used in executing the routine.
- It provides a convenient "trace back" capability.

For example, if a programmer is interested in a particular program, say INSERT in the routine ACCEPT, the diagram indicates that the INSERT subroutine is called by the subroutine DIRECT. It can also be seen from the diagram that the INSERT subroutine does not call any other subprograms. It should also be noted that Table 2 in the Introduction of the manual may be referenced to determine whether any of the other routines utilize the INSERT subroutine.

The tenth section is entitled "Summary of Individual Programs." This section contains a brief description of the operations performed

by each of the programs used in executing the routine. Since the operations performed by the MAIN program and the control functions performed by the subroutine DIRECT have been discussed in Section I (GENERAL OPERATION) they have not been included in this section. The operations performed by the subroutine DIRECT with respect to the particular routine of interest are, of course, discussed.

Function

The ACCEPT routine accepts trip generations, trip lengths, sector interchange bias compensations, and the production-interaction curve from data cards.

Execution Requirements

ACCEPT is an independent routine when executed in conjunction with EXPAND. It requires no initialization, destroys no key arrays, and prepares some key arrays used by other programs.

The ACCEPT routine is a dependent routine in all other applications. It must be preceded by an execution of SET even if a survey data trip matrix is not available. Intervening executions of any routines which destroys key arrays will jeopardize the functioning of ACCEPT. The ACCEPT routine prepares key arrays which are used by other programs.

Parameter References

Required	Defined
XP (required only if INTERACTION cards are not provided and EXEMPT=F)	TV
	UT
	AN
TYPE = [blank] (optional)	
	PN

Data Set References

None

Data Card References

Input

FORMAT (for Generation cards)

GENERATION cards

FORMAT (for Length cards)*

LENGTH cards*

FORMAT (for BIAS cards)*

BIAS cards

FORMAT (for INTERACTION cards)*

INTERACTION cards*

Operation

The ACCEPT routine begins by attempting to read a FORMAT cards. If this card is not encountered, execution of the Trip Distribution Package is terminated immediately with a STOP code of 5. If the identification code on the FORMAT card is equal to the type parameter, or if either of these is blank, the format is accepted. If additional format records are encountered, they are judged by these criteria. The last encountered acceptable format is used. If no acceptable format is found, the program terminates with a STOP code of 3.

All GENERATION cards are read. These cards may be in any sequence, but if one entry is not encountered for every centroid and external station, the missing centroid and/or external station numbers are printed and the Trip Distribution Package terminates with a STOP code of 3.

After processing the GENERATION cards, the program searches for LENGTH cards, BIAS cards, INTERACTION cards, and associated FORMAT cards. All of these entries are optional, including the FORMAT cards. If FORMAT

*Optional

cards are not provided for the LENGTH cards and BIAS cards, the last encountered FORMAT card will be used. This is inappropriate since the formats are usually incompatible. The INTERACTION cards, if present, must have a format card. If LENGTH, BIAS, and INTERACTION cards are being supplied, the LENGTH cards should be entered first and the INTERACTION cards last.

Printed Output

A listing of the INTERACTION cards.

User Considerations

If existing trips are being distributed and survey data are available the internal productions read in through the GENERATION cards are scaled so that the total productions from the survey data will equal the total productions fed in on GENERATION cards. The internal attractions are always scaled so that the total attractions will equal the total productions. The trip length distribution for internal movements is scaled so that the total trips in the trip length distribution equals the total internal productions. Likewise, the external distribution is scaled so that the total trips equals the total trip generations through the external stations.

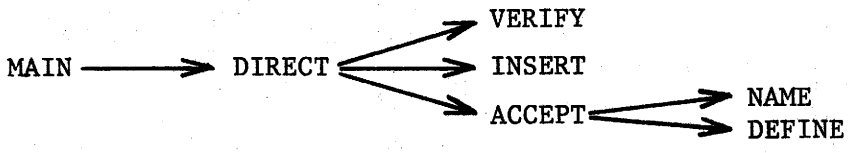
If existing trips are being distributed and survey data are available, the trip length distribution will be obtained from the survey data. Any values entered on LENGTH cards will override those found in the survey data.

If any BIAS cards are encountered, the corresponding factors are applied during the trip distribution and no bias detection is attempted

by the program, even if existing trips are being distributed and survey data are available.

If ACCEPT is being used in conjunction with the EXPAND routine, no scaling is performed.

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine ACCEPT, the subroutine DIRECT performs the following operations:

1. If the parameter N is zero (its default value), the header record of the MT data set is read to obtain the value of N.
2. If the parameter NF is one (its default value), an attempt is made to read the header record from the MS data set.
   (a) If MS is a dummy data set, the unit is rewound and the program proceeds to step 3.
   (b) If MS is <u>not</u> a dummy data set:
      (1) The following is obtained from the header record: values for the parameters N, NF, and M; an

array containing the heading obtained from the heading card and the date that the data set MS was built; and an array containing the number of zone pairs at each separation.

(2) The subroutine VERIFY is called

(3) The logical variable GROWTH is set to .FALSE. to indicate that a trip distribution is being performed. This also means that the routine EXPAND will not be executed.

3. The subroutine INSERT is called.

4. The subroutine ACCEPT is called.

5. The value for each parameter in the VALUES namelist is printed

6. If the value of the last centroid is larger than the value of the last external station, the package terminates with a stop code of eleven, otherwise control is returned to the program MAIN.

VERIFY: This program checks the following parameters to determine if their values have exceeded the capacity of this package; N, NF, NR.

INSERT: This program initializes the bias factors to be used on the sector movements to 1.0

ACCEPT: This program reads trip generations from GENERATION cards by first reading the FORMAT for the GENERATION cards and then reading the trip generations into arrays P and A. If provided, the FORMAT for the LENGTH cards, the LENGTH cards, the FORMAT for

the BIAS cards, and the BIAS cards are also read. The trip length data, if available, is stored in array F. The bias data, if available, is stored in array RR. The Production Interaction curve is read if available and stored in labeled common ELIM.

NAME: This program prints the name of the routine being executed at the time it is called.

DEFINE The arrays built in ACCEPT are used to calculate the relative attraction model values and trip length distribution model values. These values are used later to calculate the model trip matrix. It should be noted that if LENGTH cards are read by ACCEPT, the values calculated in the trip length distribution model will be ignored.

Function

The ALTER routine alters a trip matrix to compenstate for changes in accessibility created through changes in a transportation system.

Execution Requirements

The ALTER Program is a dependent routine. It must be preceeded by the execution of ACCEPT or REFINE to establish the desired trip length frequency. It does not affect the key arrays.

Parameter References

Required	Defined
	MS = NEWSEP
	MT = ALTRP

Data Set References

Input	Output
NOWSEP = [4]	ALTTRP = [22]
NEWSEP = [16]	
MODTRP = [3]	

Data Card References

None

Operation

The ALTER routine reads one record from the trip matrix, one record from the old separation matrix, and one record from the new separation matrix. Each interchange volume within the record being considered is

adjusted based upon the change in travel separation between the old and new separation matrix. A record containing the adjusted interchange volumes is then written and the process repeated.

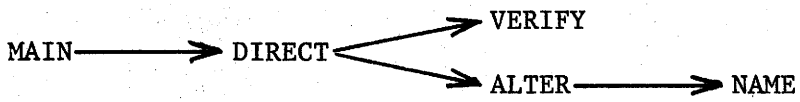
Printed Output

None

User Considerations

This routine is currently of research interest only and has been included only for the convenience of on-going research. It is not, therefore, recommended for use in urban transportation studies.

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine ALTER, the subroutine DIRECT performs the following operations:

1. If the parameter N is zero (its default value), the header record on the data set MODTRP is read to define this value.
2. The subroutine VERIFY is called.
3. The subroutine ALTER is called.
4. The parameter MS is set equal to NEWSEP
5. The parameter MT is set equal to ALTTRP.
6. Control is returned to the program MAIN.

VERIFY: This program checks the following parameters to determine if their values have exceeded the capacity of this package; N, NF, NR.

ALTER: This program alters the trip volumes based on the change in the separations between centroids. It first reads the trip volumes off the data set MODTRP, the original separations off the data set NOWSEP, and the new separations off the data set NEWSEP for each centroid. New trip volumes are then computed based on the change in separation and written on the data set ALTTRP and the process is repeated.

NAME: This program prints the name of the routine being executed at the time it is called.

Function

The BUILD routine builds trip matrices of the type and trip purpose specified by a trip CATEGORY card from the sorted, abbreviated, trip records.

Execution Requirements

BUILD is an independent routine. It requires no initialization. It does not effect the key arrays. If BUILD is executed after SCREEN has been executed, the SORTOUT data set is copied to the RECORD data set for preservation. If SCREEN is not executed prior to BUILD, the RECORD data set is assumed to have been previously prepared and the SORTOUT data set is not copied.

Parameter References

Defined
N
M
TYPE
MT = NOWTRP

Data Set References

Input	Output
SORTOUT = 10	RECORD = [14]
RECORD = [14]	NOWTRP = [2]

Data Card References

Input
CATEGORY

Operation

The BUILD routine reads and interprets one CATEGORY card. The CATEGORY card provides the information which controls the selection of the trip reports used to construct the trip matrix. If a CATEGORY card is not encountered, the Texas Trip Distribution Package terminates immediately with a STOP code of 8. After the CATEGORY card is interpreted, the abbreviated trip records are scanned for entries of the desired category, and the desired trip matrix is formed. The trip matrix is written on the NOWTRP data set.

Printed Output

One line is printed during the execution of the BUILD routine. This line displays the trip matrix identification as supplied through the CATEGORY card, the sum of the trips contained in the matrix, and a string of consecutive zeroes and ones. The sum is printed for the user to check against any other source which he has available, and the number string is printed to aid in examining the CATEGORY card if an error is apparent. The number string may be interpreted as forty one-digit numbers which are referenced by position. The first nine should be ignored. The tenth refers to category 10, etc. A zero means it is ignored; a one means it is desired.

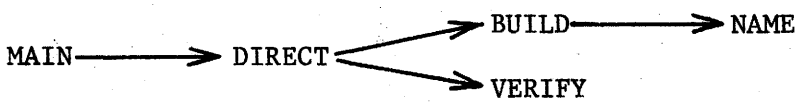
User Considerations

If no executions of MATCH or SWITCH are planned, and if BUILD is not to be re-executed during the processing job, the RECORD data set may be defined as a dummy data set and the SORTOUT data set preserved

in its place and later entered as the RECORD data set at the next execution of BUILD.

If a CATEGORY card is not entered, the SORTOUT data set is copied on the RECORD data set. This will preserve the data set thereby avoiding the re-execution of the SCREEN routine.

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine BUILD, the subroutine DIRECT performs the following operations:

1. If the variable LB is equal to the parameter RECORD, the header record on the data set RECORD is read to define the value of the parameter N. LB is equal to RECORD only if the routine SCREEN has not been executed.
2. The subroutine BUILD is called.
3. The parameter MT is set equal to the parameter NOWSEP.
4. The subroutine VERIFY is called.
5. Control is returned to the program MAIN.

BUILD: The primary function of this program is to build a trip matrix. To achieve this purpose, a CATEGORY card is read to determine the type of trip matrix desired. The data set REPORT is then read to obtain the desired trips and construct the trip matrix which is outputted on the data set NOWTRP.

NAME: This program prints the name of the routine being executed.

VERIFY: This program checks the following parameters to determine if their values have exceeded the capacity of this package; N, NF, NR.

Function

The EDIT routine edits the interzonal travel separations obtained from the assignment package, and writes a separation matrix for use in trip distribution.

Execution Requirements

EDIT is an independent routine. It requires that the value of parameter M be preset. It does not prepare any key arrays, but if executed indiscriminately it could destroy some of them. However, since EDIT prepares the separation matrix used by most of the other routines, this controls its execution sequence and almost eliminates the danger of destroying key arrays.

Parameter References

Required	Defined
M	NF
EXTEND = [0]	MS = NOWSEP

Data Set References

Input	Output
RAWSEP = [8]	NOWSEP = [4]

Data Card References

Input

SEPARATION (optional)

Operation

The EDIT routine is used to edit the interzonal separations that result from the assignment package and convert them to a form usable by the Trip Distribution Package. The EDIT routine first scans the entire interzonal separation data set in order to determine the largest value. This value is then written in a parameter record at the front of the data set. If additional codes are to be used, the largest value is incremented by the number indicated by the variable EXTEND. Any zero value found in the interzonal separations is replaced by a value of unity. Separation cards are optional and may be supplied to replace any value found in the interzonal separation data set with any desired value.

Printed Output

The EDIT routine prints the table titled "SEPARATION REVISIONS RESULTING FROM THE EDITING PROCESS" and the maximum internal separation.

User Considerations

An optional field is provided in the SEPARATION cards for special separation codes. These special separation codes must be integers in the range of one to the value of the parameter EXTEND plus one. When a SEPARATION card is encountered with a special separation code, the EDIT routine will compute a replacement separation value for the specified zone pair as follows:

$$\begin{bmatrix}\text{replacement}\\ \text{separation}\\ \text{value}\end{bmatrix} = \begin{bmatrix}\text{largest internal separation}\\ \text{detected in the RAWSEP data}\\ \text{set}\end{bmatrix} + \begin{bmatrix}\text{special}\\ \text{separation}\\ \text{code}\end{bmatrix}$$

The user must be careful when using special separation codes for interzonal separations since the selection of eligible zone pairs for the interaction constraint in the MODEL routine is based on the accessibility measure:

$$\frac{\text{Attraction volume}}{\text{Separation}}$$

It is possible, therefore, that few, if any, of the interzonal movements with a special separation code would be selected as eligibile zone pairs. To avoid this problem will require that the interzonal movements with a special separation code be imposed via ADMIT cards in the IMPOSE routine. Intrazonal movements do not pose a problem since they are selected as eligible zone pairs so long as they have non-zero production and attraction volumes regardless of their separation.

If SEPARATION cards are used, the EDIT routine will check the SEPARATION cards for the following conditions:

- A special separation code which is greater than the value of the parameter EXTEND plus one.
- A separation value which is greater than the largest internal separation (including the separation values computed for the special separation codes).
- A SEPARATION card with both a separation value and a special separation code.
- An invalid centroid number (i.e., a centroid number which is greater than the value of the parameter N)

If either of these conditions exist then a warning message will be printed and the JOB will be abnormally terminated following the EDIT routine with a stop code of 16. When either of these conditions are encountered, the following values will be entered in the separation matrix built by EDIT:

- If a special separation code is encountered which is greater than the value of EXTEND plus one, then the SEPARATION card is ignored.
- If a separation value is encountered which is greater than the value of the largest internal separation (including the separation values computed for the special separation codes), then the SEPARATION card is ignored.
- If a SEPARATION card is encountered with both a separation value and a special separation code, then the special separation code is ignored and the separation value (if valid) is used.
- If an invalid centroid number is encountered then the SEPARATION card is ignored.

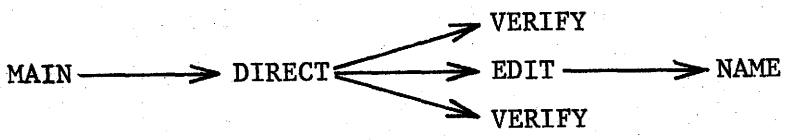
The EDIT routine sets the value of the parameter NF as follows:

$$NF = \left[\begin{array}{l}\text{Largest internal separation} \\ \text{detected in the RAWSEP data} \\ \text{set}\end{array}\right] + \text{EXTEND} + 1$$

User Considerations

None

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine EDIT, the subroutine DIRECT performs the following operations:

1. If the value of the parameter N is zero (its default value), the header record on the data set RAWSEP is read to define its value.

2. If the value of the parameter M is greater than the value of the parameter N, the package terminates with a stop code of eleven.
3. The subroutine VERIFY is called.
4. The subroutine EDIT is called.
5. The subroutine VERIFY is called.
6. The parameter MS is set equal to the parameter NOWSEP.
7. Control is returned to the program MAIN.

VERIFY: This program checks the following parameters to determine if their values have exceeded the capacity of this package; N, NF, NR.

EDIT: The primary function of this program is to build a separation matrix for use by subsequent programs in this package. The RAWSEP data set, which contains the separation matrix built by the Large or Small Assignment Package, is used as input. The separations for each centroid in the network are read and rounded off to the nearest whole number. Any zero separation is set to one. All internal to external trips are given the same separation value (i.e., maximum internal separation + 1). External to internal, and through trips are given the values: maximum internal separation + 2 and maximum internal separation + 3 respectively. Separation cards, if provided, are used to change the separation between any two centroids. The revised separations are written on the data set NOWSEP.

NAME: This program prints the name of the routine being executed.

Function

The EQUATE routine equates centroids to sectors.

Execution Requirements

EQUATE is an independent routine. It requires that the value for parameter N be present. It prepares one key array which defines the sector equivalency. It does not destroy any key arrays. The SET and GET routines contain automatic calls to the EQUATE program provided that sector equivalences have not been established previously. The EQUATE routine may be used to replace one set of sector equivalences with a different set.

Parameter References

Required

N

Data Set References

None

Data Card References

Input

EQUALS (optional)

Operation

Sector to centroid equivalence are obtained by the EQUATE routine through EQUALS cards. The EQUATE routine attempts to read an Equals

card. In the event that this card is not encountered, the EQUATE routine establishes an equivalence of all zones with sector one. Subsequent processing interprets this to mean that the sector equivalence feature is not being used. If equals cards are encountered, they are processed until the last equals card has been read. All zones are then examined to see if equivalences with sectors have been established. If any unequivalenced zones are discovered, a default sector is established. The default sector is assigned the next number larger than the last defined sector. All remaining zones are equated to the default sector. Multiple entries for any zone are noted in a message and the last encountered equivalence is retained. A table describing the resulting equivalences is finally written.

It should be noted that centroid and external station numbers and sector numbers are checked during processing. Any invalid entries are disregarded. EQUALS cards may be processed in any order. It is recommended that sectors be numbered consecutively starting with the value of one, but this is not a requirement. If the number of sectors used exceeds the capacity of the package, a message will be written and processing terminated. It is cautioned that the use of more than about 15 sectors may be found to be unwieldy in the printed output. Only 15 numbers are printed per line and if more than 15 sectors are used the output tables become "folded".

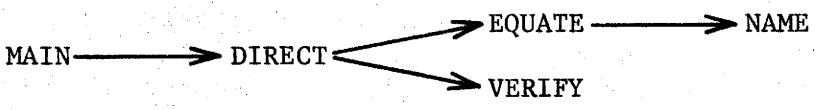
Printed Output

A table of the zone to sector equivalences is printed. If any zones are equated to a default sector a message is printed identifying the sector to which they were equivalenced. A message is printed if multiple entries are encountered for any zones.

User Considerations

The routine EQUATE allows the user a convenient means for correcting mistakes made in keypunching EQUALS cards. Since the EQUATE routine uses the last encountered equivalence, all that is required is that corrected EQUALS cards be added to the back of the already punched EQUALS cards. Messages regarding multiple entries should be ignored in this situation.

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine EQUATE, the subroutine DIRECT performs the following operations:

1. The subroutine EQUATE is called.
2. The subroutine VERIFY is called.
3. Control is returned to the program MAIN.

EQUATE: The primary function of this program is to establish zone to sector equivalences. EQUALS cards containing the sector numbers and the zone numbers which are in those sectors are read and an array is established containing the zone to sector equivalences. If any zone in the network has not been

equivalenced to a sector, that zone is equivalenced to a default sector whose value is one greater than the largest sector number. If no EQUALS cards are encountered, all zones are equivalen-ed to sector one.

NAME: This program prints the name of the routine being executed.

VERIFY: This program checks the following parameters to determine if their values have exceeded the capacity of this package; N, NF, NR.

Function

The EXPAND routine expands a trip matrix, directionally.

Execution Requirements

The EXPAND program is a dependent routine. It must always follow the execution of either SET or ACCEPT. Intervening executions of either GET or MATCH will destroy input arrays which are required by EXPAND.

Parameter References

Required	Defined
LIMIT = [5]	MT = MODTRP

Data Set References

Input	Output
NOWTRP = [2]	MODTRP = [3]

Data Card References

None

Operation

The EXPAND routine performs iteratively. The limit parameter governs the number of iterations which are repeated. The resulting trip matrix is written on the MODTRP data set during the last iteration.

Printed Output

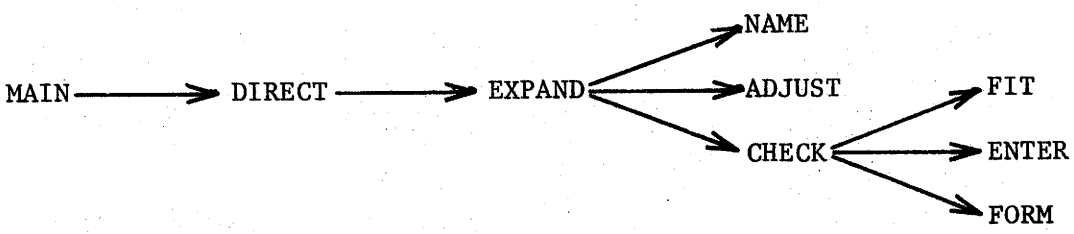
At the end of each iteration through the routine EXPAND, a table is printed which reflects the success of the balancing process in

applying a destination volume constraint. Each entry in the table refers to an origin zone number, and the user is referred to the discussion presented for the routine MODEL for an interpretation of this output.

User Considerations

In using the EXPAND routine for expanding external-through movements, each iteration requires skipping through all of the internal centroids just to reach the external centroids.

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine EQUATE, the subroutine DIRECT performs the following operations:

1. The subroutine EXPAND is called.
2. The parameter MT is set equal to the parameter MODTRP.
3. Control is returned to the program MAIN.

EXPAND: This program expands the trips in a trip matrix. All the trips for each zone (or origin) are read from the data set NOWTRP. The external trip volumes are then selected and expanded using the productions and relative attraction factors. This process is performed iteratively and after the last iteration, all the trip volumes are written on the data set MODTRP.

NAME: This program prints the name of the routine being executed.

ADJUST: This program adjusts the relative attraction factors in an effort to balance the resulting trip volumes with the desired trip volumes.

CHECK: This program calls several other programs which perform a comparison of the resulting and desired values.

FIT: This program zeroes the variables to be used in the following curve fit.

ENTER: This program is an entry in the program FIT. It takes the variables zeroed in FIT and performs summations on the resulting and desired values.

FORM: This program is an entry in the program FIT. It takes the values calculated in ENTER and does a linear curve fit on the resulting versus the desired values. Statistical measures are printed which indicate the success of the curve fit.

Function

The GET routine gets trip generation, trip length, and sector interchange data and prints these data for inspection.

Execution Requirements

GET is an independent routine. It requires no initialization. It prepares no key arrays but can destroy some if executed improperly.

Parameter References

Required	Defined
PLOT = [F]	TV
XP (if plot = T and INTERACTION cards have not been input)	AN PN

Data Set References

Input	Output
MT	PLOTTAPE (if plot = T)
MS (DD Dummy optional)	

Data Card References

Input

EQUALS (optional)

Operation

The GET routine first checks to see if secotr equivalences have been defined. If they have not been defined and EQUALS cards are available; then the sector equivalences are established. The GET routine checks to see if a separation matris is available, and if it is not, the trip length data will be sacrificed. If the parameter PLOT is equal to TRUE, then Calcomp plots will be prepared.

Printed Output

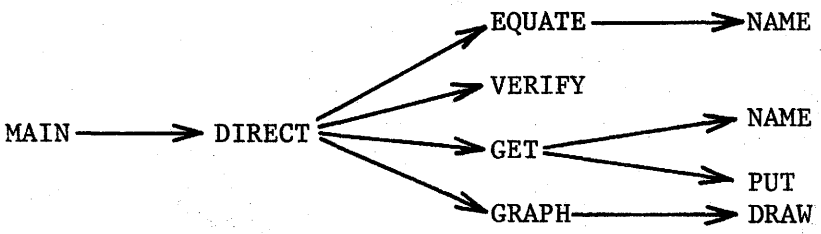
The reader is referred to the SET routine for a discussion of the printed output.

User Considerations

The availability of a separation matrix has been made optional to allow the user to examine the trip generations by zone and by sector without having to wait for the network coding to be completed. Furthermore, it permits examining the results from the EXPAND routine which is an application that does not require a separation matrix and represents an instance when one is not likely to be available. Calcomp plots will not be prepared if the separation matrix is defined as a dummy data set.

If GET is executed as an isolated entry with PLOT equal ture, only the trip length distribution of the associated trip and separation matrices will be plotted. If GET is executed in a sequence with prior executions of either MODEL or REFINE, the plot will show both the desired and resulting trip length distributions on the same graph, for comparison.

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine GET, the subroutine DIRECT performs the following operations:

1. If either of the parameters N or M is zero, the header record on the data set MT is read to define both parameters.
2. If the parameter MR is less than or equal to one (its default value), the subroutine EQUATE is called.
3. If the parameter NF is equal to one (its default value) an attempt is made to read a header record from the data set MS.
   (a) If MS is a dummy data set, the unit is rewound and the logical variable PLOT is set. FALSE.
   (b) If MS is <u>not</u> a dummy data set, the header record on MS is read to define the value of NF.
4. The subroutine VERIFY is called.
5. The subroutine GET is called.
6. If the logical variable PLOT is .TRUE., the subroutine DRAW is called.
7. Control is returned to the program MAIN.

EQUATE: This program is called by DIRECT only if MR (the largest sector number) is less than or equal to one. The primary function

of this program is to establish the zone to sector equivalences. The EQUALS cards are read and an array is established containing the zone to sector equivalences (this is a one-dimensional array which has a position for each zone and contains the sector equivalences. The array is then scanned for unequivalenced zones. Unequivalenced zones are assigned a default sector number which is one greater than the largest sector number found on the EQUALS cards. If no EQUALS cards are encountered, all zones are equivalenced to sector one.

NAME: This program prints the name of the routine being executed (i.e., EQUATE or GET).

VERIFY: This program checks the following parameters to determine if their values have exceeded the capacities of this package; N, NF, NR.

GET: This program is an entry in the SET program (see flow chart for SET). It builds the same arrays as SET using the trip volumes read from the data set MT and the separations read from the data set MS. The primary difference between GET and SET is that GET will not build the NEGSEP data set. In most applications, GET is used to output the movements and interactions found in the modeled trip matrix.

PUT: This program takes selected arrays built in GET and prints the trip length characteristics of the trip matrix on the data set MT.

GRAPH: This program builds arrays to be plotted on the calcomp plotter.

DRAW: This program takes two sets of arrays which were built in GRAPH and calls the appropriate system plot routines, which in turn build a data set on the unit PLOTTAPE. PLOTTAPE is the input to the calcomp plotter.

Function

The IMPOSE routine imposes movements which are to be included in the trip distribution.

Execution Requirements

IMPOSE is an independent routine. It requires no initialization. It does not affect key arrays.

Parameter References

	Defined
	MS = IMPSEP

Data Set References

Input	Output
MS	IMPSEP = [15]

Data Card References

Input

ADMIT (or LOCAL) cards

Operation

The IMPOSE routine reads ADMIT cards and determines movements to impose during the trip distribution. If ADMIT cards are not encountered, then the Trip Distribution Package terminates with a STOP code of 6. ADMIT cards must be in numerical sort on the production (or origin) centroid numbers. Each ADMIT card is read, interrogated for errors, and processed.

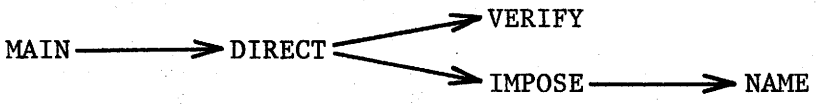
Printed Output

No printed output results from a successful execution of the IMPOSE program. However, any errors detected during its execution are printed.

User Considerations

The user should be aware that the IMPOSE routine makes only one entry for each entry represented on the ADMIT cards. In order to admit both directions of travel between two zones, two distinct entries must be made through the ADMIT cards.

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine IMPOSE, the subroutine DIRECT performs the following operations:

1. If the parameter N is zero (its default value), the header record on the data set MS is read to define the values of N and NF.
2. The subroutine VERIFY is called.
3. The subroutine IMPOSE is called.
4. The parameter MS is set equal to the parameter IMPSEP.
5. Control is returned to the program MAIN.

VERIFY: This program checks the following parameters to determine if they have exceeded the capacity of this package; N, NF, NR.

IMPOSE: This program inputs movements which are imposed on the modeled trip distribution. An ADMIT or LOCAL card is read. The separations for the production zone indicated on the ADMIT or LOCAL card are read from the data set MS and the separations for the specified zone pairs are set to the negative of their original value. The separations for that production zone are then written on the data set IMPSEP and the process repeated. The IMPSEP data set will, therefore, contain the separation matrix with the separations for the imposed movements set to negative.

Function

The LIST routine prints the trip length distribution for each zone individually.

Execution Requirements

LIST is an independent routine. It requires no initialization. It does not affect any key arrays.

Parameter References

None

Data Set References

Input

MT = [2]

MS = [4]

Data Card References

None

Operation

The LIST routine simply reads the trip matrix and separation matrix, simultaneously, and prints the trip length characteristics for each production zone.

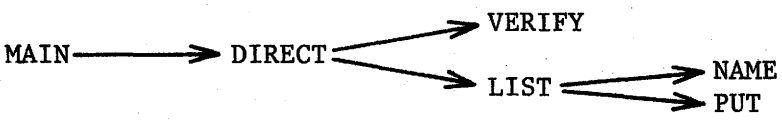
Printed Output

The output resulting from the LIST routine is similar to that described for the trip length characteristics under the SET routine.

User Considerations

Due to the execution time required and the amount of printed output prepared, the LIST routine should be used only when necessary.

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine LIST, the subroutine DIRECT performs the following operations:

1. If the parameter N is zero (its default value), the header record on the data set MT is read to define its value.
2. If the parameter NF is equal to one (its default value), the header record on the data set MS is read to define its value.
3. The subroutine VERIFY is called.
4. The subroutine LIST is called.
5. Control is returned to the program MAIN.

VERIFY: This program checks the following parameters to determine if their values have exceeded the capacity of this package; N, NF, NR.

LIST: This program reads the separations and trip volumes for one centroid from the data sets MS and MT respectively. Arrays are then built using those values and passed into PUT. This process is repeated for each centroid.

NAME: This program prints the name of the routine being executed.

PUT: This program takes the arrays built by LIST and for each centroid, prints its trip length characteristics in tabular form.

Function

The MATCH routine matches the characteristics of two trip matrices against each other.

Execution Requirements

MATCH is an independent routine. It requires no initialization. It destroys all key arrays.

Parameter References

Required	Defined
SAMPLE = [0.125]	SIZE
AMOUNT = [110000]	

Data Set References

Input	Scratch
NOWSEP = [4]	SORTIN = 9 (DD DUMMY optional)
NOWTRP = [2]	SORTOUT = 10
MODTRP = [3]	

Data Card References

None

Operation

The MATCH routine reads a record from three data sets (the survey trip matrix, the model trip matrix, and the separation matrix), then performs comparisons with regard to corresponding interchange volumes. The reciprocal

of the nominal sampling rate is used to establish the cell intervals for a cross classification of interchange frequencies between the survey and model trip matrices. High-volume interchanges which do not fall within the limits of this table are listed on the SORTOUT data set for a separate analysis. If the SORTIN data set is a dummy data set only movements in which the differences between the survey interchange volumes and the model interchange volumes which do not exceed N/10 (where N = the number of centroids and external stations) will be presented on the trip volume difference analysis.

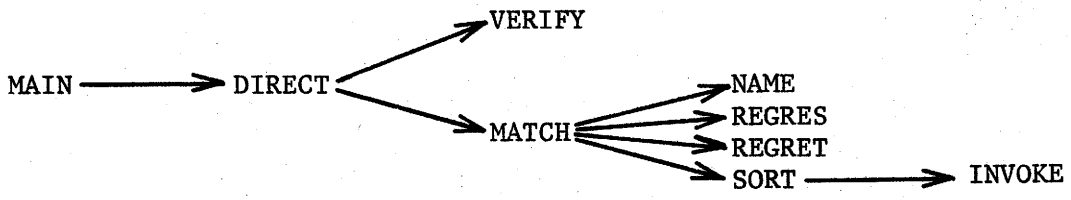
Printed Output

There are five different types of comparisons that are performed by the MATCH routine. The first of these is an analysis of the interchange volumes with respect to production centroids. The second is a volume frequency cross-classification table. Third, a comparison of the high-volume interchanges is printed. Fourth, a comparison of the low-volume interchanges is printed. The fifth is an analysis of the trip volume differences.

User Considerations

If SORTIN is a dummy data set, the BLKSIZE for the SORTOUT data set can be reduced to a small value, such as 244, with no loss of efficiency and some saving of core storage.

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine MATCH, the subroutine DIRECT performs the following:

1. If the value of the parameter N is zero (its default value), the header record on the data set NOWTRP is read to define its value.
2. If the value of the parameter NF is one (its default value), the header record on the data set NOWSEP is read to define its value.
3. The subroutine VERIFY is called.
4. If the maximum capacity of sector combinations for the package is less than 320, the package terminates with a stop code of none.
5. The subroutine MATCH is called.
6. Control is returned to the program MAIN.

VERIFY: This program checks the following parameters to determine if they have exceeded the capacity of this package; N, NF, NR.

MATCH: The primary function of this program is to compare two trip matrices: the trip matrix on the NOWTRP data set (the survey trip matrix) and the trip matrix on the MODTRP data set (the

modeled trip matrix). The zones are processed one at a time. For each zone, the program reads the modeled trips from the data set MODTRP, the survey trips from the data set NOWTRP, and the separations from the data set NOWSEP and makes the appropriate comparisons. Arrays are built for the comparison of the trip volumes for each centroid. High trip volumes are written on unit 10 for use later. The small trip volumes are written on unit 9. Arrays are also built which contain the volume frequency distributions in the model and survey trip matrices, for use in the comparison of the trip volumes between zone pairs at each separation and for use in the comparison of the entire trip matrices. The results are printed for further analysis. The comparisons performed for each zone, for the zone pairs at each separation, and for the trip matrices are accomplished by performing a linear curve fit on the model trip volumes versus survey trip volumes and the results of the fit are printed. A table is printed which shows the frequency that identical trip volumes occurred in both trip matrices. Tables are also printed for both the high trip volumes and the small trip volumes found in the two trip matrices along with their frequency. The final comparison performed and printed is the comparison of the trip volume differences between the trip matrices. This is done for two cases; where the model trips are greater than equal to the survey trips and where the model trips are less than the survey trips.

REGRES: This program is an entry in the program FIT (see the flow chart for FIT). Using the arrays built in MATCH, a linear curve fit is performed and the results of the curve fit printed.

REGRET: This program is an entry in the program FIT (see the flow chart for FIT). It essentially performs the same functions and prints the same output as REGRES.

SORT: This program specifies the fields in the records which are to be sorted and the data control block to be used.

INVOKE: This is an assembly language program which calls the system sort routine. The records on unit 9 are then sorted and placed on unit 10.

Function

The MODEL routine models the distribution of travel interchanges, and writes a modeled trip matrix.

Execution Requirements

MODEL is a dependent routine. It must be preceded by executions of REFINE and/or ACCEPT. Intervening executions of any routine which destroys key arrays will jeopardize the functioning of MODEL.

Parameter References

Required	Defined
FUTURE	MT = MODTRP
UT	
LIMIT = [5]	MS = NOWSEP
EXEMPT = [F]	
DUMP = [T]	

IMPSEP = [15] (DUMMY if not required)

Data Set References

Input	Output
MS	MODTRP = [3]
	SV = [25] (if DUMP = T)

Data Card References

	Output
	FORMAT (if FUTURE = F)
	BIAS

Operation

If EXEMPT is false, the model will be subjected to the interaction constraint, and the eligible zone pairs are selected in a preprocessing phase. The desired number of eligible zone pairs for a given production zone is determined by the production-interaction curve and the eligible zone pairs are selected based on their accessibility to the production zone. The trip distribution for the first interation is then performed. No trip matrix is written until the last iteration is reached. After the initial distribution is performed, the relative values are corrected, and the processes reiterated.

The parameter LIMIT indicates the number of iterations to be repeated. If the parameter FUTURE is FALSE, bias factors will be computed two iterations before the iteration limit is reached. Of course, if no sector structure is utilized, the bias correction feature is inoperative.

If the EXEMPT parameter is TRUE, the model operation is exempted from the interaction constraint and interchange volumes will be calculated for all zone pair combinations. Otherwise, interchange volumes will be computed only if: (1) the eligibility of the zone pair has been imposed through application of the IMPOSE routine, (2) the eligibility of the zone pair has been imposed because of a non-zero survey volume, or (3) because the zone pair was selected as an eligible zone pair during the preprocessing phase is of large enough volume to escape elimination by the interaction constraint.

If the DUMP Parameter is true, various parameters and arrays will be saved after each iteration so that the process can be restarted using the RESTART routine.

Printed Output

If EXEMPT is false three tables result from the preprocessor phase which selects the eligible zone pairs. All three of these tables have "ITERATION 0" in their headings. The first table is the Accessibility Eliminator Function. The columns of this table contain the zone number, the production volume, the desired number of interactions as determined by the production-interaction curve and the number of eligible zone pairs including eligible zone pairs imposed either from survey data or the prior execution of the IMPOSE routine. The remainder of this table has been included only for the purpose of monitoring the operation of the program and, therefore, should be of no interest to the transportation analyst.

The second table is entitled "INTERNAL AND EXTERNAL ELIGIBLE ATTRACTION INTERACTIONS". The three columns in this table contain the zone number, the attraction volume and the number of production zones with which the attraction zone may interact. The third table produced during the preprocessor phase is entitled "ELIGIBLE TRAVEL INTERACTIONS". The two columns in this table contain the separations and the number of eligible zone pairs at each separation. The total number of eligible zone pairs is printed as the sum of the second column. If EXEMPT is true the preprocessor phase is omitted and the above three tables are, of course, not produced.

Three tables of printed output result from each iteration of the model. Each of these tables reflects the success of the balancing process in applying the indirect constraints. First in the printed output is the Attraction Volume Balance. Each entry refers to an attraction

zone number and successive columns show the desired attraction volume, the resulting attraction volume from the model application, the difference between these two volumes, the percentage of error in the model volume as opposed to the desired volume, the weighted significance of the combination of absolute and relative error, the relative attraction value, the correction factor applied to improve the results of the next iteration, and the new relative attraction value which reflects the adjustment of the correction factor. After these items are listed for each of the attraction zones, some statistical measures are printed which indicate the overall agreement between desired and resulting attraction volumes for all zones considered in the group.

The Trip Length Balance is printed next. The same measures are printed as in the Attraction Volume Balance. Each entry, however, refers to a separation value. The last three entries represent the external movements. In addition, the desired and resulting percentage of trips is printed for each separation. Summary statistics are presented at the end.

The SECTOR INTERCHANGE BALANCE is the last of the printed output that appears. Each line of output refers to one sector-to-sector movement. The table contains the sector numbers, the desired sector interchange volume, the resulting sector interchange volume, the difference between the two volumes, the tolerance volume, the percent error, the weighted significance of the combination of absolute and relative errors, the correction factor, and the new bias factor. Values in the column headed "RELATIVE" of the sector interchange balance can be observed to be flagged with an asterisk. In all such cases, the corresponding value will

be observed to be 1.0. This notation is used to designate movements which are not being corrected. Only during the last two iterations will any movements be corrected. Of course, if bias corrections are feed in as input data or if no survey data are available, the sector interchange balance will not even appear.

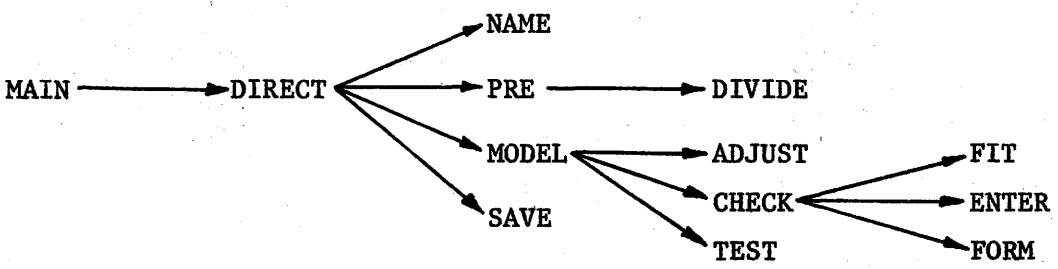
User Considerations

The user should be cautious in interpreting the statistical measures which are provided to indicate the degree of agreement between between desired and model resulting values. The statistical measures can be deceptive. For instance, what normally might be deemed to be an excellent correlation can very easily accompany only a mediocre correspondence between desired and resulting values. The reason for this is simple: there is a very obvious correlation between desired and resulting values. It is the degree of agreement which needs to be evaluated. However, no single index yet discovered does an adequate job of supplying this information. Therefore, the user should examine the individual data values and come to his own conclusion regarding the acceptability of the agreement. This does not mean that the summary statistics cannot be used as a guide.

The column entitled chi-square has some interesting properties. The chi-square sum is shown at the end of the data list and this can indeed be interpreted as the chi-square goodness-of-fit test and this statistic checked against a tabled value. It should be recognized, however, that the chi-square test is very sensitive to "tail" discrepancies, and consequently, a single entry may produce a significant statistical difference with respect to the test. The chi-square column

is presented here as a means for identifying which individual entries contribute most to the disagreement. Individual chi-square entries represent the product of the difference and the percent error columns. The difference column is not an acceptable measure alone since large differences are important with respect to small volumes but may be in a practical sense insignificant with sufficiently large volumes. This reasoning would suggest that the percent error column might be an adequate indicator, and it is with respect to large volumes, but a modeled value may be in error by 100 percent for a small volume and this error be of no real significance. Since chi-square represents the product of the absolute and relative error, it has some attractive characteristics. If both the absolute and relative errors are small, their products will be very small. If either the absolute or relative error is large and the other is very small, the product will be small. As the magnitudes of either error increase, the product increases. When both errors are large, the product is very large. Therefore, large chi-square terms will serve to identify entries which may have unacceptable errors in a combined absolute and relative sense. If there exist many entries, as there will in the attraction volume balance for a large urban area such as Houston, there is no cause for alarm simply because one or two of the entries display large chi-square values and thus, cause the sum to be large enough to imply that significant statistical differences exist.

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine MODEL, the subroutine DIRECT performs the following operations:

1. Subroutine NAME is called to print the routine NAME.
2. If the logical variable FUTURE is TRUE, and attempt is made to read the header record from the data set IMPSEP.
   (a) If IMPSEP is a dummy data set, the unit is rewound.
   (b) If IMPSEP is not a dummy data set, the parameter MS is set equal to IMPSEP.
3. If the logical variable EXEMPT is .FALSE. the subroutine PRE is called.
4. The subroutine MODEL is called.
5. Parameters and arrays used by subroutine MODEL are saved on unit SV if DUMP is .TRUE., then if the iteration number is less than LIMIT subroutine DIRECT goes to Step 4 again.
6. The parameter MT is set equal to the parameter MODTRP.
7. The parameter MS is set equal to the parameter NOWSEP.
8. The variable CLOLD is set to one.
9. If the logical variable FUTURE is .FALSE., the subroutine PUNCH is called.
10. Control is returned to the program MAIN.

DIVIDE: This program takes a set of Accessibility measures and attempts to find an Accessibility measure E for which D Accessibility measures are greater than or equal E within an accuracy of $\pm$A.

MODEL: This program performs a trip distribution using a constrained interactence model (see formulation in Section VI - OTHER INFORMATION).

NAME: This program prints the name of the routine being executed.

PRE: This program initializes the relative arrays for the attractions, trip length frequency, and the accessibility eliminator array which selects the eligible interactions. This program also prints the number of eligible interactions by attraction zone, production zone and separation.

SAVE: This program writes all arrays used by MODEL on unit SV.

ADJUST: This program is used to adjust the relative attraction factors, trip length distributions, travel interactions, and bias factors. These adjustments are performed after each iteration to balance the resulting volumes with the desired volumes.

CHECK: This program calls other programs to determine how well the resulting values match the desired values.

FIT: This program initializes the variables used in the curve fit to zero.

ENTER: This program is an entry in the program FIT. The variables

initialized in FIT are used here to perform summations on the desired and resulting trip volumes.

FORM: This program is another entry in the program FIT. The variables calculated in ENTER are used to perform a linear curve fit and return the values for the intercept and slope. Statistical measures are printed which indicate the success of the fit.

TEST: This program performs a Kolmogorov - Smirnov test on the resulting values versus the desired values. The results are then printed.

Function

The MODIFY routine provides an instrument to define or modify any parameter value at any desired point during program execution.

Execution Requirements

MODIFY is an independent routine. It requires no initialization. It does not affect the key arrays. It may be executed at any point in which it is desired to change any value appearing in the VALUES namelist.

Parameter References

Defined

Any desired parameter in the VALUES namelist

Data Set References

None

Data Card References

Input

&VALUES

Operation

Execution of the MODIFY routine causes an immediate read of the next card in the data card input stream for an &VALUES record. This record is interpreted by the FORTRAN namelist feature. Any parameter appearing in the VALUES namelist may be entered on the &VALUES card.

The value entered for the parameter will replace the former value. If the &VALUES card is coded improperly, the Texas Trip Distribution Package will terminate with a STOP code of 12.

Printed Output

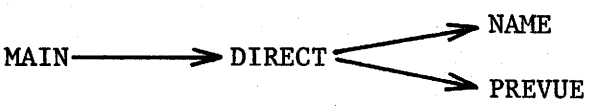
After every execution of the MODIFY routine, the entire VALUES namelist is printed to permit inspection of the current status of the parameter values. This provides the user with the opportunity to verify that his changes were entered as desired, and provides a permanent record in the printed output of the parameter values which were used.

User Considerations

Only the parameters and corresponding values which may be fitted on one data card may be entered during any single execution of the MODIFY routine. Therefore, each &VALUES card requires a separate execution of the MODIFY routine.

A parameter defined by the MODIFY routine may be overridden or re-defined by the execution of any routine which defines the same parameter. For example, if the parameter XP was defined using the MODIFY routine, the subsequent execution of the REFINE routine would calculate a value for XP and substitute the calculated value for the current value of XP.

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine MODIFY, the subroutine DIRECT performs the following operations:

1. The subroutine NAME is called.
2. The subroutine PREVUE is called.
3. The &VALUES card is read.
4. The next data card is read to replace the &VALUES card on the buffer.
5. The value of each parameter in the VALUES namelist is printed
6. If the parameter M is greater than the parameter N and N is not zero (its default value), the package terminates with a stop code of eleven.
7. Control is returned to the program MAIN.

NAME: This program prints the name of the routine that is being executed.

PREVUE: This program verifies that the &VALUES card has been correctly punched. This avoids the possibility of the system reread program becoming caught in a loop which would continue until the system terminates the job.

Function

The PACK routine reformats any trip matrix prepared by this package into the format required by the Texas Large Network Package and the Texas Small Network Package.

Execution Requirements

PACK is an independent routine. It requires no initialization. It does not affect any key arrays.

Parameter References

None

Data Set References

Required	Defined
MT	ASSIGN = [13]

Data Card References

None

Operation

The operation of the PACK routine is simple. It reads a trip matrix (in the format used by this package) record by record, converts each record to the format used by the Texas Large Network Package and Texas Small Network Package, and outputs the new record on the ASSIGN data set. If any interchange volumes are encountered during the process which are too large to be packed in assignment form, the maximum acceptable volume is

substituted, and a message is written to signal this change. It will be very rare for this condition to occur since the interchange volume must have a numerical value that exceeds 64,000.

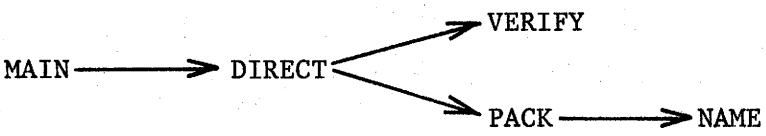
Printed Output

There is no printed output from a successful execution of the PACK routine. However, if an interchange volume in excess of 64,000 is encountered, a message will be printed which reads VOLUME TO LARGE TO ASSIGN and three numbers will follow. The first number represents the production zone, and the second number represents the attraction zone. The third number indicates the magnitude of the trip volume.

User Considerations

None

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine PACK, the subroutine DIRECT performs the following operations:

1. If the value of the parameter N is zero (its default value), the header record on the data set MT is read to define its value.
2. The subroutine VERIFY is called.
3. The subroutine PACK is called.

4. Control is returned to the program MAIN.

VERIFY: This program checks the following parameters to determine if their values have exceeded the capacity of this package; N, NF, NR.

PACK: This program reads a trip matrix prepared by the Texas Trip Distribution Package from the data set pointed to by the MT parameter. As each record is read, it is reformated to the packed format used in the Large and Small Assignment Packages. The "packed" record is written on the data set ASSIGN.

NAME: This program prints the name of the routine being executed.

PERUSE

Function

The PERUSE routine provides a means to print the current parameter values at any desired point during program execution.

Execution Requirements

PERUSE is an independent routine. It requires no initialization. It does not affect the key arrays. It may be executed at any point in which a printed record of the current parameter values is desired. It should be noted that the parameter values are printed at key points in the execution of the Texas Trip Distribution Package.

Parameter References

None

Data Set References

None

Data Card References

None

Operation

Execution of the PERUSE routine merely causes printing of the VALUES namelist so that the current status of the parameter values will be displayed.

User Considerations

None

Sequence of Programs Called

MAIN ⟶ DIRECT ⟶ NAME

Summary of Individual Programs

DIRECT: In the execution of the routine PERUSE, the subroutine DIRECT performs the following operations:

1. The subroutine NAME is called.
2. The value for each parameter in the VALUES namelist is printed.
3. If the parameter M is greater than the parameter N and N is not zero (its default value), the package terminates with a stop code of eleven.
4. Control is returned to the program MAIN.

NAME: This program prints the name of the routine being executed.

Function

The PRINT routine prints a trip matrix for inspection.

Execution Requirements

PRINT is an independent routine. It requires no initialization. It does not affect any key arrays.

Parameter References

None

Data Set References

Input

MT

Data Card References

None

Operation

The PRINT routine simply reads a trip matrix and prints the volumes. Each production or origin zone is treated separately, and the interchange volumes to successive attraction zones are printed ten per row.

Printed Output

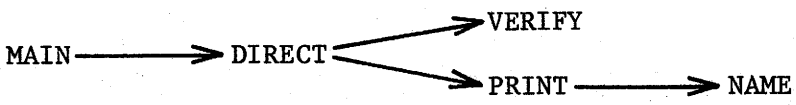
The PRINT routine prints the trip matrix contained on the MT data set. A table is printed for each non-zero production zone. Each table contains the trip volumes from the production zone to each centroid in the network. The table consists of ten columns which are read from left to right such that the first row contains the trip volumes to

The first ten centroids, the second row contains the trip volumes to centroids 11 through 20, etc.

User Considerations

The printed output from this routine is quite lengthy and, therefore, should not be executed unless needed. If the trip matrix is saved, this program can always be executed later, if desired.

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine PRINT, the subroutine DIRECT performs the following operations:

1. If the value of the parameter N is zero (its default value), the header record on the data set MT is read to define its value and the value of the parameter M.
2. The subroutine VERIFY is called.
3. The subroutine PRINT is called.
4. Control is returned to the program MAIN.

VERIFY: This program checks the following parameters to determine if their values have exceeded the capacity of this package; N, NF, NR.

PRINT: This routine reads the trip matrix from the data set MT. The trips for each production or origin zone are read and printed in tabular form.

NAME: This program prints the name of the routine being executed.

Function

The REFINE routine refines the parameter estimates of a set of auxiliary models.

Execution Requirements

REFINE is a dependent routine. It must be preceded by an execution of SET. Intervening executions of any routine which destroys key arrays will jeopardize the functioning of REFINE. The REFINE routine may only be executed when survey data are available. It prepares key arrays which are used by other routines.

Parameter References

Required	Defined
SAMPLE = [0.125]	XP
PLOT = [F]	UT

Data Set References

	Output
	PLOTTAPE (if PLOT = T)

Data Card References

	Output
	VALUES
	FORMAT
	LENGTH

Operation

The REFINE routine uses the array values established by the SET routine and refines various model parameters based upon the survey data. If SET is not executed prior to REFINE, execution is terminated with a STOP code of 4.

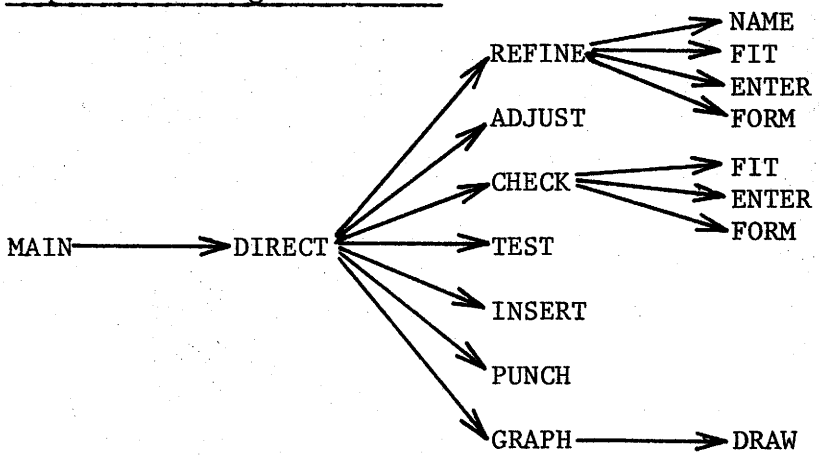
Printed Output

One line is printed for each of the four models which are calibrated and this line of output displays various statistical indicaters which describe the relative success of the calibration.

User Considerations

None

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine REFINE, the subroutine DIRECT performs the following operations:

1. If the value of the parameter NF is one (its default value), the package terminates with a stop code of four.
2. The value of the variable VT is set to the total number of internal trips found from survey data.
3. The subroutine REFINE is called.
4. The subroutine ADJUST is called.
5. If the value of the parameter UT is not zero (its default value), the subroutines CHECK and TEST are called.
6. The subroutine INSERT is called.
7. The values for the parameters in the VALUES namelist are outputed on punched cards.
8. The subroutine PUNCH is called.
9. The logical variable FUTURE is set to .FALSE.
10. If the logical variable PLOT is .TRUE., the subroutine GRAPH is called.
11. The logical variable PLOT is set to .FALSE.
12. The value for each parameter in the VALUES namelist is printed.
13. If the value of the parameter M is greater than the value of the parameter N and N is not zero (its default value), the program is terminated with a stop code of eleven.
14. Control is returned to the program MAIN.

REFINE: This program initializes the curve fits which calculate the coefficients and exponents for the relative trip length model and the interchange limit constraint, and the exponents for the relative attraction and production models. These are used to calculate the relative attraction factors and the relative trip length factors.

NAME: This program prints the name of the routine being executed.

FIT: This program is called for each curve fit. It zeroes out the variables to be used.

ENTER: This program is an entry in the program FIT. Using the variables zeroed in FIT, summations are performed on the data to be fitted.

FORM. This program is another entry in the program FIT. ENTER performs linear curve fit and returns the values of the intercept and slope. Statistical measures are calculated and printed which indicate the success of the fit.

ADJUST: This program adjusts the estimated trip length distributions in an effort to balance the estimated and desired values.

CHECK: This program calls other programs (FIT, ENTER, and FORM) to compare the estimated trip length distribution with the desired distribution.

TEST: This program performs a Kolmogorov-Smirnov test on the estimated trip length distribution versus the desired distribution and prints the results.

INSERT: This program sets the bias factors for the sector movements to 1.0.

PUNCH: This program punches the LENGTH cards.

GRAPH: This program builds arrays to be plotted using the calcomp plotter.

DRAW: This program takes two sets of arrays which were built in GRAPH and calls the appropriate system plot routines which build a data set on the unit PLOTTAPE. PLOTTAPE is the input data set for the calcomp plotter.

Function

The RESTART routine provides the capability of executing additional iterations in the MODEL routine without rerunning previous iterations.

Execution Requirements

RESTART is a dependent routine. It must be preceded by execution of MODEL with DUMP = T in a previous JOB.

Parameter References

Required	Defined	
LIMIT = [5]	MT = MODTRP	NF
DUMP = [T]	MS = NOWSEP	NR
	AN	OMIT
	EXEMPT	ONE
	EXTEND	PN
	FUTURE	SAMPLE
	M	TV
	MR	UT
	N	XP

Data Set References

Input	Output
MS (separation matrix used in previous execution of MODEL)	MODTRP = [3]
RS = [26] (the SV data set outputted from MODEL)	SV = [25] (if DUMP = T)
IMPSEP (if FUTURE = F)	

Data Card References

Output	
FORMAT	
BIAS	(if FUTURE = F)

Operation

The RESTART routine uses the information stored on the RS data set (built by MODEL in a previous JOB) to initialize various parameters and arrays so that the MODEL routine may perform additional iterations without rerunning previous iterations. The RESTART routine then calls the appropriate subroutines within the MODEL routine in order to resume the iterative process.

Printed Output

Three tables of printed output are produced for each additional iteration. These tables are the same tables produced by the MODEL routine for each iteration (i.e., the "Attraction Volume Balance" table, the "Trip Length Balance" table, and the "Sector Interchange Balance" table).

User Considerations

The LIMIT parameter does not specify the number of additional iterations but the total number of iterations. For example, if the MODEL routine had run five iterations in the previous JOB and two additional iterations are desired, then the LIMIT parameter should be set to seven by using the MODIFY routine immediately before the RESTART routine.

The RS data set used as input to the RESTART routine is the SV data set built by the MODEL routine (or the RESTART routine) in the previous JOB. A new SV data set will be built by RESTART after each iteration if

the value of the parameter DUMP is true thus providing the capability of again restarting the process to perform still additional iterations at a later time.

Sequence of Programs Called

		NAME	ADJUST	FIT
MAIN	DIRECT	MODEL	CHECK	ENTER
		SAVE	TEST	FORM

Summary of Individual Programs

DIRECT: In the execution of the routine RESTART, the subroutine DIRECT performs the following operations:

1. Subroutine NAME is called to print the routine NAME.
2. Variables and arrays saved from a previous run of MODEL or RESTART are read from unit RS and the iteration number for the next iteration is set equal to the last iteration run plus one.
3. The VALUES name list is printed.
4. The subroutine VERIFY is called.
5. If the logical variable FUTURE is .TRUE., an attempt is made to read the header record from the data set IMPSEP.
   (a) If IMPSEP is a dummy data set, the unit is rewound.
   (b) If IMPSEP is not a dummy data set, the parameter is set equal to IMPSEP.
6. The subroutine MODEL is called.
7. Parameters and arrays used by subroutine MODEL are saved on unit SV if DUMP is .TRUE., then if the iteration number is

less than LIMIT, subroutine DIRECT goes to Step 6 again.

8. The parameter MT is set equal to the parameter MODTRP.
9. The parameter MS is set equal to the parameter NOWSEP.
10. If the logical variable FUTURE is .FALSE., the subroutine PUNCH is called.
11. Control is returned to the program MAIN.

MODEL: This program performs a trip distribution using a constrained interactance model (see formulation in Section VI - OTHER INFORMATION).

NAME: This program prints the name of the routine being executed.

SAVE: This program writes all arrays of the routine being executed.

ADJUST: This program is used to adjust the relative attraction factors, trip length distributions, travel interactions, and bias factors. These adjustments are performed after each iteration to balance the resulting volumes with the desired volumes.

CHECK: This program calls other programs to determine how well the resulting values match the desired values.

FIT: This program initializes the variables used in the curve fit to zero.

ENTER: This program is an entry in the program FIT. The variables initialized in FIT are used here to perform summations on the desired and resulting trip volumes.

FORM: This program is another entry in program FIT. The variables calculated in ENTER are used to perform a linear curve fit and return the values for intercept and slope. Statistical measures are printed which indicate the success of the fit.

TEST: This program performs a Kolmogorov - Smirnov test on the resulting values versus the desired values. The results are then printed.

Function

The SCREEN routine screens out everything but trip reports from the origin/destination survey data set, and writes another data set with abbreviated trip records containing only the data from the trip reports which are required for the trip distribution.

Execution Requirements

SCREEN is an independent routine. It requires no initialization. It does not affect the key arrays. Either the SORTOUT data set should be protected, or else the BUILD routine should be executed immediately after SCREEN to preserve the sorted trip records.

Parameter References

Required	Defined
OMIT = [F]	N
AMOUNT = [110000]	SIZE

Data Set References

Input	Scratch	Output
REPORT = [12]	NOWTRP = [2]	SORTOUT = 10
	SORTIN = 9	

Data Card References

None

Operation

The SCREEN routine serves as the first step in preparing a trip matrix from the trip reports resulting from an origin/destination survey. It reads the data set of trip reports, screens out all but the data essential for trip distribution, and writes another data set containing only this abbreviated information. Dwelling unit reports and other extraneous information on the trip report data set are disregarded. Trips with external terminals which are reported in the internal survey are disregarded. Volumes of trips merely passing through the study area are divided in half. The passenger's trip purpose, referred to as the secondary trip purpose, is substituted as the purpose of trip for all internal serve passenger trips. Trips having destinations at home are entered twice in the abbreviated data set. This double entry permits future construction of either origin/destination or production/attraction trip matrices. After the end of the trip report data set is reached, the abbreviated trip record data set is sorted. The number of records involved in the sort is indicated by the parameter SIZE. Since external trip reports constitute a large portion of the records involved in the processing, an indicator named OMIT has been provided to omit external trips if they are not desired in any trip matrix. The indicator OMIT simply needs to be set to TRUE.

The SCREEN routine uses the system sorting routines. A parameter named AMOUNT designates the amount of computer storage to be used as a sort work area. This amount may be adjusted to regulate the program region size. It should never be reduced below 40,000 bytes, and larger amounts improve sorting efficiency.

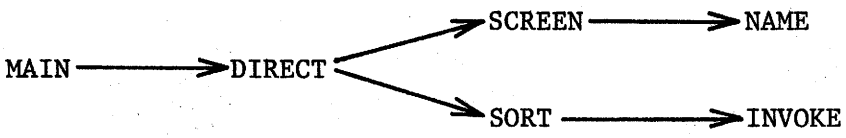
Printed Output

None

User Considerations

If the SCREEN routine is executed without the subsequent execution of the BUILD routine the sorted trip reports are placed on unit 10. This data set should be saved for input into the BUILD routine thereby avoiding an unnecessary execution of the SCREEN routine.

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine SCREEN, the subroutine DIRECT performs the following operations:

1. The subroutine SCREEN is called.
2. The subroutine SORT is called.
3. Control is returned to the program MAIN.

SCREEN: This program reads the survey trip reports from the data set REPORT. As each report is read, the program verifies that it is the type of report desired. The report is then processed into an abbreviated form and written on unit 9.

NAME: This program prints the name of the routine being executed.

SORT: This program calls INVOKE specifying the fields in the records which are to be used as sort keys and the data control block to be used.

INVOKE: This is an assembly language program which calls the system sort routine. The records on unit 9 are then sorted and placed on unit 10.

Function

The SET routine sets arrays with trip generation, trip length, and sector interchange data. It also produces a "flagged" separation matrix which may be used to impose the movements which were found in the survey data upon the trip distribution.

Execution Requirements

SET is an independent routine. It requires no initialization. It prepares key arrays which are used by other routines.

Parameter References

Defined
AN
PN
SAMPLE
TV
MS = NEGSEP (if not DUMMY)

Data Set References

Input	Output
MT = [2] (DD DUMMY optional)	NEGSEP = [1] (DD DUMMY optional)
MS = [4]	

Data Card References

Input

EQUALS (optional)

Operation

The SET routine sets array values for later use in the Trip Distribution Package. The routine first checks to see if sector equivalences have been defined and if not, checks to see if EQUALS cards are available. If EQUALS cards are encountered, they are processed by the EQUATE routine.

If an existing trip matrix is not encountered, the distribution of separations in the separation matrix is initialized and control is returned after printing the parameter namelist.

Printed Output

During the operation of SET several tables are printed. The first of these tables is a Trip Generation Summary which indicates a trip production volume, trip attraction volume, intrazonal trip volume, the number of production and attraction interactions, and the average volume per interaction for each zone. At the end of this table, the total volume over all zones is shown, as well as the number of potential travel interactions among all zone pair combinations which are not eliminated by having zero generations at one terminal.

The next table which is printed exhibits the trip length characteristics for the entire urban area. Each separation interval which exists is shown along with its corresponding zone pair incidence, interaction frequency, trip volume, and other measures calculated from combinations of these parameters. Totals are shown at the bottom of

the table plus the characteristics of trips with external terminals are also summarized at the bottom.

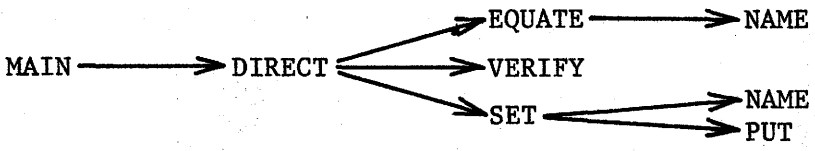
The next four tables summarize the travel characteristics by sector. The first table indicates the number of zone pair combinations which exist among various sectors. The second table indicates the number of sector entries which have travel interactions between the zone pairs. The third table indicates the number of trip interchanges between zone pairs in the sectors. The fourth table indicates the tolerances based on the variance in interchange volumes between the zone pairs within the sectors.

If the trip matrix was prepared by the BUILD program in the Trip Distribution Package, the average sample proportion is calculated. The number of zones having non-zero production and attraction volumes are counted.

User Considerations

In executing the routine SET, the user should note two things. If the data set on unit MT is a dummy data set, a synthetic study is assumed, execution of subroutine SET is bypassed, and the parameter namelist is printed. Also, if the trip matrix on unit MT has not been built by the execution of the routine BUILD, the sampling rate, SAMPLE, will not be calculated and thus will retain its default value of 0.125. In such a case, it would be necessary for the user to run the routine MODIFY to input the sampling rate for the trip matrix being used.

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine SET, the subroutine DIRECT performs the following operations:

1. If the value of the parameters N is zero, NF is one, or M is zero (their default values), the header record on the data set pointed to by the MS parameter is read to define their values and input an array which contains the number of zone pairs at each separation.
2. If the value of the parameter MR is less than or equal to one (its default value), the subroutine EQUATE is called.
3. An attempt is made to read the header record on the data set MT.
   (a) If MT is a dummy data set:
      (1) the unit is rewound
      (2) the value of each parameter in the VALUES namelist is printed
      (3) If the value of the parameter M is greater than the value of the parameter N and N is not zero (its default value), the program terminates with a stop code of eleven
      (4) Control is returned to the program MAIN

(b) If MT is not a dummy data set:

(1) The subroutine VERIFY is called

(2) The subroutine SET is called

(3) Control is returned to the program MAIN.

EQUATE: The primary function of this program is to establish zone to sector equivalences. EQUALS cards containing the sector numbers and the zone numbers which are in those sectors are read and an array is then established containing the zone to sector equivalences. If any zone in the network has not been equivalenced to a sector, that zone is equivalenced to a default sector whose value is one greater than the largest sector number found in the EQUALS cards. If no EQUALS cards are encountered, all zones are equivalenced to sector one.

VERIFY: This program checks the following parameters to determine if their values have exceeded the capacity of this package; N, NF, NR.

SET: This program builds most of the important arrays used in this package. If survey movements are to be imposed in the distribution of the modeled trips, the separations for each zone are read from the data set MS, the separations corresponding to movements which are to be imposed are set to negative, and the separation matrix is written on the data set NEGSEP. If no separation matrix is found on the data set MS (i.e., MS is a dummy data set), all separations are assumed to have a value of one and the logical variable THERE is set to FALSE. The program then proceeds to

read the trips from the data set MT and (if THERE equals TRUE) the separations from the data set MS for each centroid. As the trips and separations (if available) are read for each centroid, arrays are built which indicate the sector movements; the productions, attractions, intrazonal volumes and interactions for each centroid; and the trip volumes and interactions at each separation value.

NAME: This program prints the name of the routine being executed.

PUT: This program takes selected arrays built in set and prints the trip length characteristics for the trip matrix contained in the data set MT.

Function

The SUM routine sums two to five trip matrices.

Execution Requirements

SUM is an independent routine. It requires no initialization. It does not affect key arrays.

Parameter References

Required	Defined
ADDNUM = [2]	MT = SUMTRP
ONE = [1.0]	

Data Set References

Input	Output
ADD1 = [3]	SUMTRP = [20]
ADD2 = [17]	
ADD3 = [18]	
ADD4 = [19]	
ADD5 = [23]	

Data Card References

None

Operation

The SUM routine interrogates the value of parameter ADDNUM to determine how many matrices are to be summed, and it then assumes these

are located sequentially on the add units beginning with ADD1. The sum routine reads each matrix record one at a time, sums the trip volumes, and writes the sum record. Normally, the parameter ONE should have a value of 1.0 as it defaults. Each value in the second and all subsequent trip matrices which are being summed are multiplied by the parameter ONE. This feature is provided to permit factoring up or down a trip matrix during a SUM process.

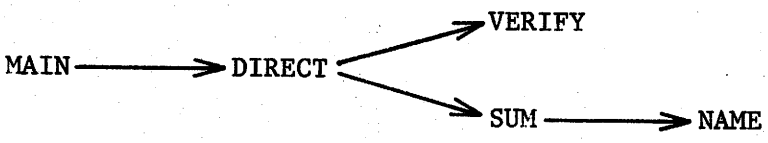
Printed Output

None

User Considerations

The parameter ONE may be set to a value of minus one if it should be desired to subtract one trip matrix from another. Caution should be exercised since a danger exists that negative volumes might result. This feature was originally provided to allow subtracting a trip matrix of only internal movements from a trip matrix containing both internal and external movements to obtain only external movements.

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine SUM, the subroutine DIRECT performs the following operations:

1. It the value of the parameter N is zero (its default value), the header record on the unit ADD1 is read to define its value.
2. The subroutine VERIFY is called.
3. The subroutine SUM is called.
4. The value of the parameter MT is set equal to the parameter SUMTRP.
5. Control is returned to the program MAIN.

VERIFY: This program checks the following parameters to determine if their values have exceeded the capacity of this package; N, NF, NR.

SUM: This program sums from two to five trip matrices as specified by the parameter ADDNUM. The trip matrices to be summed are read one record at a time from the data sets ADD1, ADD2, ADD3, ADD4, or ADD5 (depending upon the number of matrices to be summed) and the trips are multiplied by the parameter ONE. The trips for corresponding zone pairs summed and the summed trips are written on the data set SUMTRP.

NAME: This program prints the name of the routine being executed.

Function

The SWITCH routine switches a production/attraction trip matrix to an origin/destination trip matrix.

Execution Requirements

SWITCH is an independent routine. It requires no initialization. It does not affect key arrays.

Parameter References

Required	Defined
AMOUNT = [110000]	MT = SWTTRP
	SIZE

Data Set References

Input	Scratch	Output
MT = [2]	SORTIN = 9	SWTTRP = [24]
	SORTOUT = 10	

Data Card References

None

Operation

The SWITCH routine reads a trip matrix, divides each interchange volume by two, writes each non-zero half volume on a sort input data set. The data set of half volumes is sorted. The trip matrix is reread (again

dividing volumes by two) and the half volumes from the trip matrix and the sorted data set are merged and the output trip matrix is written on the SWTTRP data set.

The SIZE parameter reflects a count of the number of sort records to be sorted. The amount parameter controls the amount of core storage which is allocated for sort usage.

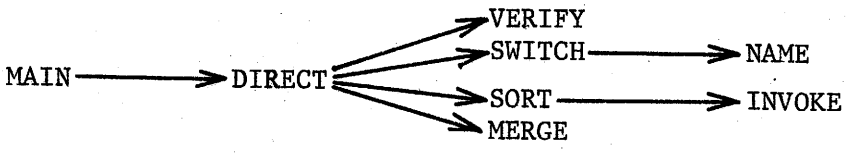
Printed Output

None

User Considerations

Due to the presence of the sort, the SWITCH routine consumes a sizeable amount of computer time and computer storage. Therefore, it is advisable not to execute it more than is absolutely necessary.

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine SWITCH, the subroutine DIRECT performs the following operations:

1. If the value of the parameter N is zero (its default value), the header record on the data set MT is read to define its value.
2. The subroutine VERIFY is called.

3. The subroutine SWITCH is called.
4. The subroutine SORT is called.
5. The subroutine MERGE is called.
6. The value of the parameter MT is set equal to the value of the parameter SWTTRP.
7. Control is returned to the program MAIN.

VERIFY: This program checks the following parameters to determine if their values have exceeded the capacity of this package; N, NF, NR.

SWITCH: This program reads the trip volumes for each centroid from the data set MT. Each trip volume is divided by two and a record is written on unit 9 which contains the attraction zone number, the production zone number, and the half volume.

SORT: This program specifies the fields in the records which are to be used as sort keys (the attraction zone number is used as the primary sort key and the production zone is used as the secondary sort key) and the data control block to be used.

INVOKE: This is an assembly language program which calls the system sort routine. The records on unit 9 are then sorted and placed on unit 10.

MERGE: This program is an entry in the program SWITCH. It reads the trip volumes for each centroid off the data set MT and converts each to half of its original amount. The trip volumes for the same centroid are then read off unit 10 and summed with the half

volumes just computed. The result is that each centroid will have an equal number of origins and destinations. The origin and destination trip matrix is written on the data set SWTTRP.

Function

The UNPACK routine reformats a trip matrix from the format used in the Texas Large Network Package and Texas Small Network Package and writes the trip matrix in the format used by this package.

Execution Requirements

UNPACK is an independent routine. It requires no initialization. It does not affect the key arrays.

Parameter References

Required

M

TYPE = [blank]

Data Set References

Input	Output
INTRIP = [21]	NOWTRP = [2]

Data Card References

None

Operation

The UNPACK routine simply reads each record from a trip table in the format used by the Texas Large Network Package and the Texas Small Network Package, reformats each record for use by this package, and writes the reformatted records on the NOWTRP data set.

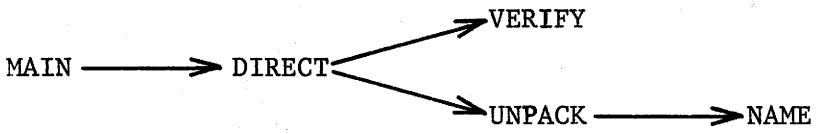
Printed Output

None

User Considerations

None

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine UNPACK, the subroutine DIRECT performs the following operations:

1. The data set INTRIP is rewound.
2. The header record on the data set INTRIP is read to define the value of the parameter N.
3. The subroutine VERIFY is called.
4. The subroutine UNPACK is called.
5. Control is returned to the program MAIN

VERIFY: This program checks the following parameters to determine if their values have exceeded the capacity of this package; N, NF, NR.

UNPACK: This program reads a trip matrix from the data set INTRIP which was built by the Large or Small Assignment Packages. As each record is read, it is reformated from the "packed" format of the Large and Small Assignment Packages to the format used by the

Trip Distribution Package and the reformated record is written on the data set NOWTRP.

NAME: This program prints the name of the routine being executed.

Function

The WRITE routine prints the separation matrix.

Execution Requirements

WRITE is an independent routine. It requires no initialization. It does not affect any key arrays.

Parameter References

None

Data Set References

Input

MS

Data Card References

None

Operation

The WRITE routine is used to print the separation matrix. Each zone is treated individually and within each zone each separation value is treated individually. Starting with the separation value of one, all destination zones having this particular separation will be listed in a string. Then the next separation value will be treated.

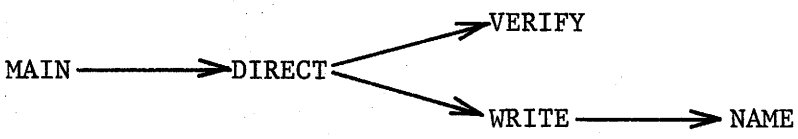
Printed Output

WRITE outputs the separation matrix contained on the MS data set. This is done in tabular form for each centroid and external station.

User Consideration

Due to the execution time required and the amount of printed output (a minimum of one page for each centroid and external station), the WRITE routine should be used only when necessary.

Sequence of Programs Called



Summary of Individual Programs

DIRECT: In the execution of the routine WRITE, the subroutine DIRECT performs the following operations:

1. If the values of the parameters N is zero or NF is 1, the header record on the data set MS is read to define their values.
2. The subroutine VERIFY is called.
3. The subroutine WRITE is called.
4. Control is returned to the program MAIN.

VERIFY: This program checks the following parameters to determine if their values have exceeded the capacity of this package; N, NF, NR.

WRITE: This program reads the separations for a centroid I from the data set MS. Each separation is then printed along with all the centroids having that separation from centroid I. The process is repeated for each centroid.

NAME: This program prints the name of the routine being executed.

ABNORMAL TERMINATIONS

The following table lists the stop codes which may be encountered, the locations in which they may be encountered, and the probable cause.

Stop Code	Location	Cause
0	DIRECT	Normal termination (STOP instruction encountered)
1	SCAN	Invalid CONTROL entry encountered.
2	EDIT	Fewer centroids in RAWSEP than in trip reports.
3	ACCEPT	GENERATION cards incomplete; if none listed, no FORMAT identification matched TYPE sought.
4	REFINE	Array values not prepared by SET.
5	ACCEPT	FORMAT card for GENERATIONS not encountered.
6	IMPOSE	ADMIT cards not encountered.
7	VERIFY	Array lengths exceed specified capacities.
8	BUILD	CATEGORY card not encountered.
9	MATCH	Sector combinations must be set at 320 or more when executing MATCH.
10	MAIN	The number of routines specified for execution exceeded the capacity of 40.
11	DIRECT	M exceeds N.
12	PREVUE	Invalid &VALUES card encountered.
13	ACCEPT and REFINE	FORMAT card for INTERACTION cards not encountered when INTERACTION cards are present.
14	ACCEPT and REFINE	Production volumes out of sort on INTERACTION cards.

Stop Code	Location	Cause
15	MODEL	One or more separation values were encountered at which there are no eligible zone pairs while the expected trip volume at each of these separations were greater than one percent of the total trips.
16	EDIT	Either special separation code has been encountered in SEPARATION cards which is greater than the value of EXTEND plus one or a SEPARATION card has been encountered which has both a replacement separation value and a special separation code.
17	RESTART	The restart data set is not complete or a data error occurred in reading it.

F L O W C H A R T S

INTRODUCTION

FLOWCHARTS OF CONTROL SECTIONS

INTRODUCTION

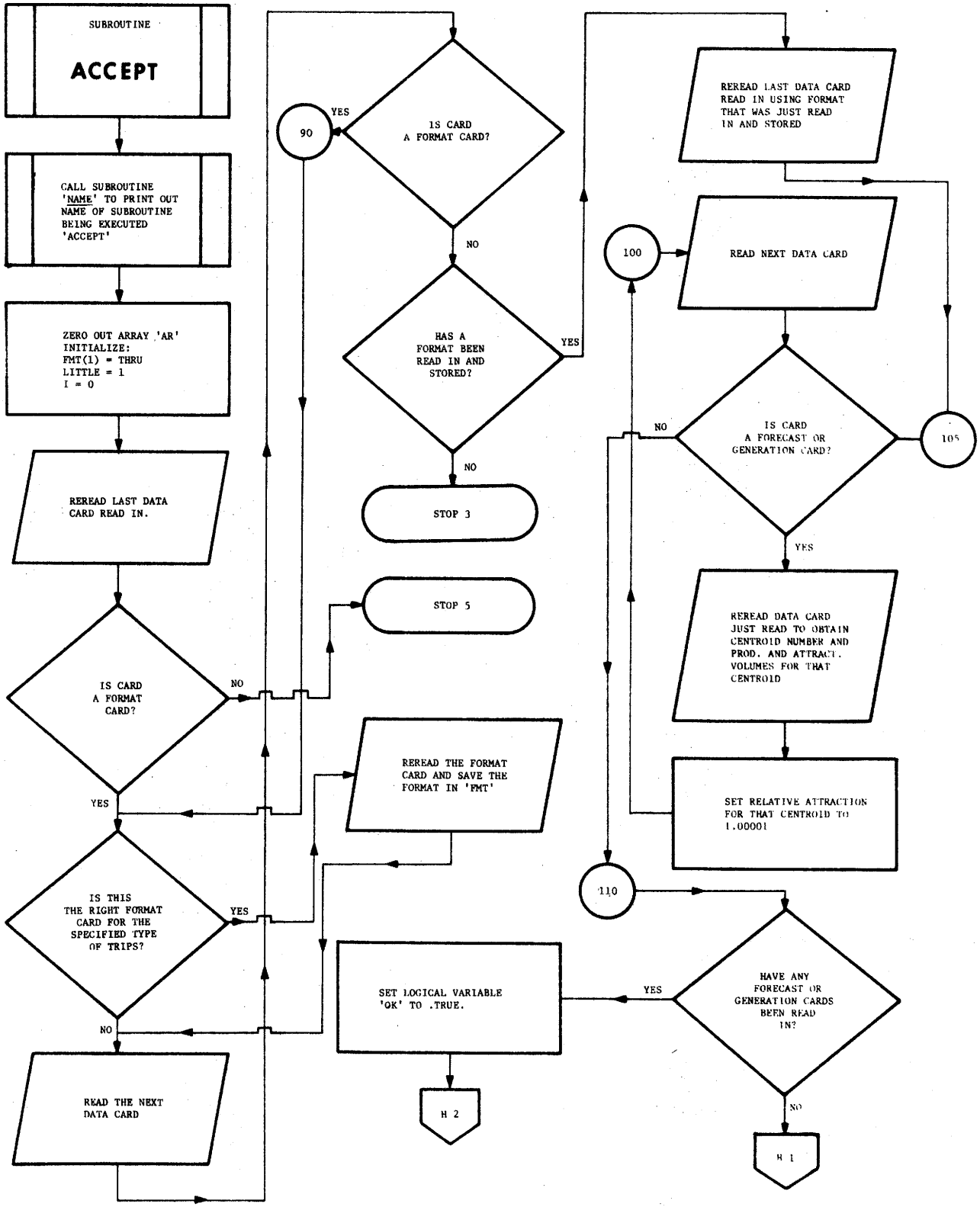
The flowcharts for each of the control sections developed for the package (except the BLOCK DATA subprogram) are contained in this section. No flowcharts have been provided for the system subroutines used by the package. The flowcharts vary from one to ten pages in length depending on the length and complexity of the programs. Likewise, level of detail varies depending on the complexity and significance of the program.

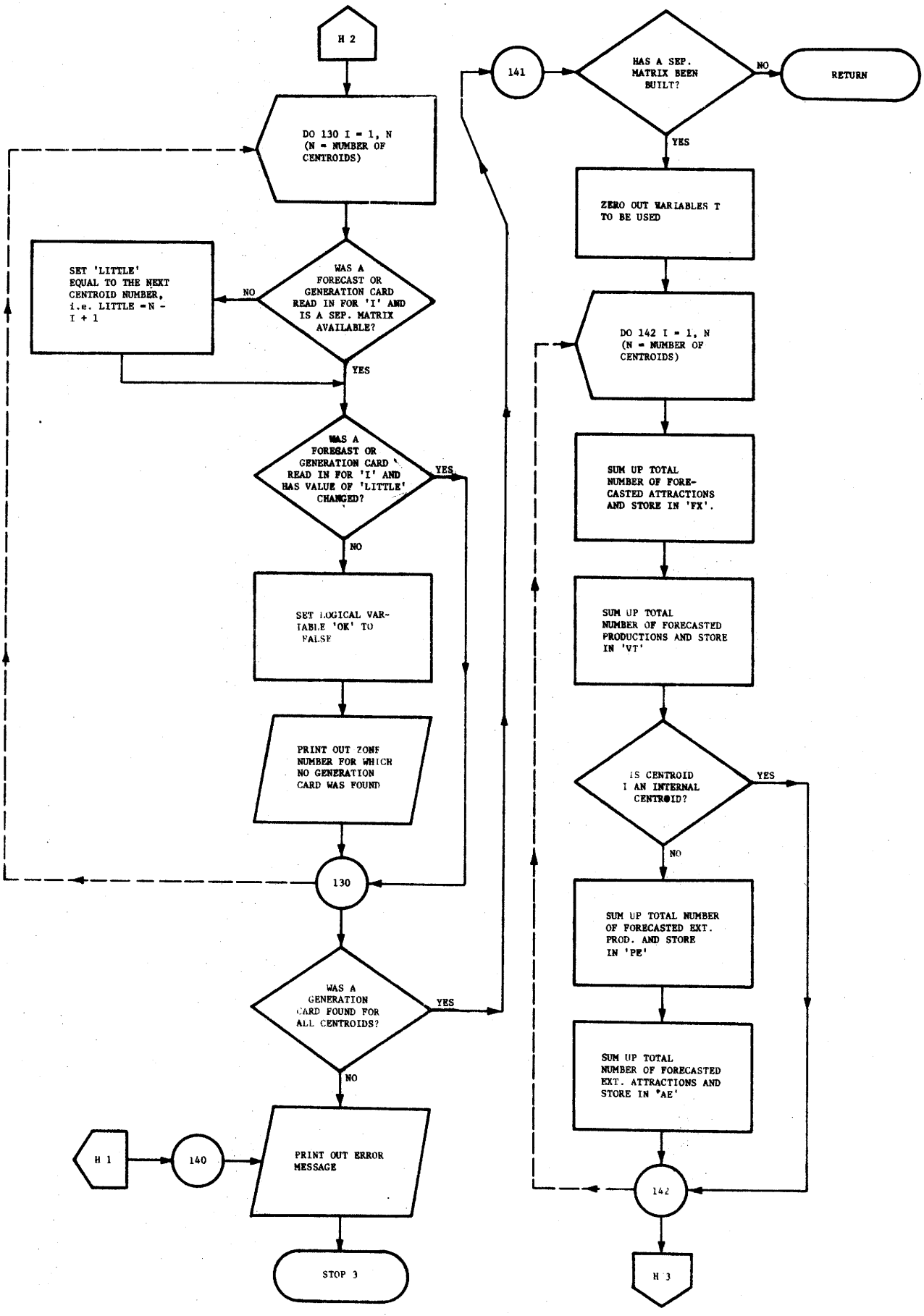
The flowcharts are program flowcharts as opposed to broad generalized logic flowcharts which may bear very little resemblance to the coded program. In all cases, the level of detail should be at least sufficient so that the data processing programmer can conveniently follow the program coding when studying a particular program listing.

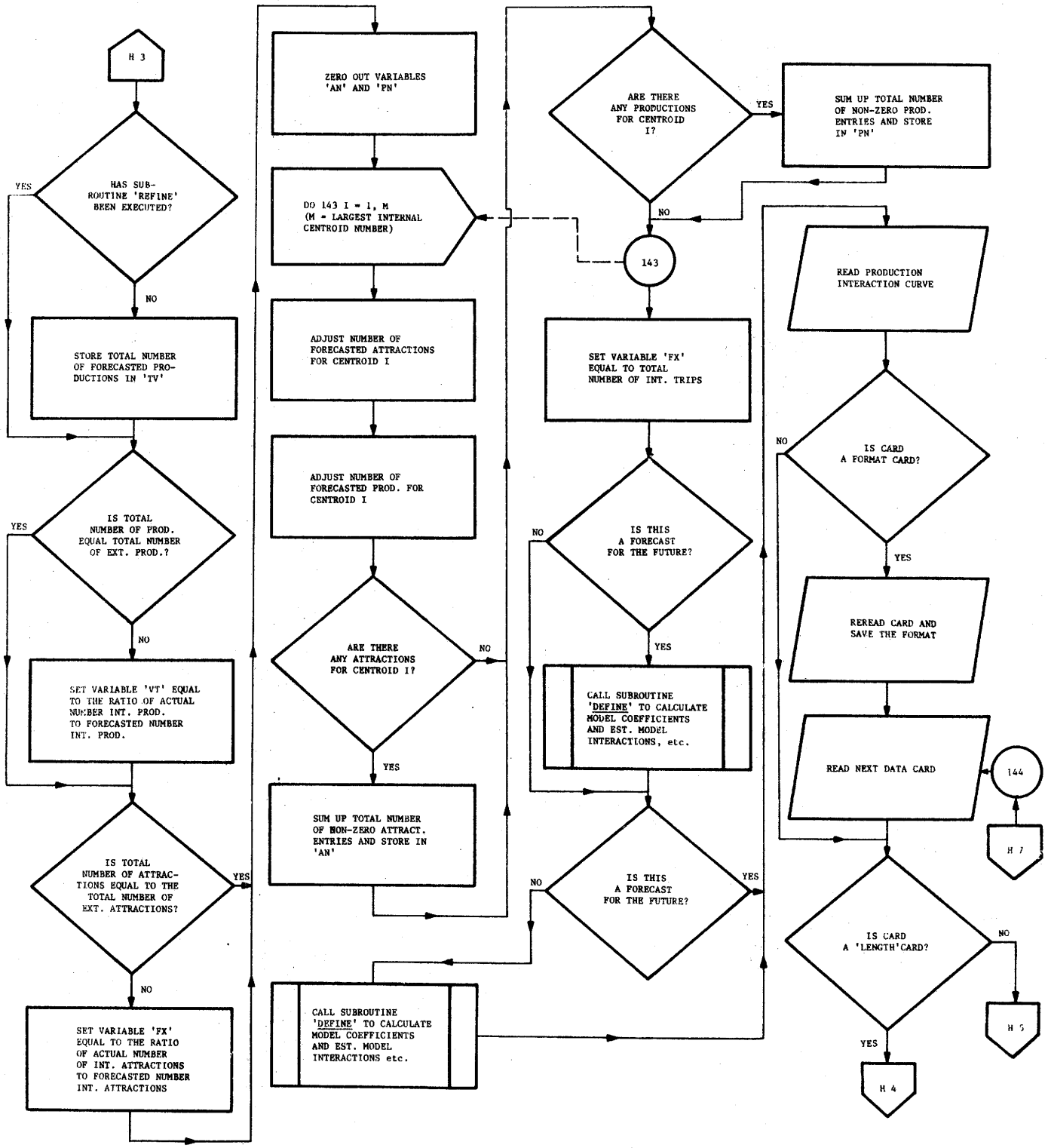
The flowcharts are arranged in alphabetical order with respect to the name of the subroutine. For convenience, the name of the subroutine along with the name of any other entry points in the subroutine have been printed in the upper right hand corner of every page.

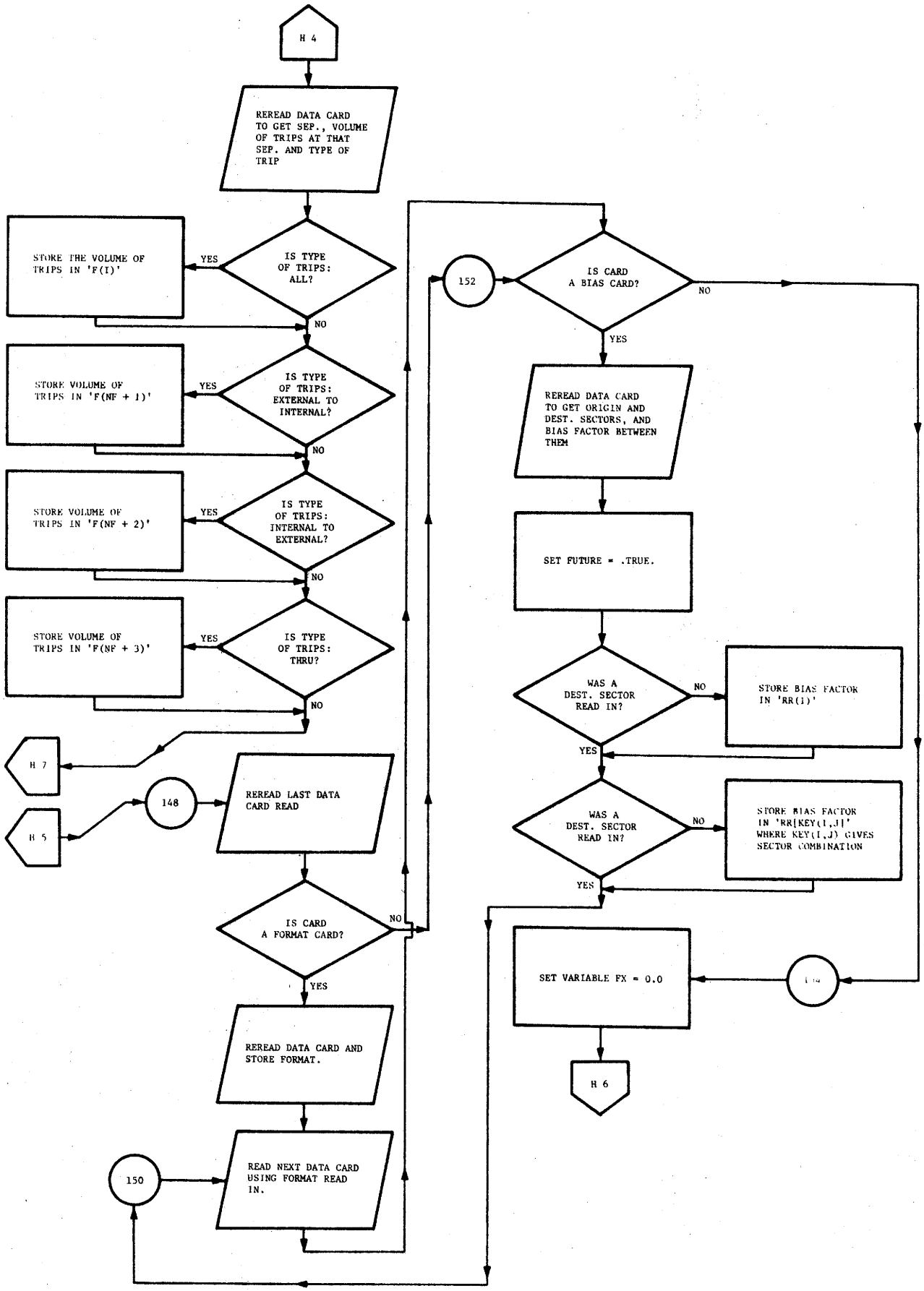
FLOWCHARTS OF CONTROL SECTIONS

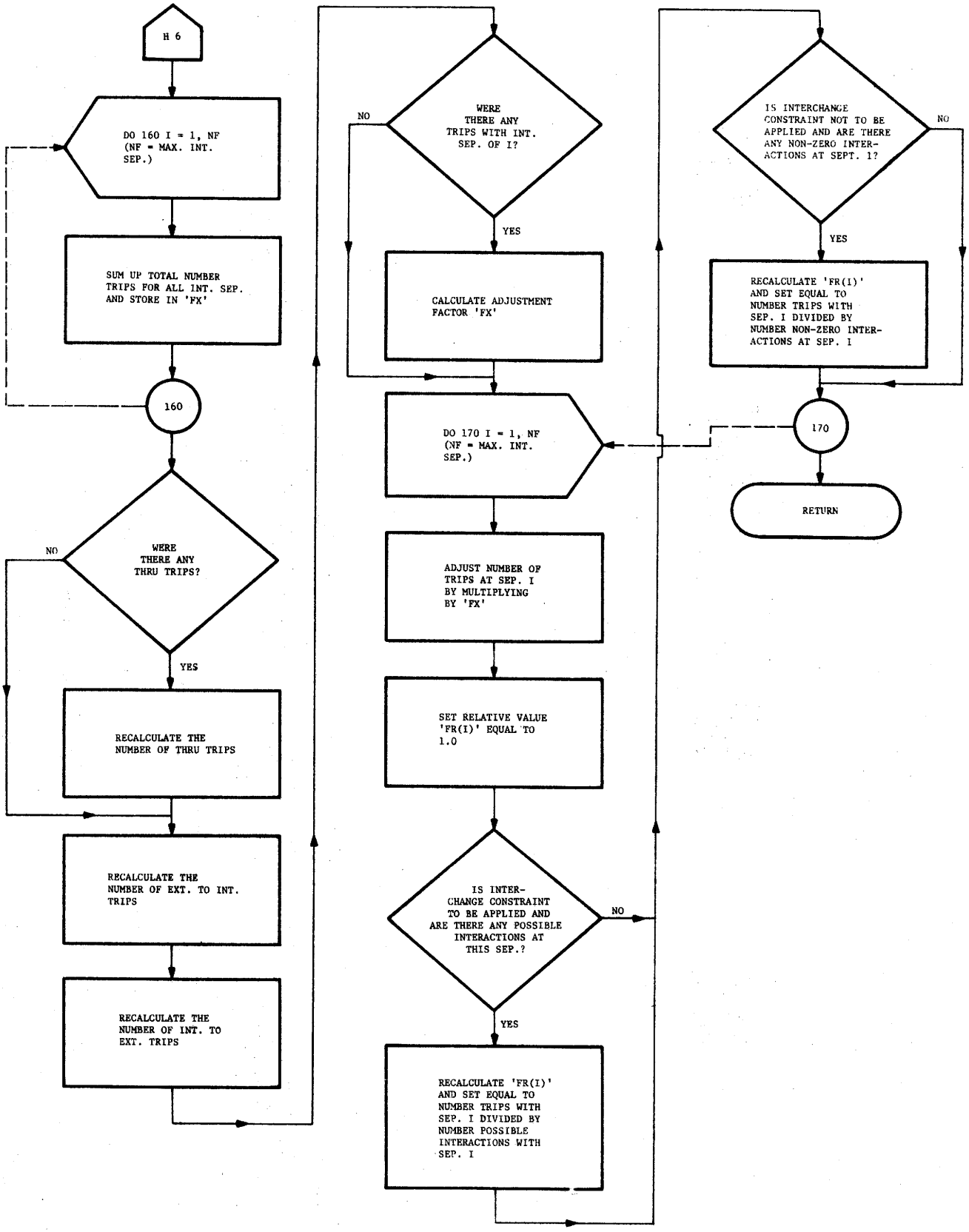
The following are the flowcharts of the 50 control sections comprising the Texas Trip Distribution Package.

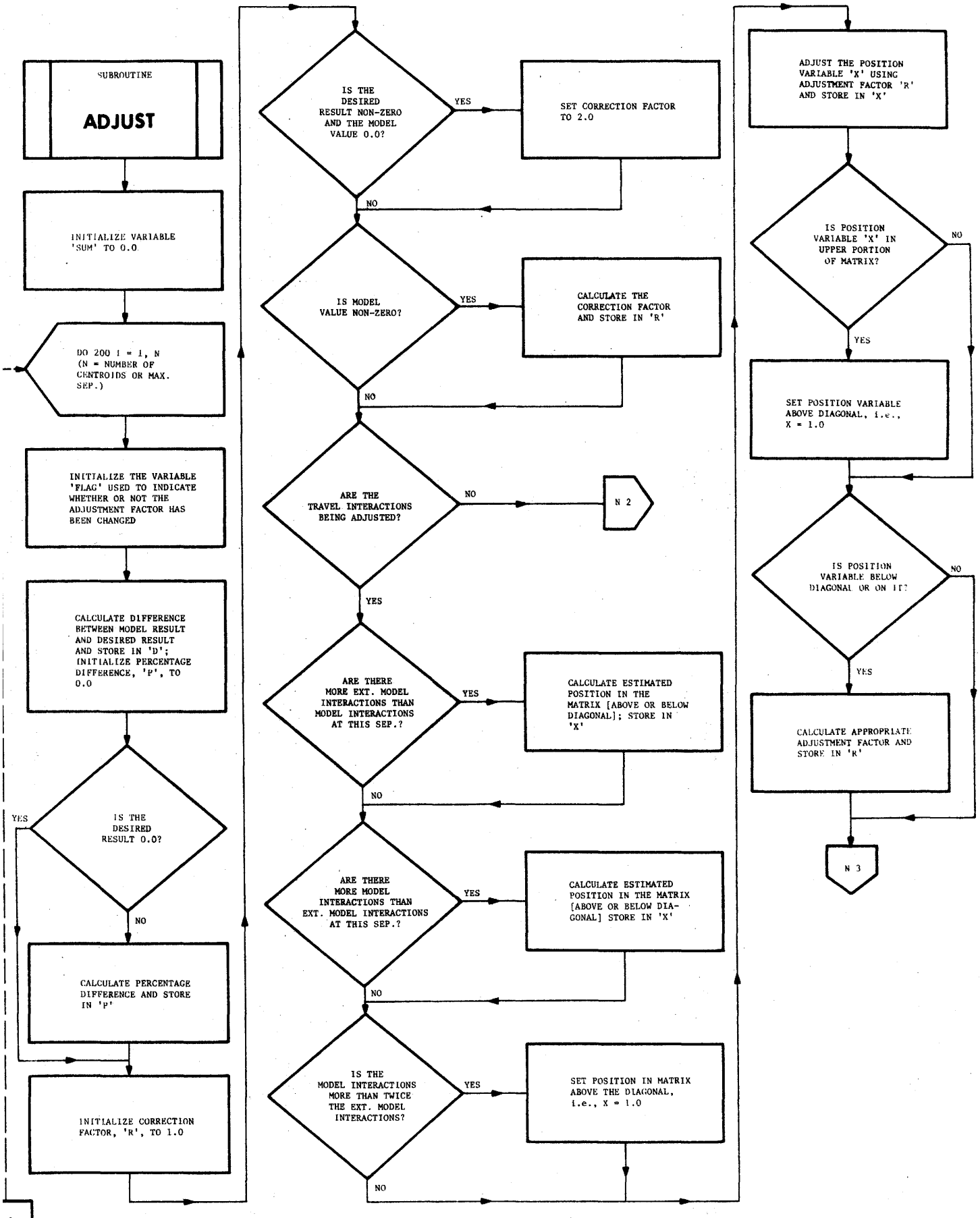

SUBROUTINE
ACCEPT
CALL SUBROUTINE 'NAME' TO PRINT OUT NAME OF SUBROUTINE BEING EXECUTED 'ACCEPT'
ZERO OUT ARRAY 'AR' INITIALIZE: FMT(1) = THRU LITTLE = 1 I = 0
REREAD LAST DATA CARD READ IN.
IS CARD A FORMAT CARD?
NO
YES
STOP 5
IS THIS THE RIGHT FORMAT CARD FOR THE SPECIFIED TYPE OF TRIPS?
YES
NO
REREAD THE FORMAT CARD AND SAVE THE FORMAT IN 'FMT'
READ THE NEXT DATA CARD
90
YES
IS CARD A FORMAT CARD?
NO
HAS A FORMAT BEEN READ IN AND STORED?
YES
NO
STOP 3
SET LOGICAL VARIABLE 'OK' TO .TRUE.
H 2
REREAD LAST DATA CARD READ IN USING FORMAT THAT WAS JUST READ IN AND STORED
100
READ NEXT DATA CARD
IS CARD A FORECAST OR GENERATION CARD?
NO
105
YES
REREAD DATA CARD JUST READ TO OBTAIN CENTROID NUMBER AND PROD. AND ATTRACT. VOLUMES FOR THAT CENTROID
SET RELATIVE ATTRACTION FOR THAT CENTROID TO 1.00001
110
HAVE ANY FORECAST OR GENERATION CARDS BEEN READ IN?
YES
NO
H 1

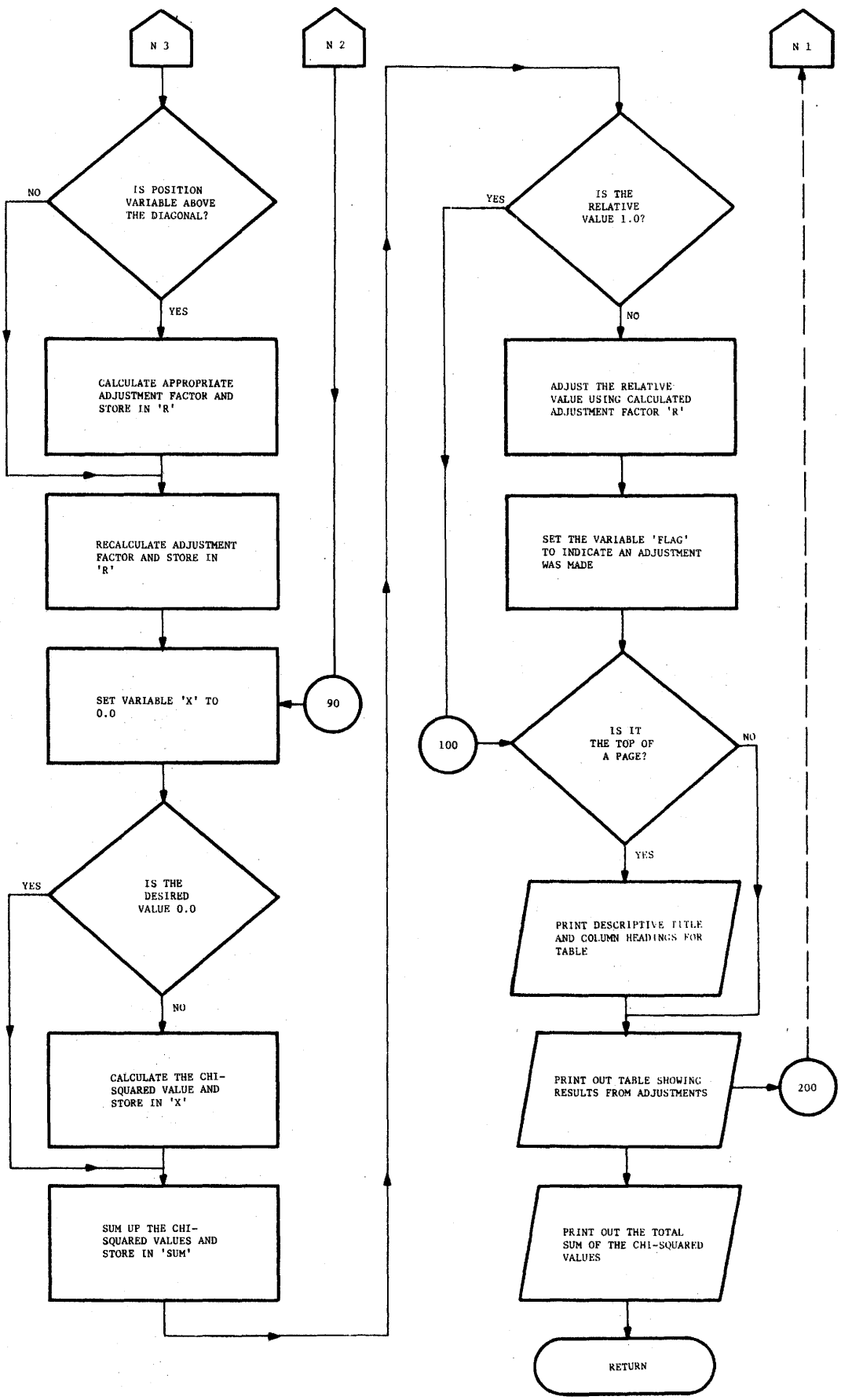

H 2
DO 130 I = 1, N (N = NUMBER OF CENTROIDS)
WAS A FORECAST OR GENERATION CARD READ IN FOR 'I' AND IS A SEP. MATRIX AVAILABLE?
NO
SET 'LITTLE' EQUAL TO THE NEXT CENTROID NUMBER, i.e. LITTLE = N - I + 1
YES
WAS A FORECAST OR GENERATION CARD READ IN FOR 'I' AND HAS VALUE OF 'LITTLE' CHANGED?
YES
NO
SET LOGICAL VARIABLE 'OK' TO FALSE
PRINT OUT ZONE NUMBER FOR WHICH NO GENERATION CARD WAS FOUND
130
WAS A GENERATION CARD FOUND FOR ALL CENTROIDS?
YES
NO
H 1
140
PRINT OUT ERROR MESSAGE
STOP 3
141
HAS A SEP. MATRIX BEEN BUILT?
NO
RETURN
YES
ZERO OUT VARIABLES T TO BE USED
DO 142 I = 1, N (N = NUMBER OF CENTROIDS)
SUM UP TOTAL NUMBER OF FORECASTED ATTRACTIONS AND STORE IN 'FX'.
SUM UP TOTAL NUMBER OF FORECASTED PRODUCTIONS AND STORE IN 'VT'
IS CENTROID I AN INTERNAL CENTROID?
YES
NO
SUM UP TOTAL NUMBER OF FORECASTED EXT. PROD. AND STORE IN 'PE'
SUM UP TOTAL NUMBER OF FORECASTED EXT. ATTRACTIONS AND STORE IN 'AE'
142
H 3

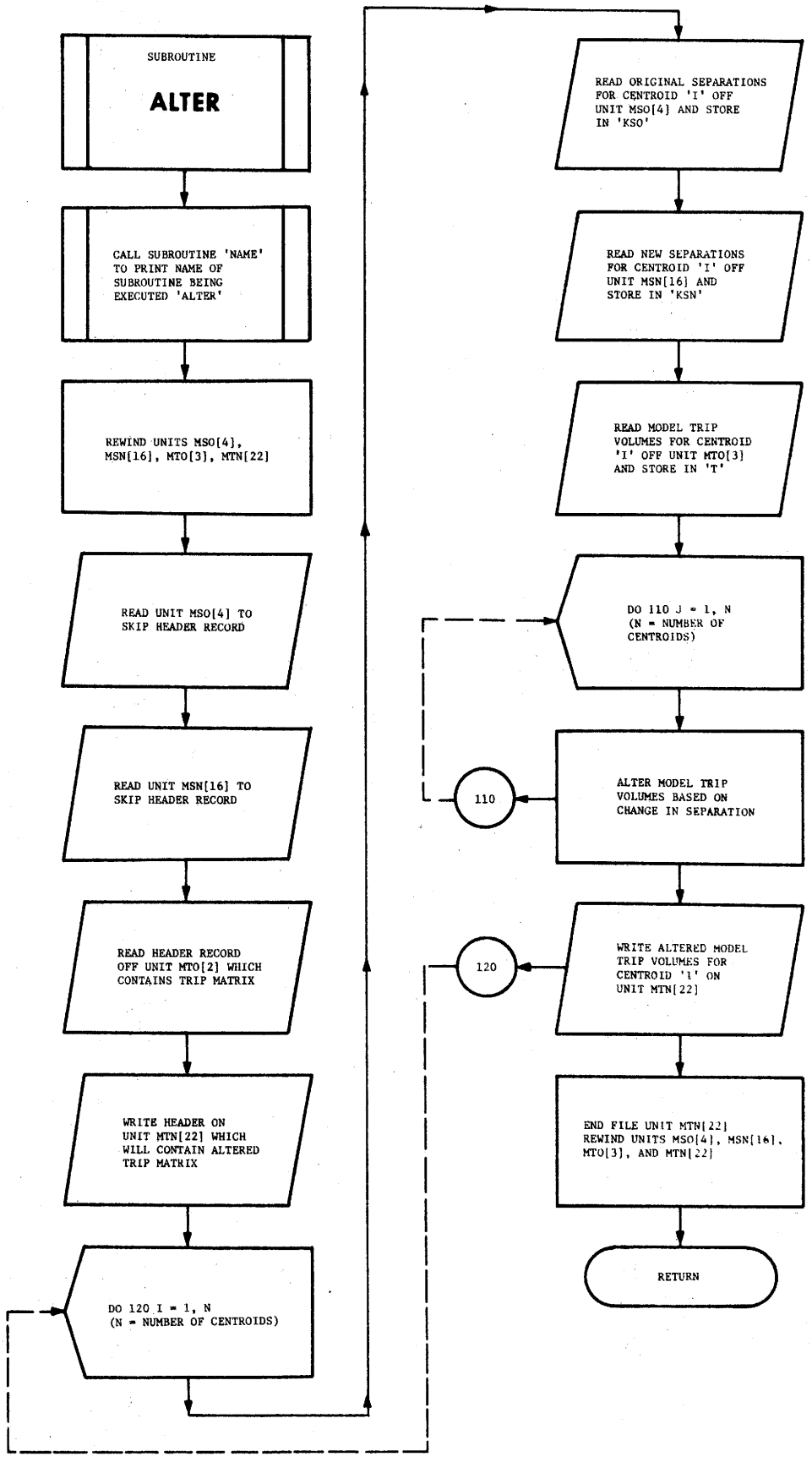

H 3
HAS SUB-ROUTINE 'REFINE' BEEN EXECUTED?
YES
NO
STORE TOTAL NUMBER OF FORECASTED PRO-DUCTIONS IN 'TV'
IS TOTAL NUMBER OF PROD. EQUAL TOTAL NUMBER OF EXT. PROD.?
YES
NO
SET VARIABLE 'VT' EQUAL TO THE RATIO OF ACTUAL NUMBER INT. PROD. TO FORECASTED NUMBER INT. PROD.
IS TOTAL NUMBER OF ATTRAC-TIONS EQUAL TO THE TOTAL NUMBER OF EXT. ATTRACTIONS?
YES
NO
SET VARIABLE 'FX' EQUAL TO THE RATIO OF ACTUAL NUMBER OF INT. ATTRACTIONS TO FORECASTED NUMBER INT. ATTRACTIONS
ZERO OUT VARIABLES 'AN' AND 'PN'
DO 143 I = 1, M (M = LARGEST INTERNAL CENTROID NUMBER)
ADJUST NUMBER OF FORECASTED ATTRACTIONS FOR CENTROID I
ADJUST NUMBER OF FORECASTED PROD. FOR CENTROID I
ARE THERE ANY ATTRACTIONS FOR CENTROID I?
NO
YES
SUM UP TOTAL NUMBER OF NON-ZERO ATTRACT. ENTRIES AND STORE IN 'AN'
CALL SUBROUTINE 'DEFINE' TO CALCULATE MODEL COEFFICIENTS AND EST. MODEL INTERACTIONS etc.
ARE THERE ANY PRODUCTIONS FOR CENTROID I?
YES
NO
SUM UP TOTAL NUMBER OF NON-ZERO PROD. ENTRIES AND STORE IN 'PN'
143
SET VARIABLE 'FX' EQUAL TO TOTAL NUMBER OF INT. TRIPS
IS THIS A FORECAST FOR THE FUTURE?
NO
YES
CALL SUBROUTINE 'DEFINE' TO CALCULATE MODEL COEFFICIENTS AND EST. MODEL INTERACTIONS, etc.
IS THIS A FORECAST FOR THE FUTURE?
NO
YES
READ PRODUCTION INTERACTION CURVE
IS CARD A FORMAT CARD?
NO
YES
REREAD CARD AND SAVE THE FORMAT
READ NEXT DATA CARD
144
H 7
IS CARD A 'LENGTH' CARD?
NO
YES
H 5
H 4

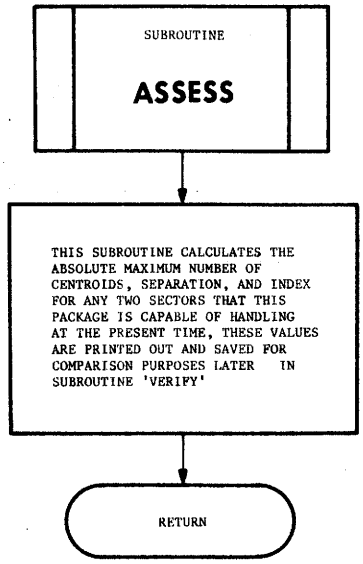

H 4
REREAD DATA CARD TO GET SEP., VOLUME OF TRIPS AT THAT SEP. AND TYPE OF TRIP
IS TYPE OF TRIPS: ALL?
YES
STORE THE VOLUME OF TRIPS IN 'F(I)'
NO
IS TYPE OF TRIPS: EXTERNAL TO INTERNAL?
YES
STORE VOLUME OF TRIPS IN 'F(NF + 1)'
NO
IS TYPE OF TRIPS: INTERNAL TO EXTERNAL?
YES
STORE VOLUME OF TRIPS IN 'F(NF + 2)'
NO
IS TYPE OF TRIPS: THRU?
YES
STORE VOLUME OF TRIPS IN 'F(NF + 3)'
NO
H 7
H 5
148
REREAD LAST DATA CARD READ
IS CARD A FORMAT CARD?
NO
YES
REREAD DATA CARD AND STORE FORMAT.
150
READ NEXT DATA CARD USING FORMAT READ IN.
152
IS CARD A BIAS CARD?
NO
YES
REREAD DATA CARD TO GET ORIGIN AND DEST. SECTORS, AND BIAS FACTOR BETWEEN THEM
SET FUTURE = .TRUE.
WAS A DEST. SECTOR READ IN?
NO
STORE BIAS FACTOR IN 'RR(I)'
YES
WAS A DEST. SECTOR READ IN?
NO
STORE BIAS FACTOR IN 'RR[KEY(I,J)' WHERE KEY(I,J) GIVES SECTOR COMBINATION
YES
SET VARIABLE FX = 0.0
H 6

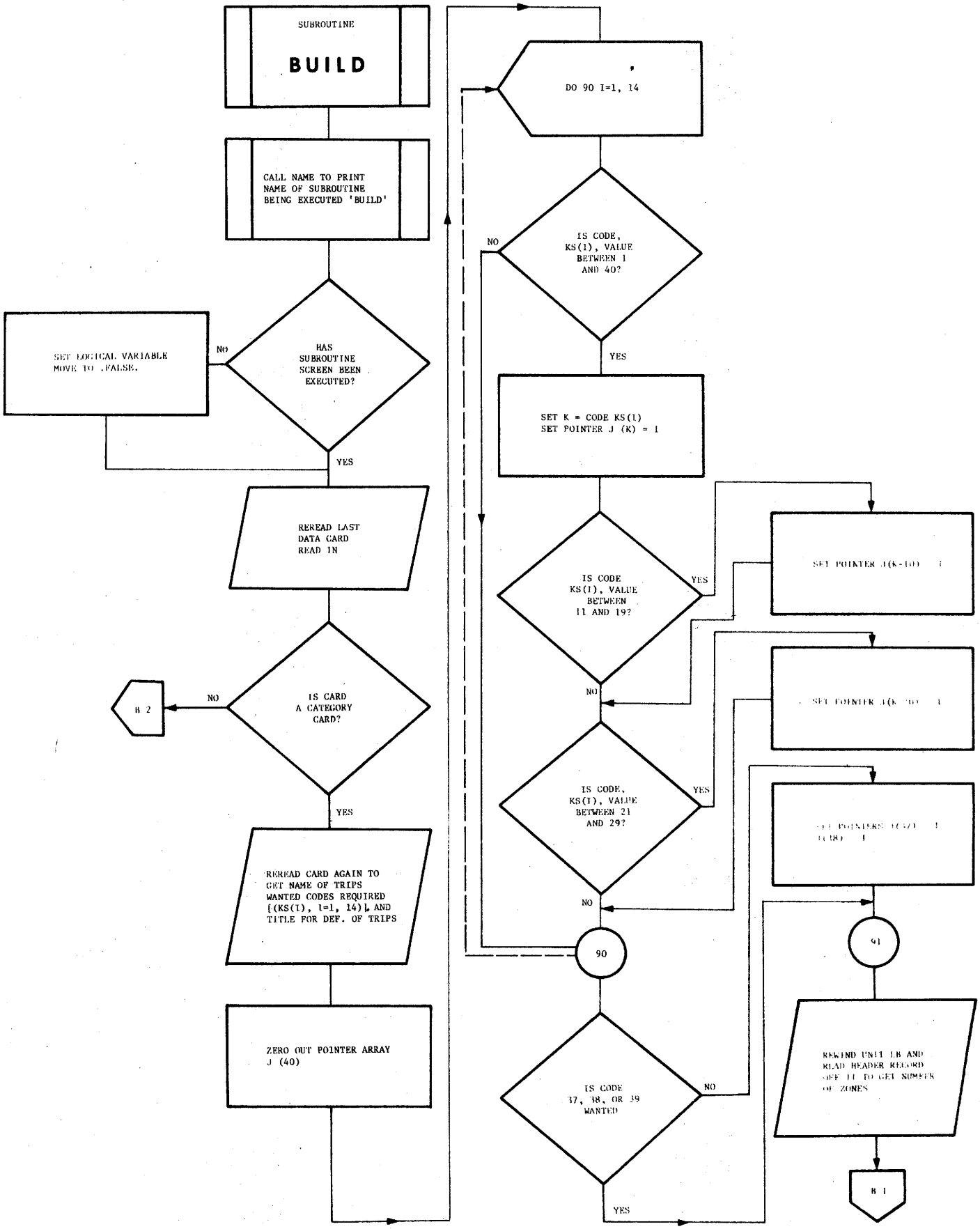

H 6
DO 160 I = 1, NF (NF = MAX. INT. SEP.)
SUM UP TOTAL NUMBER TRIPS FOR ALL INT. SEP. AND STORE IN 'FX'
160
WERE THERE ANY THRU TRIPS?
NO
YES
RECALCULATE THE NUMBER OF THRU TRIPS
RECALCULATE THE NUMBER OF EXT. TO INT. TRIPS
RECALCULATE THE NUMBER OF INT. TO EXT. TRIPS
WERE THERE ANY TRIPS WITH INT. SEP. OF I?
NO
YES
CALCULATE ADJUSTMENT FACTOR 'FX'
DO 170 I = 1, NF (NF = MAX. INT. SEP.)
ADJUST NUMBER OF TRIPS AT SEP. I BY MULTIPLYING BY 'FX'
SET RELATIVE VALUE 'FR(I)' EQUAL TO 1.0
IS INTER-CHANGE CONSTRAINT TO BE APPLIED AND ARE THERE ANY POSSIBLE INTERACTIONS AT THIS SEP.?
NO
YES
RECALCULATE 'FR(I)' AND SET EQUAL TO NUMBER TRIPS WITH SEP. I DIVIDED BY NUMBER POSSIBLE INTERACTIONS WITH SEP. I
IS INTERCHANGE CONSTRAINT NOT TO BE APPLIED AND ARE THERE ANY NON-ZERO INTER-ACTIONS AT SEPT. 1?
NO
YES
RECALCULATE 'FR(I)' AND SET EQUAL TO NUMBER TRIPS WITH SEP. I DIVIDED BY NUMBER NON-ZERO INTER-ACTIONS AT SEP. I
170
RETURN

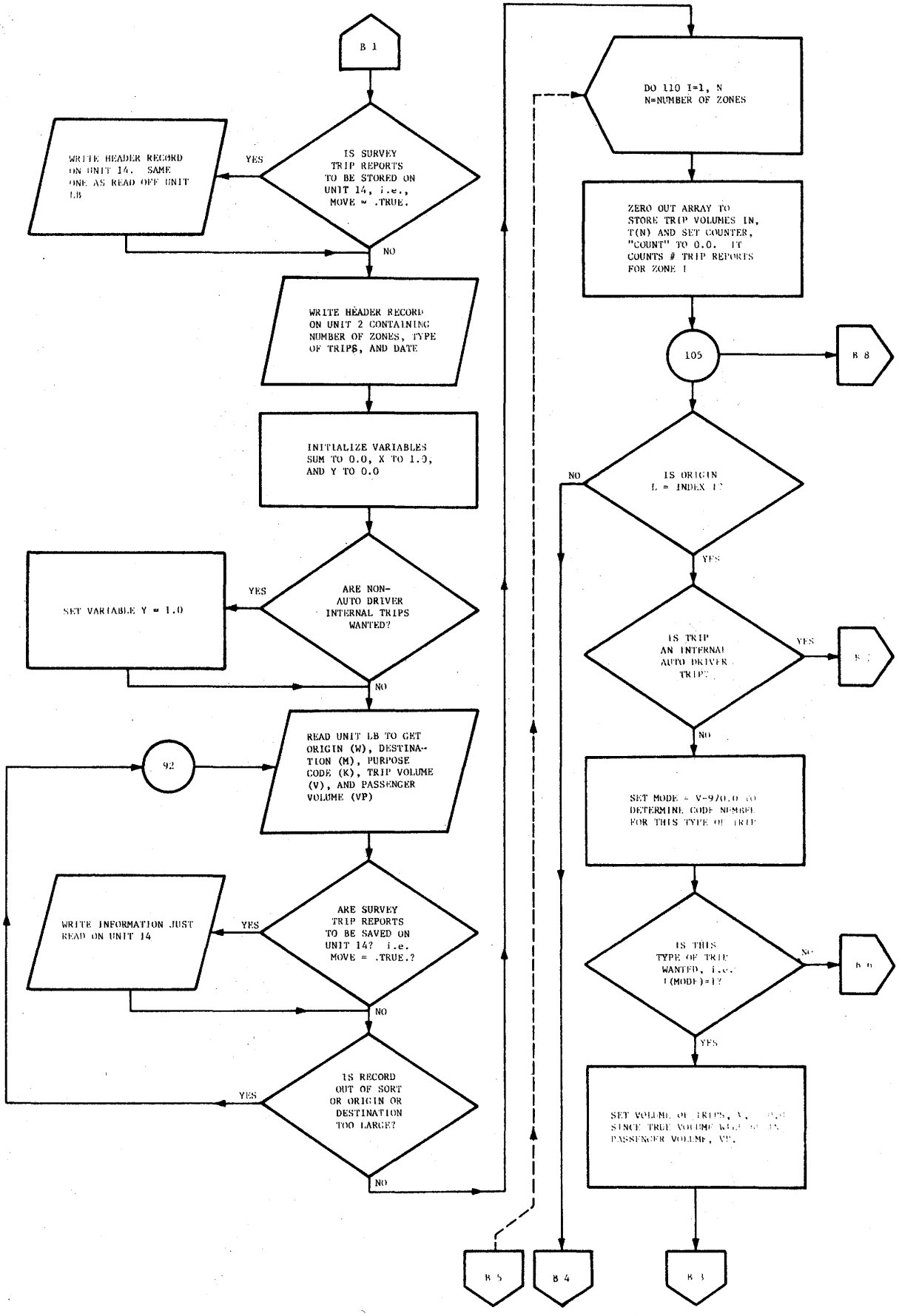

SUBROUTINE
ADJUST
INITIALIZE VARIABLE 'SUM' TO 0.0
DO 200 I = 1, N (N = NUMBER OF CENTROIDS OR MAX. SEP.)
INITIALIZE THE VARIABLE 'FLAG' USED TO INDICATE WHETHER OR NOT THE ADJUSTMENT FACTOR HAS BEEN CHANGED
CALCULATE DIFFERENCE BETWEEN MODEL RESULT AND DESIRED RESULT AND STORE IN 'D'; INITIALIZE PERCENTAGE DIFFERENCE, 'P', TO 0.0
IS THE DESIRED RESULT 0.0?
YES
NO
CALCULATE PERCENTAGE DIFFERENCE AND STORE IN 'P'
INITIALIZE CORRECTION FACTOR, 'R', TO 1.0
1
IS THE DESIRED RESULT NON-ZERO AND THE MODEL VALUE 0.0?
YES
SET CORRECTION FACTOR TO 2.0
NO
IS MODEL VALUE NON-ZERO?
YES
CALCULATE THE CORRECTION FACTOR AND STORE IN 'R'
NO
ARE THE TRAVEL INTERACTIONS BEING ADJUSTED?
NO
N 2
YES
ARE THERE MORE EXT. MODEL INTERACTIONS THAN MODEL INTERACTIONS AT THIS SEP.?
YES
CALCULATE ESTIMATED POSITION IN THE MATRIX [ABOVE OR BELOW DIAGONAL]; STORE IN 'X'
NO
ARE THERE MORE MODEL INTERACTIONS THAN EXT. MODEL INTERACTIONS AT THIS SEP.?
YES
CALCULATE ESTIMATED POSITION IN THE MATRIX [ABOVE OR BELOW DIAGONAL] STORE IN 'X'
NO
IS THE MODEL INTERACTIONS MORE THAN TWICE THE EXT. MODEL INTERACTIONS?
YES
SET POSITION IN MATRIX ABOVE THE DIAGONAL, i.e., X = 1.0
NO
ADJUST THE POSITION VARIABLE 'X' USING ADJUSTMENT FACTOR 'R' AND STORE IN 'X'
IS POSITION VARIABLE 'X' IN UPPER PORTION OF MATRIX?
NO
YES
SET POSITION VARIABLE ABOVE DIAGONAL, i.e., X = 1.0
IS POSITION VARIABLE BELOW DIAGONAL OR ON IT?
NO
YES
CALCULATE APPROPRIATE ADJUSTMENT FACTOR AND STORE IN 'R'
N 3

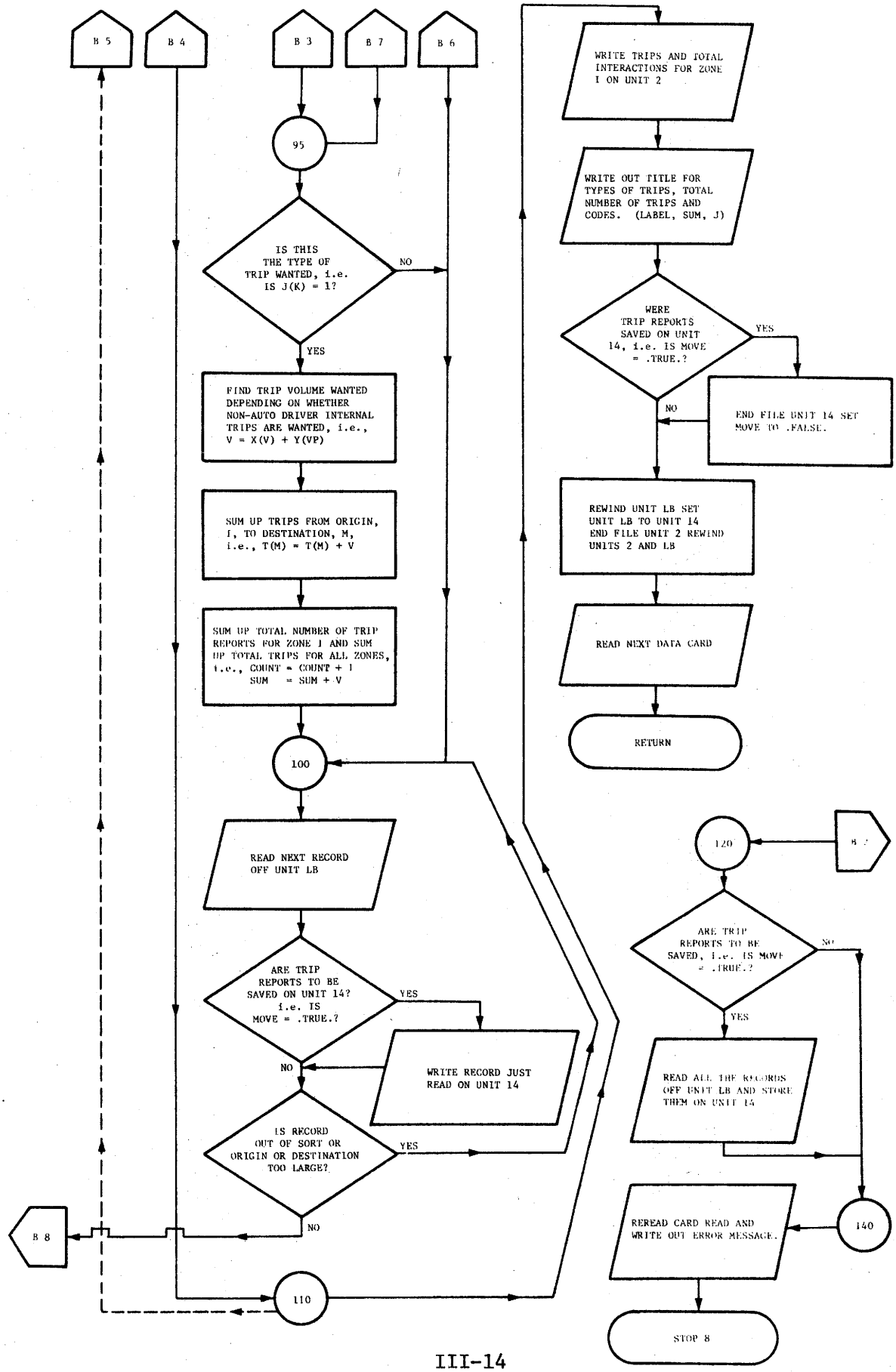

N 3
N 2
N 1
IS POSITION VARIABLE ABOVE THE DIAGONAL?
NO
YES
CALCULATE APPROPRIATE ADJUSTMENT FACTOR AND STORE IN 'R'
RECALCULATE ADJUSTMENT FACTOR AND STORE IN 'R'
SET VARIABLE 'X' TO 0.0
90
IS THE DESIRED VALUE 0.0
YES
NO
CALCULATE THE CHI-SQUARED VALUE AND STORE IN 'X'
SUM UP THE CHI-SQUARED VALUES AND STORE IN 'SUM'
IS THE RELATIVE VALUE 1.0?
YES
NO
ADJUST THE RELATIVE VALUE USING CALCULATED ADJUSTMENT FACTOR 'R'
SET THE VARIABLE 'FLAG' TO INDICATE AN ADJUSTMENT WAS MADE
100
IS IT THE TOP OF A PAGE?
NO
YES
PRINT DESCRIPTIVE TITLE AND COLUMN HEADINGS FOR TABLE
PRINT OUT TABLE SHOWING RESULTS FROM ADJUSTMENTS
200
PRINT OUT THE TOTAL SUM OF THE CHI-SQUARED VALUES
RETURN

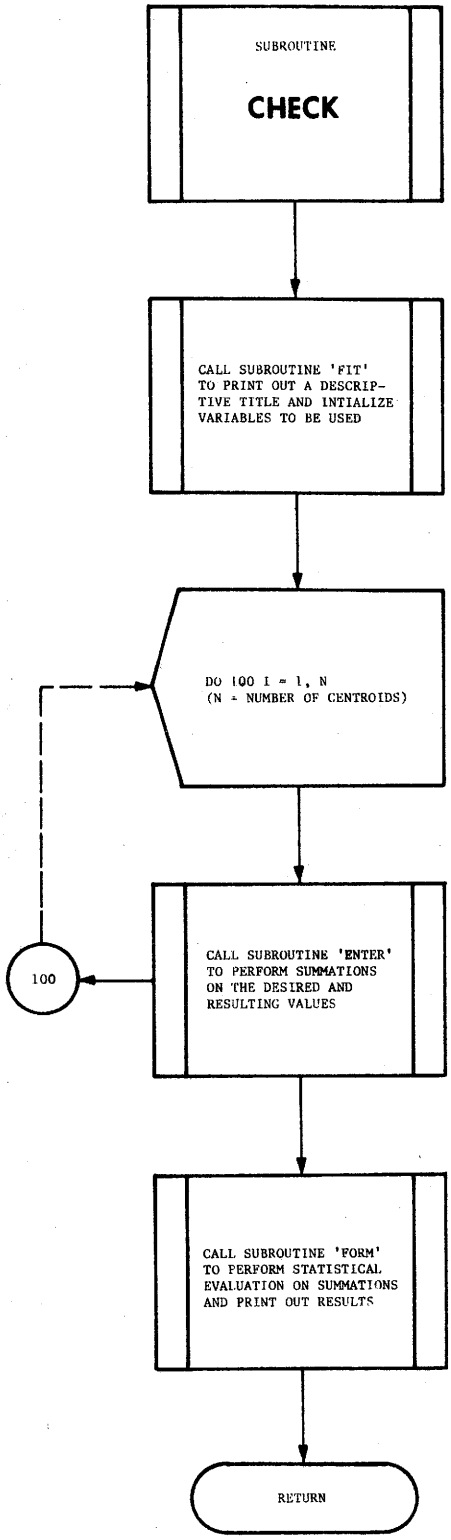

SUBROUTINE
ALTER
CALL SUBROUTINE 'NAME' TO PRINT NAME OF SUBROUTINE BEING EXECUTED 'ALTER'
REWIND UNITS MSO[4], MSN[16], MTO[3], MTN[22]
READ UNIT MSO[4] TO SKIP HEADER RECORD
READ UNIT MSN[16] TO SKIP HEADER RECORD
READ HEADER RECORD OFF UNIT MTO[2] WHICH CONTAINS TRIP MATRIX
WRITE HEADER ON UNIT MTN[22] WHICH WILL CONTAIN ALTERED TRIP MATRIX
DO 120 I = 1, N
(N = NUMBER OF CENTROIDS)
READ ORIGINAL SEPARATIONS FOR CENTROID 'I' OFF UNIT MSO[4] AND STORE IN 'KSO'
READ NEW SEPARATIONS FOR CENTROID 'I' OFF UNIT MSN[16] AND STORE IN 'KSN'
READ MODEL TRIP VOLUMES FOR CENTROID 'I' OFF UNIT MTO[3] AND STORE IN 'T'
DO 110 J = 1, N
(N = NUMBER OF CENTROIDS)
110
ALTER MODEL TRIP VOLUMES BASED ON CHANGE IN SEPARATION
120
WRITE ALTERED MODEL TRIP VOLUMES FOR CENTROID 'I' ON UNIT MTN[22]
END FILE UNIT MTN[22]
REWIND UNITS MSO[4], MSN[16], MTO[3], AND MTN[22]
RETURN

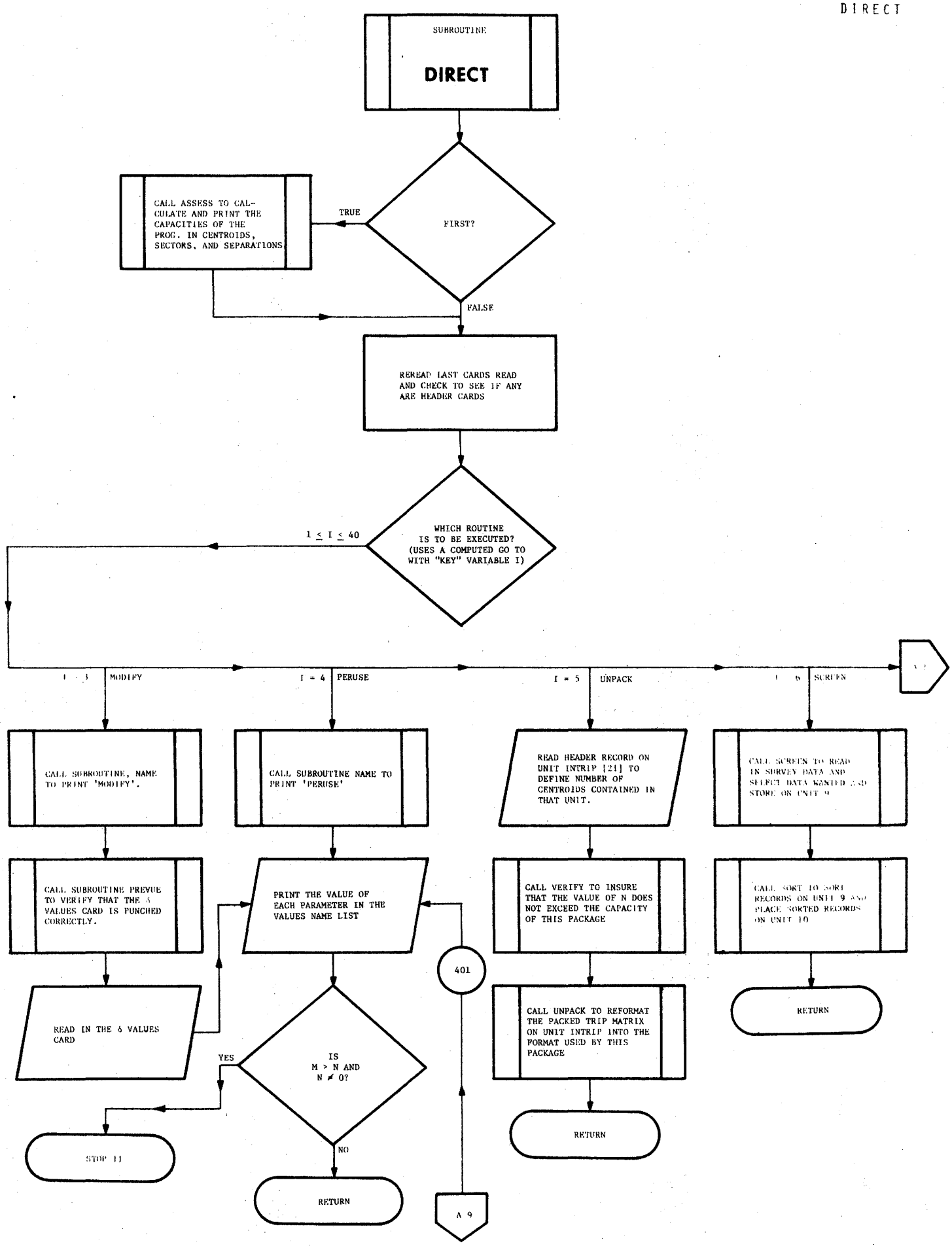

SUBROUTINE
ASSESS
THIS SUBROUTINE CALCULATES THE ABSOLUTE MAXIMUM NUMBER OF CENTROIDS, SEPARATION, AND INDEX FOR ANY TWO SECTORS THAT THIS PACKAGE IS CAPABLE OF HANDLING AT THE PRESENT TIME, THESE VALUES ARE PRINTED OUT AND SAVED FOR COMPARISON PURPOSES LATER IN SUBROUTINE 'VERIFY'
RETURN

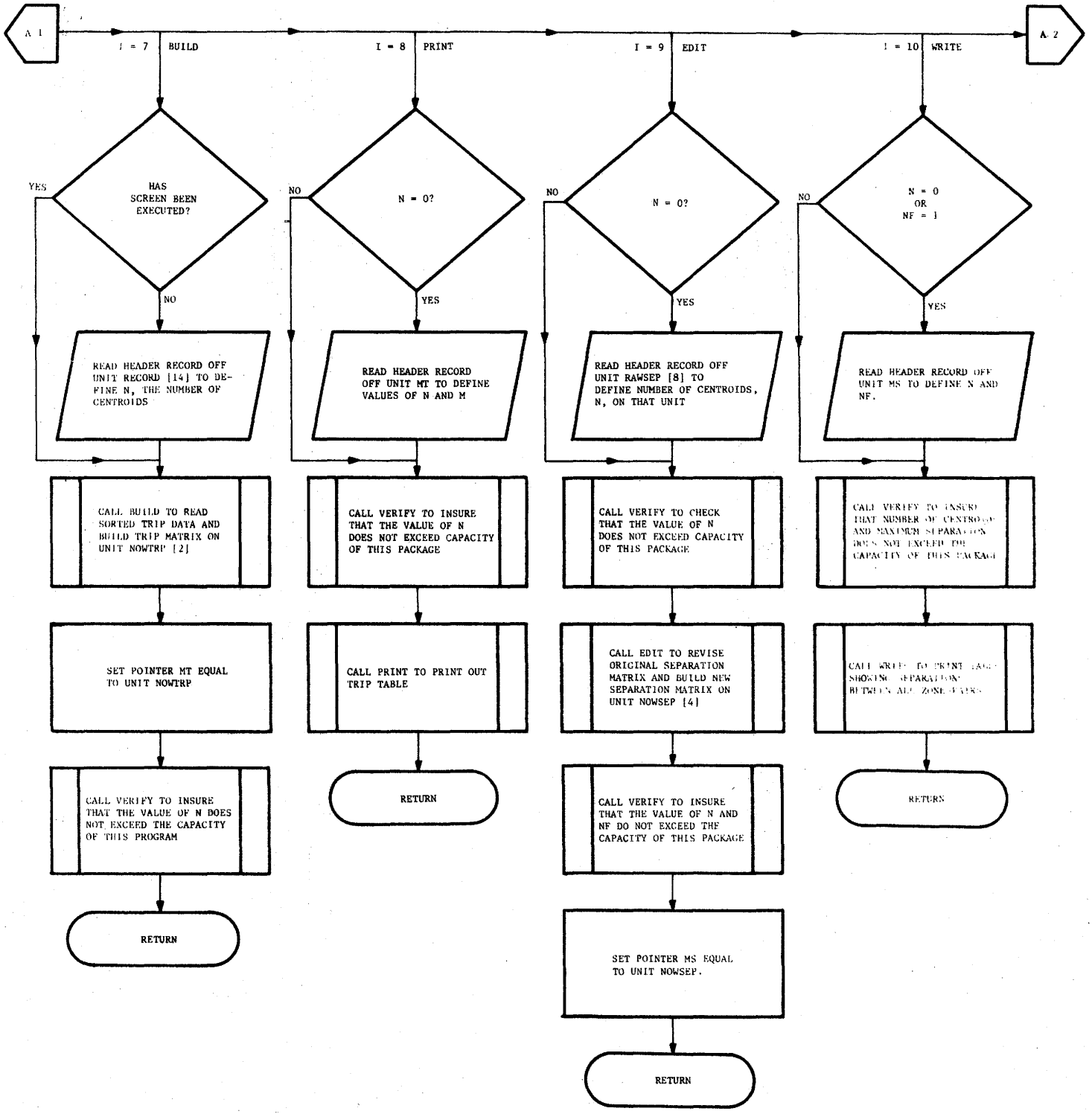

SUBROUTINE
BUILD
CALL NAME TO PRINT NAME OF SUBROUTINE BEING EXECUTED 'BUILD'
HAS SUBROUTINE SCREEN BEEN EXECUTED?
NO
SET LOGICAL VARIABLE MOVE TO .FALSE.
YES
REREAD LAST DATA CARD READ IN
IS CARD A CATEGORY CARD?
NO
B 2
YES
REREAD CARD AGAIN TO GET NAME OF TRIPS WANTED CODES REQUIRED [(KS(I), I=1, 14)], AND TITLE FOR DEF. OF TRIPS
ZERO OUT POINTER ARRAY J (40)
DO 90 I=1, 14
IS CODE, KS(I), VALUE BETWEEN 1 AND 40?
NO
YES
SET K = CODE KS(I)
SET POINTER J (K) = I
IS CODE KS(I), VALUE BETWEEN 11 AND 19?
YES
NO
IS CODE, KS(I), VALUE BETWEEN 21 AND 29?
YES
NO
90
IS CODE 37, 38, OR 39 WANTED
NO
YES
91
REWIND UNIT 18 AND READ HEADER RECORD OFF IT TO GET NUMBER OF ZONES
B 1

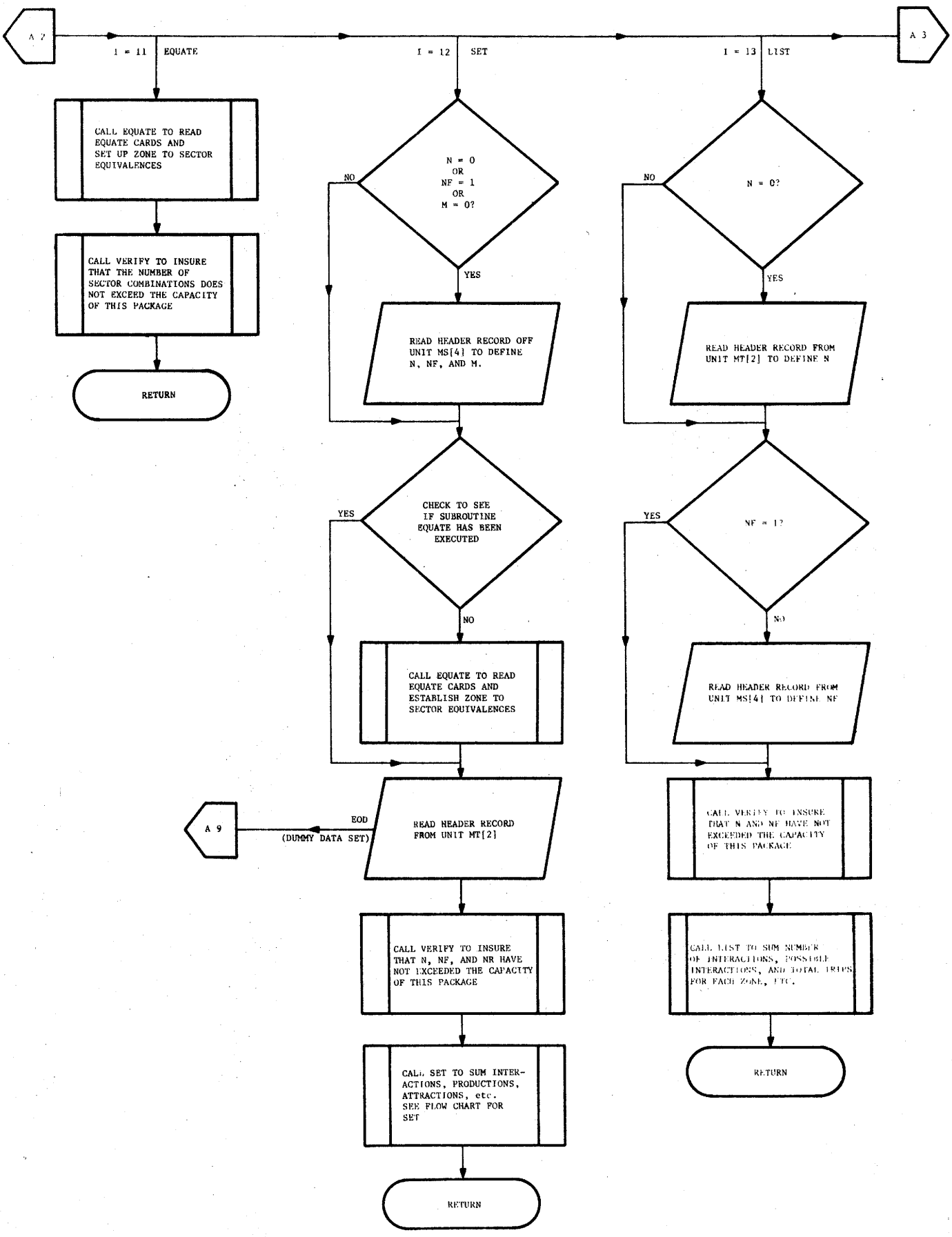

B 1
IS SURVEY TRIP REPORTS TO BE STORED ON UNIT 14, i.e., MOVE = .TRUE.
YES
WRITE HEADER RECORD ON UNIT 14. SAME ONE AS READ OFF UNIT 18
NO
WRITE HEADER RECORD ON UNIT 2 CONTAINING NUMBER OF ZONES, TYPE OF TRIPS, AND DATE
INITIALIZE VARIABLES SUM TO 0.0, X TO 1.0, AND Y TO 0.0
ARE NON-AUTO DRIVER INTERNAL TRIPS WANTED?
YES
SET VARIABLE Y = 1.0
NO
92
READ UNIT 18 TO GET ORIGIN (W), DESTINATION (M), PURPOSE CODE (K), TRIP VOLUME (V), AND PASSENGER VOLUME (VP)
ARE SURVEY TRIP REPORTS TO BE SAVED ON UNIT 14? i.e. MOVE = .TRUE.?
YES
WRITE INFORMATION JUST READ ON UNIT 14
NO
IS RECORD OUT OF SORT OR ORIGIN OR DESTINATION TOO LARGE?
YES
NO
B 5
B 4
DO 110 I=1, N
N=NUMBER OF ZONES
ZERO OUT ARRAY TO STORE TRIP VOLUMES IN, T(N) AND SET COUNTER, "COUNT" TO 0.0. IT COUNTS # TRIP REPORTS FOR ZONE I
105
B 8
IS ORIGIN L = INDEX I?
NO
YES
IS TRIP AN INTERNAL AUTO DRIVER TRIP?
YES
B 7
NO
SET MODE = V-970.0 TO DETERMINE CODE NUMBER FOR THIS TYPE OF TRIP
IS THIS TYPE OF TRIP WANTED, i.e. I(MODE)=1?
NO
B 6
YES
SET VOLUME OF TRIPS, V, EQUAL TO PASSENGER VOLUME, VP.
B 3

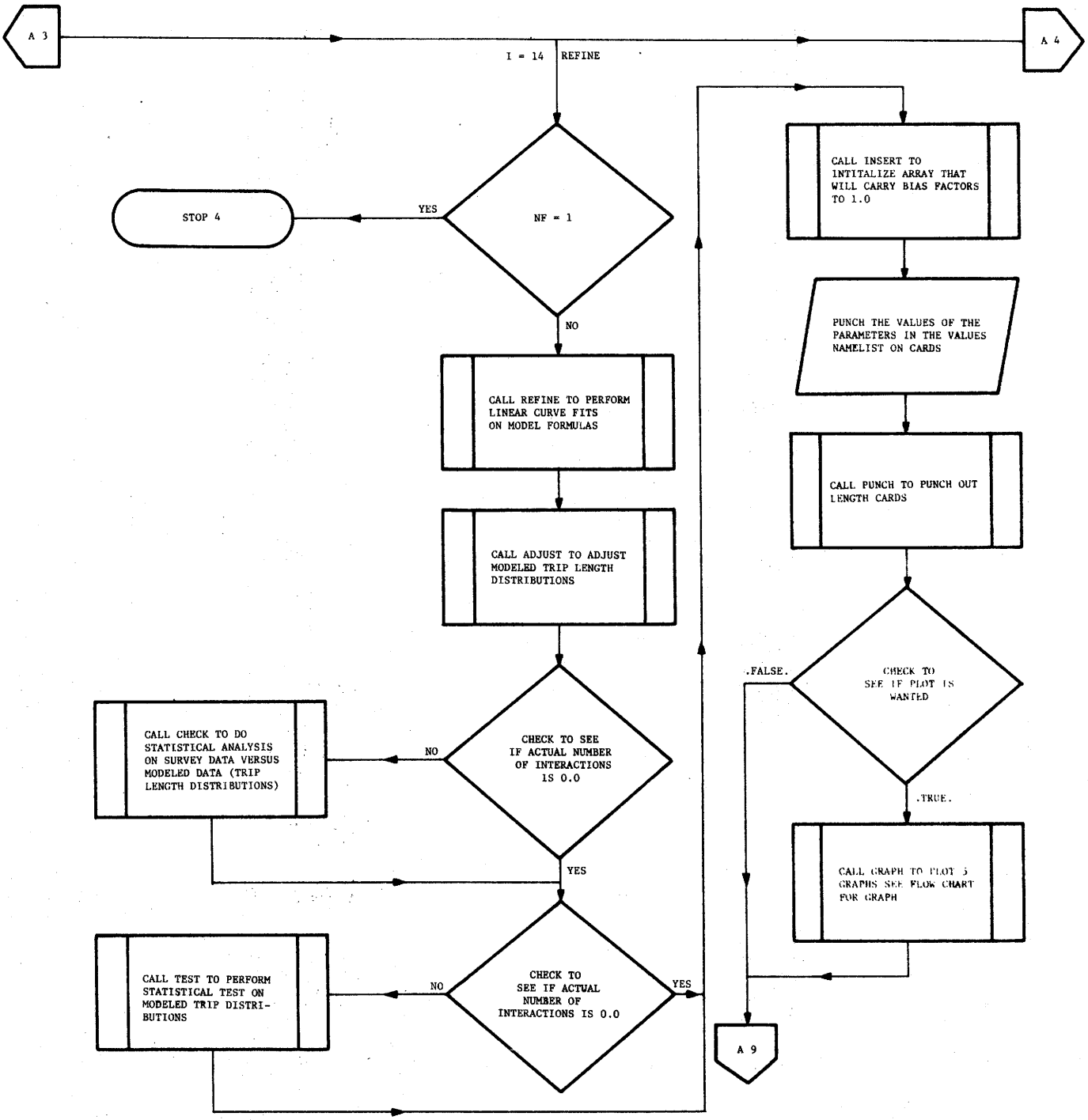

B 5
B 4
B 3
B 7
B 6
95
IS THIS THE TYPE OF TRIP WANTED, i.e. IS J(K) = 1?
NO
YES
FIND TRIP VOLUME WANTED DEPENDING ON WHETHER NON-AUTO DRIVER INTERNAL TRIPS ARE WANTED, i.e., V = X(V) + Y(VP)
SUM UP TRIPS FROM ORIGIN, I, TO DESTINATION, M, i.e., T(M) = T(M) + V
SUM UP TOTAL NUMBER OF TRIP REPORTS FOR ZONE I AND SUM UP TOTAL TRIPS FOR ALL ZONES, i.e., COUNT = COUNT + 1
SUM = SUM + V
100
READ NEXT RECORD OFF UNIT LB
ARE TRIP REPORTS TO BE SAVED ON UNIT 14? i.e. IS MOVE = .TRUE.?
YES
NO
WRITE RECORD JUST READ ON UNIT 14
IS RECORD OUT OF SORT OR ORIGIN OR DESTINATION TOO LARGE?
YES
NO
B 8
110
WRITE TRIPS AND TOTAL INTERACTIONS FOR ZONE I ON UNIT 2
WRITE OUT TITLE FOR TYPES OF TRIPS, TOTAL NUMBER OF TRIPS AND CODES. (LABEL, SUM, J)
WERE TRIP REPORTS SAVED ON UNIT 14, i.e. IS MOVE = .TRUE.?
YES
NO
END FILE UNIT 14 SET MOVE TO .FALSE.
REWIND UNIT LB SET UNIT LB TO UNIT 14 END FILE UNIT 2 REWIND UNITS 2 AND LB
READ NEXT DATA CARD
RETURN
120
B 7
ARE TRIP REPORTS TO BE SAVED, i.e. IS MOVE = .TRUE.?
NO
YES
READ ALL THE RECORDS OFF UNIT LB AND STORE THEM ON UNIT 14
140
REREAD CARD READ AND WRITE OUT ERROR MESSAGE.
STOP 8

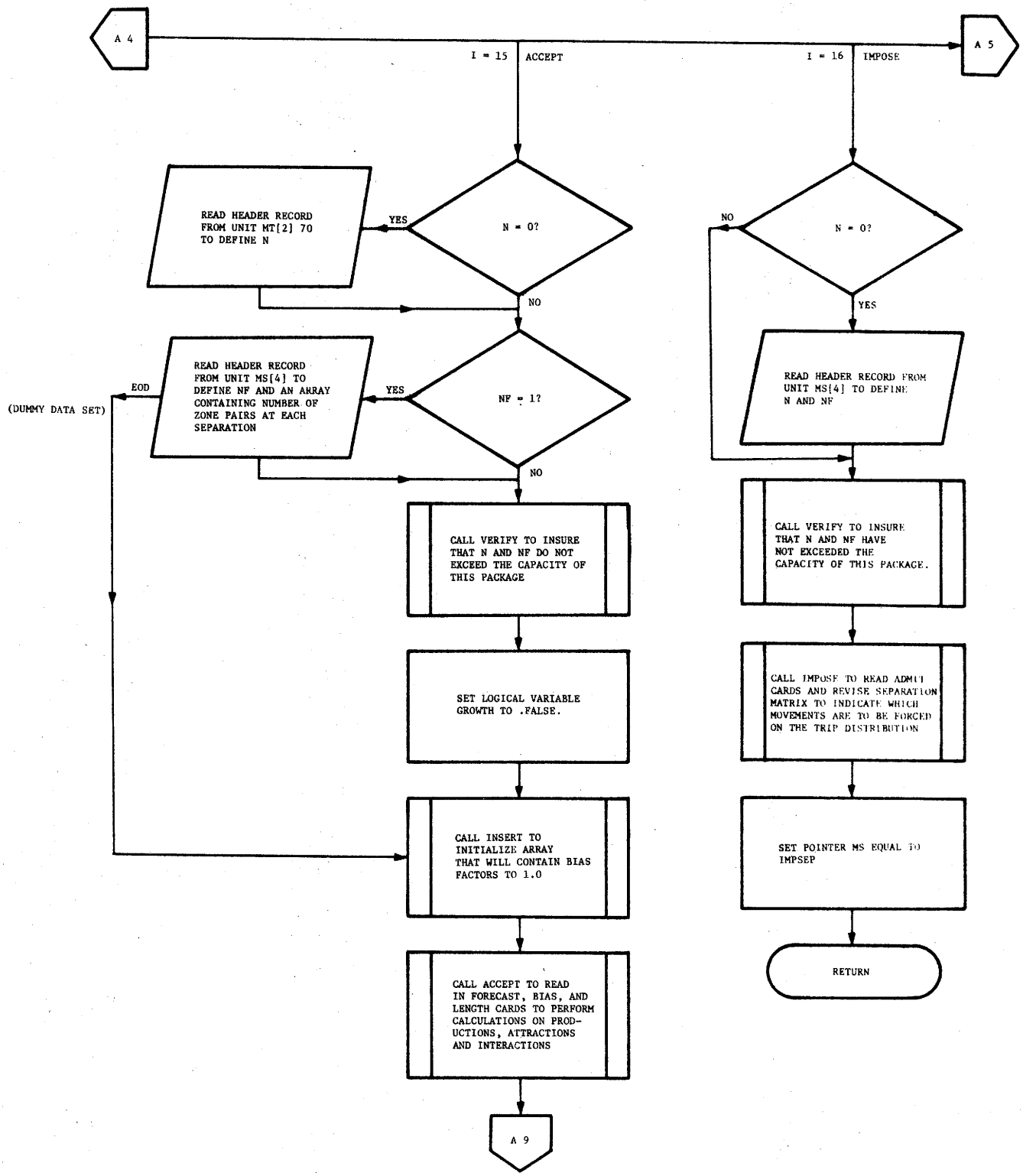

SUBROUTINE
CHECK
CALL SUBROUTINE 'FIT' TO PRINT OUT A DESCRIP-TIVE TITLE AND INTIALIZE VARIABLES TO BE USED
DO 100 I = 1, N
(N = NUMBER OF CENTROIDS)
CALL SUBROUTINE 'ENTER' TO PERFORM SUMMATIONS ON THE DESIRED AND RESULTING VALUES
100
CALL SUBROUTINE 'FORM' TO PERFORM STATISTICAL EVALUATION ON SUMMATIONS AND PRINT OUT RESULTS
RETURN

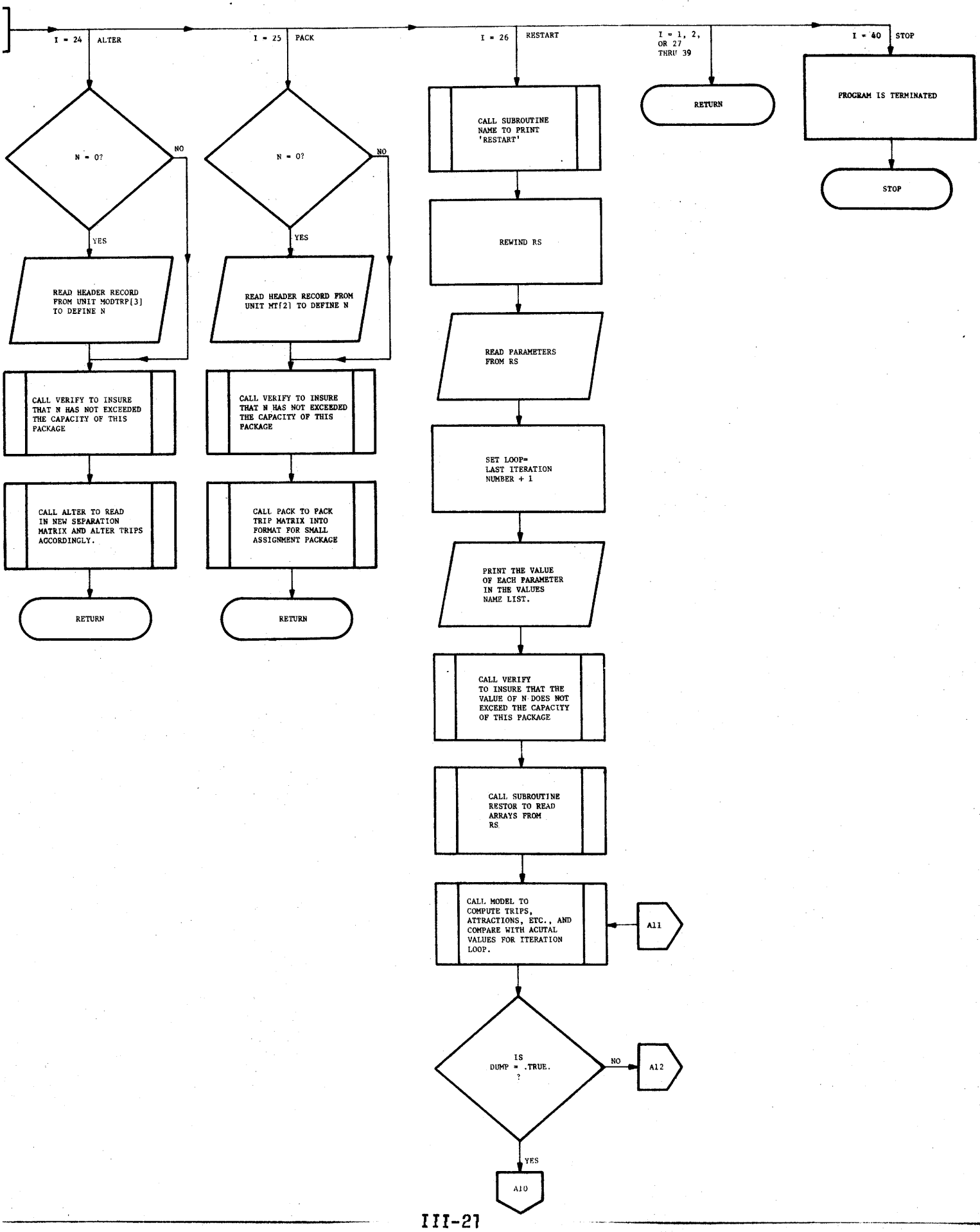

SUBROUTINE
DIRECT
FIRST?
TRUE
CALL ASSESS TO CALCULATE AND PRINT THE CAPACITIES OF THE PROG. IN CENTROIDS, SECTORS, AND SEPARATIONS
FALSE
REREAD LAST CARDS READ AND CHECK TO SEE IF ANY ARE HEADER CARDS
WHICH ROUTINE IS TO BE EXECUTED? (USES A COMPUTED GO TO WITH "KEY" VARIABLE I)
1 ≤ I ≤ 40
I = 3
MODIFY
I = 4
PERUSE
I = 5
UNPACK
I = 6
SCREEN
CALL SUBROUTINE, NAME TO PRINT 'MODIFY'.
CALL SUBROUTINE NAME TO PRINT 'PERUSE'
READ HEADER RECORD ON UNIT INTRIP [21] TO DEFINE NUMBER OF CENTROIDS CONTAINED IN THAT UNIT.
CALL SCREEN TO READ IN SURVEY DATA AND SELECT DATA WANTED AND STORE ON UNIT 9
CALL SUBROUTINE PREVUE TO VERIFY THAT THE 6 VALUES CARD IS PUNCHED CORRECTLY.
PRINT THE VALUE OF EACH PARAMETER IN THE VALUES NAME LIST
CALL VERIFY TO INSURE THAT THE VALUE OF N DOES NOT EXCEED THE CAPACITY OF THIS PACKAGE
CALL SORT TO SORT RECORDS ON UNIT 9 AND PLACE SORTED RECORDS ON UNIT 10
401
READ IN THE 6 VALUES CARD
IS M > N AND N ≠ 0?
YES
NO
CALL UNPACK TO REFORMAT THE PACKED TRIP MATRIX ON UNIT INTRIP INTO THE FORMAT USED BY THIS PACKAGE
RETURN
STOP 11
RETURN
A 9

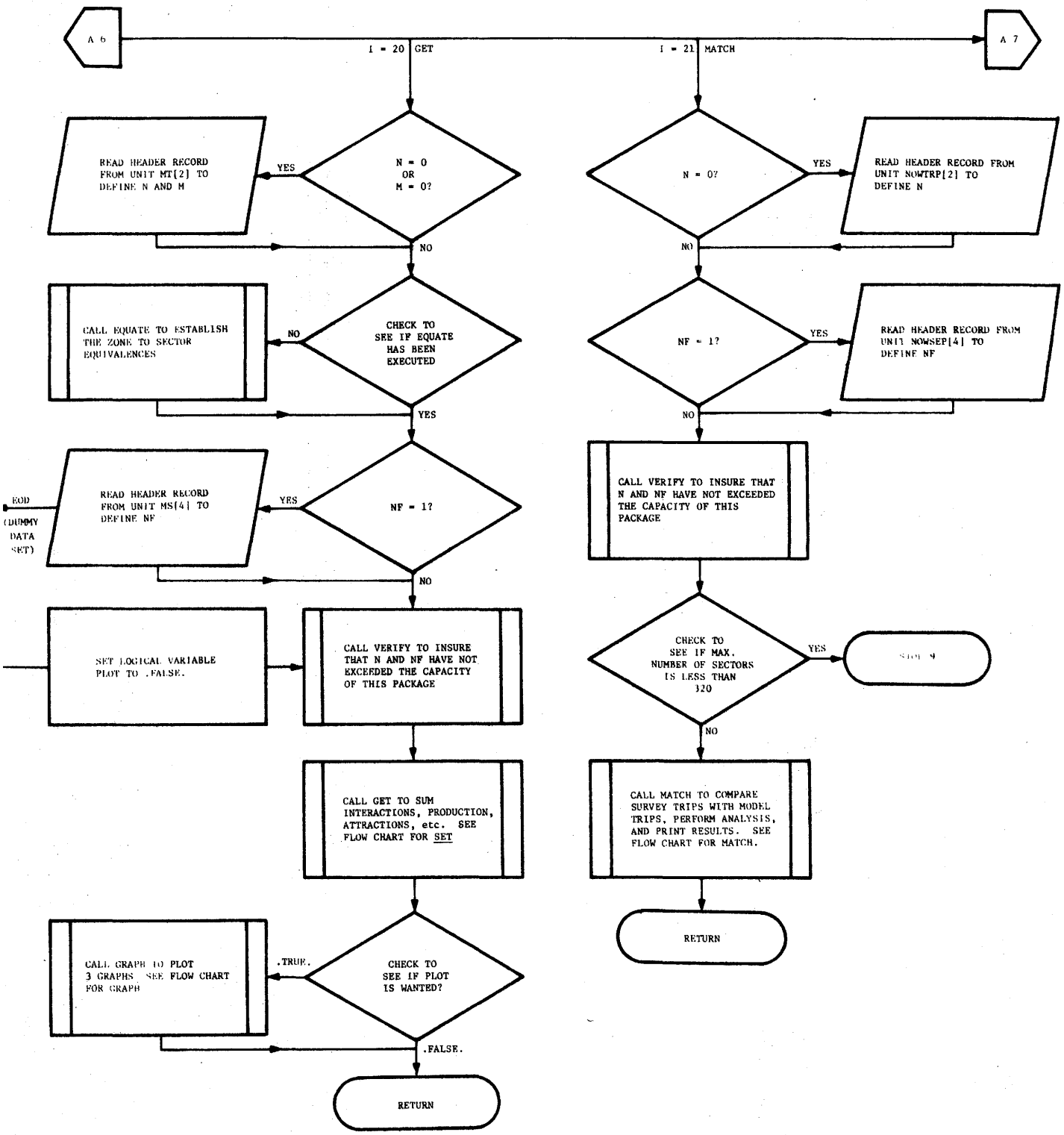

A 1
I = 7
BUILD
I = 8
PRINT
I = 9
EDIT
I = 10
WRITE
A 2
HAS SCREEN BEEN EXECUTED?
YES
NO
READ HEADER RECORD OFF UNIT RECORD [14] TO DEFINE N, THE NUMBER OF CENTROIDS
CALL BUILD TO READ SORTED TRIP DATA AND BUILD TRIP MATRIX ON UNIT NOWTRP [2]
SET POINTER MT EQUAL TO UNIT NOWTRP
CALL VERIFY TO INSURE THAT THE VALUE OF N DOES NOT EXCEED THE CAPACITY OF THIS PROGRAM
RETURN
N = 0?
NO
YES
READ HEADER RECORD OFF UNIT MT TO DEFINE VALUES OF N AND M
CALL VERIFY TO INSURE THAT THE VALUE OF N DOES NOT EXCEED CAPACITY OF THIS PACKAGE
CALL PRINT TO PRINT OUT TRIP TABLE
RETURN
N = 0?
NO
YES
READ HEADER RECORD OFF UNIT RAWSEP [8] TO DEFINE NUMBER OF CENTROIDS, N, ON THAT UNIT
CALL VERIFY TO CHECK THAT THE VALUE OF N DOES NOT EXCEED CAPACITY OF THIS PACKAGE
CALL EDIT TO REVISE ORIGINAL SEPARATION MATRIX AND BUILD NEW SEPARATION MATRIX ON UNIT NOWSEP [4]
CALL VERIFY TO INSURE THAT THE VALUE OF N AND NF DO NOT EXCEED THE CAPACITY OF THIS PACKAGE
SET POINTER MS EQUAL TO UNIT NOWSEP.
RETURN
N = 0 OR NF = 1
NO
YES
READ HEADER RECORD OFF UNIT MS TO DEFINE N AND NF.
RETURN

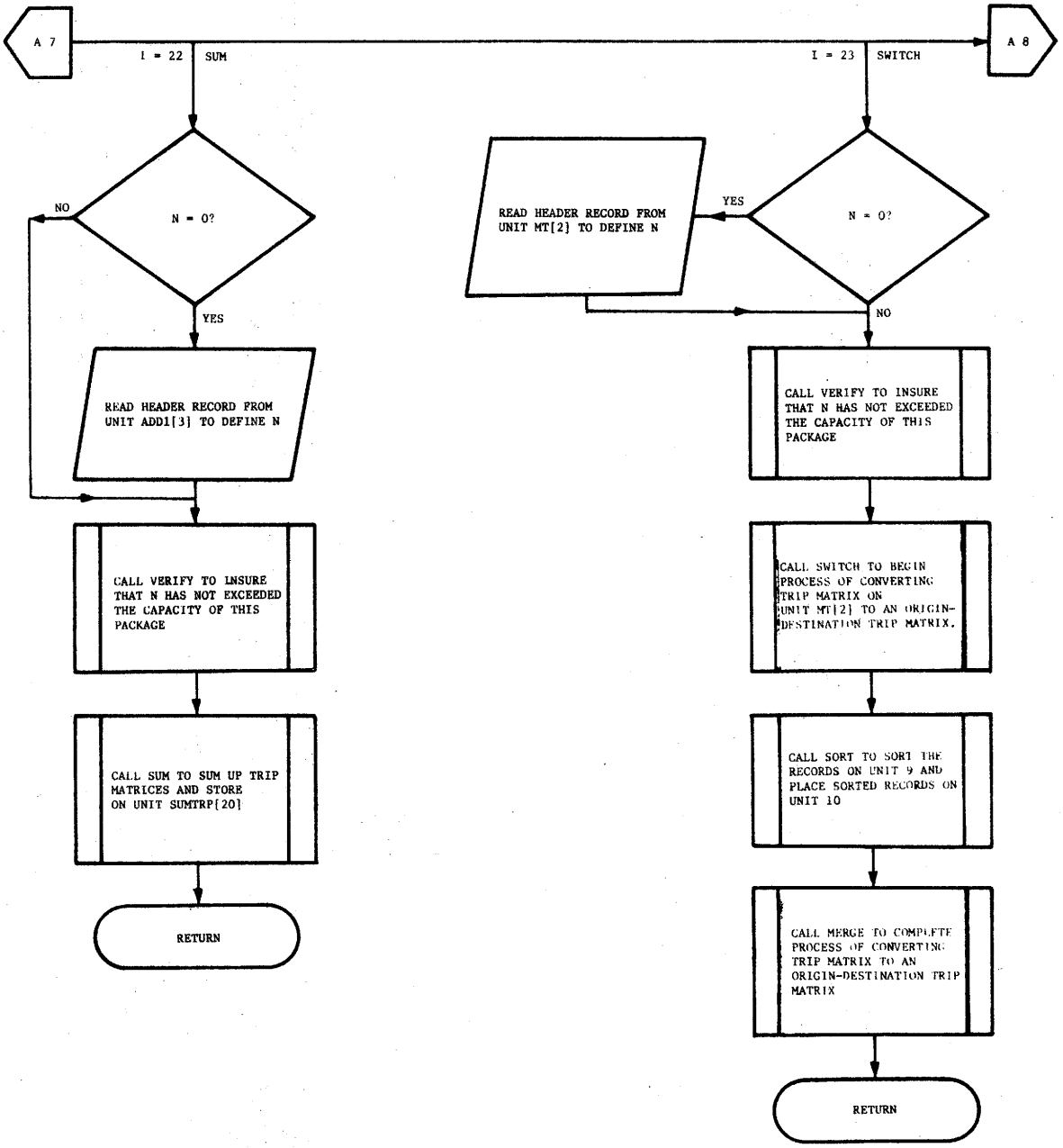

A 7
I = 11 EQUATE
I = 12 SET
I = 13 LIST
A 3
CALL EQUATE TO READ EQUATE CARDS AND SET UP ZONE TO SECTOR EQUIVALENCES
CALL VERIFY TO INSURE THAT THE NUMBER OF SECTOR COMBINATIONS DOES NOT EXCEED THE CAPACITY OF THIS PACKAGE
RETURN
N = 0 OR NF = 1 OR M = 0?
NO
YES
READ HEADER RECORD OFF UNIT MS[4] TO DEFINE N, NF, AND M.
CHECK TO SEE IF SUBROUTINE EQUATE HAS BEEN EXECUTED
YES
NO
CALL EQUATE TO READ EQUATE CARDS AND ESTABLISH ZONE TO SECTOR EQUIVALENCES
A 9
EOD (DUMMY DATA SET)
READ HEADER RECORD FROM UNIT MT[2]
CALL VERIFY TO INSURE THAT N, NF, AND NR HAVE NOT EXCEEDED THE CAPACITY OF THIS PACKAGE
CALL SET TO SUM INTERACTIONS, PRODUCTIONS, ATTRACTIONS, etc. SEE FLOW CHART FOR SET
RETURN
N = 0?
NO
YES
READ HEADER RECORD FROM UNIT MT[2] TO DEFINE N
NF = 1?
YES
NO
READ HEADER RECORD FROM UNIT MS[4] TO DEFINE NF
CALL VERIFY TO INSURE THAT N AND NF HAVE NOT EXCEEDED THE CAPACITY OF THIS PACKAGE
CALL LIST TO SUM NUMBER OF INTERACTIONS, POSSIBLE INTERACTIONS, AND TOTAL TRIPS FOR EACH ZONE, ETC.
RETURN

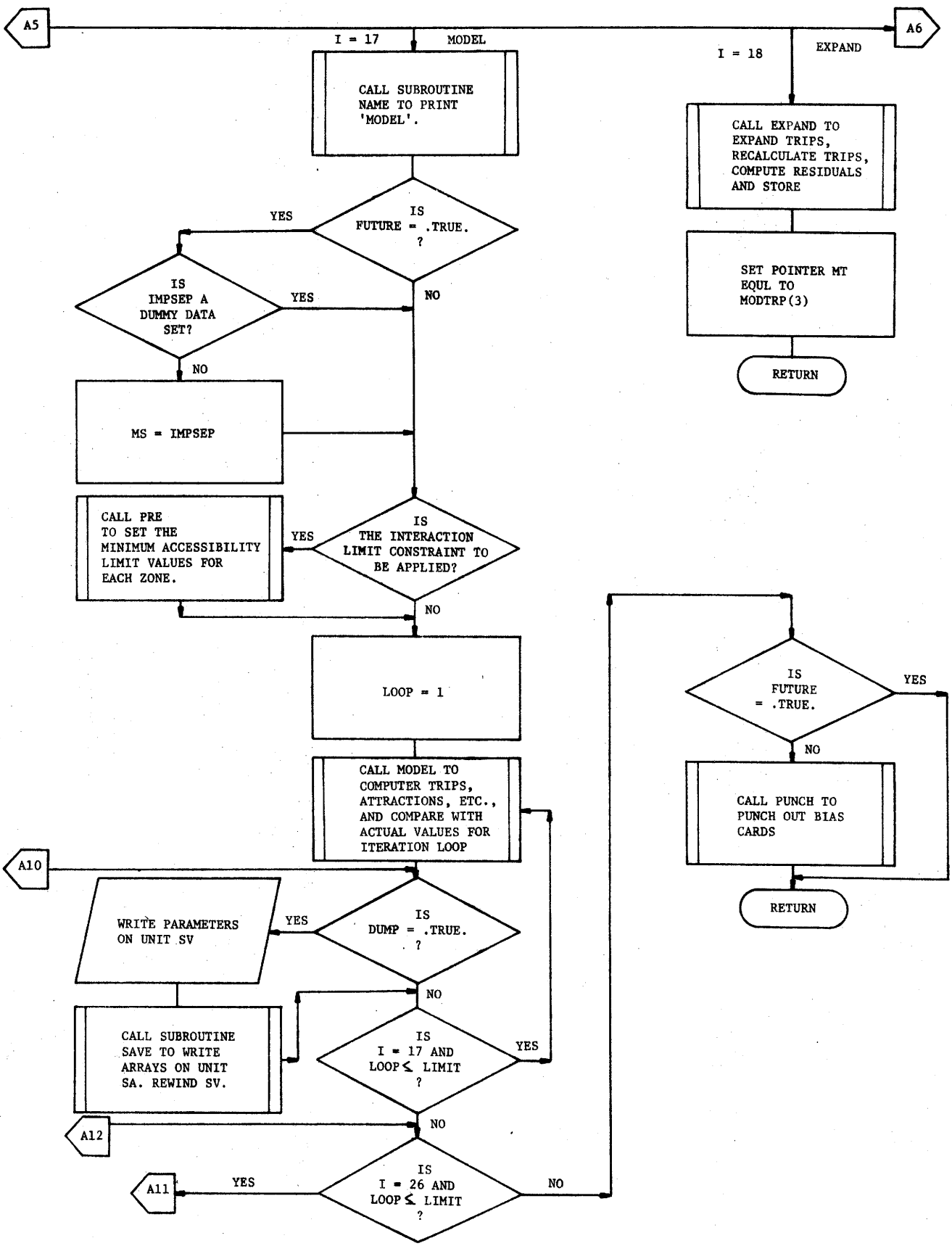

A 3
A 4
I = 14
REFINE
NF = 1
YES
STOP 4
NO
CALL REFINE TO PERFORM LINEAR CURVE FITS ON MODEL FORMULAS
CALL ADJUST TO ADJUST MODELED TRIP LENGTH DISTRIBUTIONS
CHECK TO SEE IF ACTUAL NUMBER OF INTERACTIONS IS 0.0
NO
CALL CHECK TO DO STATISTICAL ANALYSIS ON SURVEY DATA VERSUS MODELED DATA (TRIP LENGTH DISTRIBUTIONS)
YES
CHECK TO SEE IF ACTUAL NUMBER OF INTERACTIONS IS 0.0
NO
CALL TEST TO PERFORM STATISTICAL TEST ON MODELED TRIP DISTRI-BUTIONS
YES
CALL INSERT TO INTITALIZE ARRAY THAT WILL CARRY BIAS FACTORS TO 1.0
PUNCH THE VALUES OF THE PARAMETERS IN THE VALUES NAMELIST ON CARDS
CALL PUNCH TO PUNCH OUT LENGTH CARDS
CHECK TO SEE IF PLOT IS WANTED
.FALSE.
.TRUE.
CALL GRAPH TO PLOT 5 GRAPHS SEE FLOW CHART FOR GRAPH
A 9

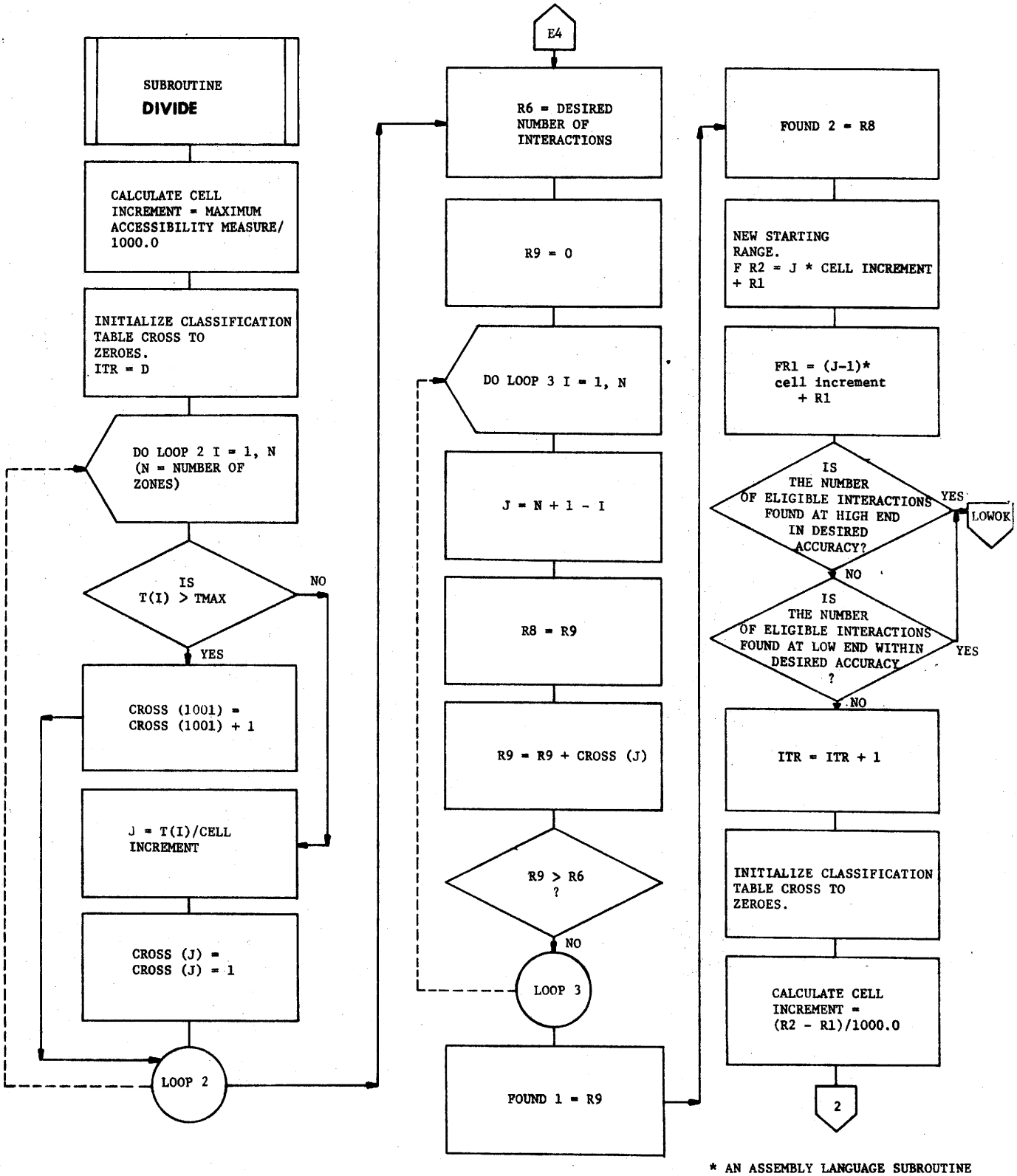

A 4
I = 15
ACCEPT
I = 16
IMPOSE
A 5
READ HEADER RECORD FROM UNIT MT[2] 70 TO DEFINE N
YES
N = 0?
NO
NO
N = 0?
YES
READ HEADER RECORD FROM UNIT MS[4] TO DEFINE N AND NF
EOD
(DUMMY DATA SET)
READ HEADER RECORD FROM UNIT MS[4] TO DEFINE NF AND AN ARRAY CONTAINING NUMBER OF ZONE PAIRS AT EACH SEPARATION
YES
NF = 1?
NO
CALL VERIFY TO INSURE THAT N AND NF DO NOT EXCEED THE CAPACITY OF THIS PACKAGE
CALL VERIFY TO INSURE THAT N AND NF HAVE NOT EXCEEDED THE CAPACITY OF THIS PACKAGE.
SET LOGICAL VARIABLE GROWTH TO .FALSE.
CALL IMPOSE TO READ ADMIT CARDS AND REVISE SEPARATION MATRIX TO INDICATE WHICH MOVEMENTS ARE TO BE FORCED ON THE TRIP DISTRIBUTION
CALL INSERT TO INITIALIZE ARRAY THAT WILL CONTAIN BIAS FACTORS TO 1.0
SET POINTER MS EQUAL TO IMPSEP
CALL ACCEPT TO READ IN FORECAST, BIAS, AND LENGTH CARDS TO PERFORM CALCULATIONS ON PRODUCTIONS, ATTRACTIONS AND INTERACTIONS
RETURN
A 9


I = 24
ALTER
N = 0?
NO
YES
READ HEADER RECORD FROM UNIT MODTRP[3] TO DEFINE N
CALL VERIFY TO INSURE THAT N HAS NOT EXCEEDED THE CAPACITY OF THIS PACKAGE
CALL ALTER TO READ IN NEW SEPARATION MATRIX AND ALTER TRIPS ACCORDINGLY.
RETURN
I = 25
PACK
N = 0?
NO
YES
READ HEADER RECORD FROM UNIT MT[2] TO DEFINE N
CALL VERIFY TO INSURE THAT N HAS NOT EXCEEDED THE CAPACITY OF THIS PACKAGE
CALL PACK TO PACK TRIP MATRIX INTO FORMAT FOR SMALL ASSIGNMENT PACKAGE
RETURN
I = 26
RESTART
CALL SUBROUTINE NAME TO PRINT 'RESTART'
REWIND RS
READ PARAMETERS FROM RS
SET LOOP= LAST ITERATION NUMBER + 1
PRINT THE VALUE OF EACH PARAMETER IN THE VALUES NAME LIST.
CALL VERIFY TO INSURE THAT THE VALUE OF N DOES NOT EXCEED THE CAPACITY OF THIS PACKAGE
CALL SUBROUTINE RESTOR TO READ ARRAYS FROM RS
CALL MODEL TO COMPUTE TRIPS, ATTRACTIONS, ETC., AND COMPARE WITH ACUTAL VALUES FOR ITERATION LOOP.
A11
IS DUMP = .TRUE. ?
NO
A12
YES
A10
I = 1, 2, OR 27 THRU 39
RETURN
I = 40
STOP
PROGRAM IS TERMINATED
STOP

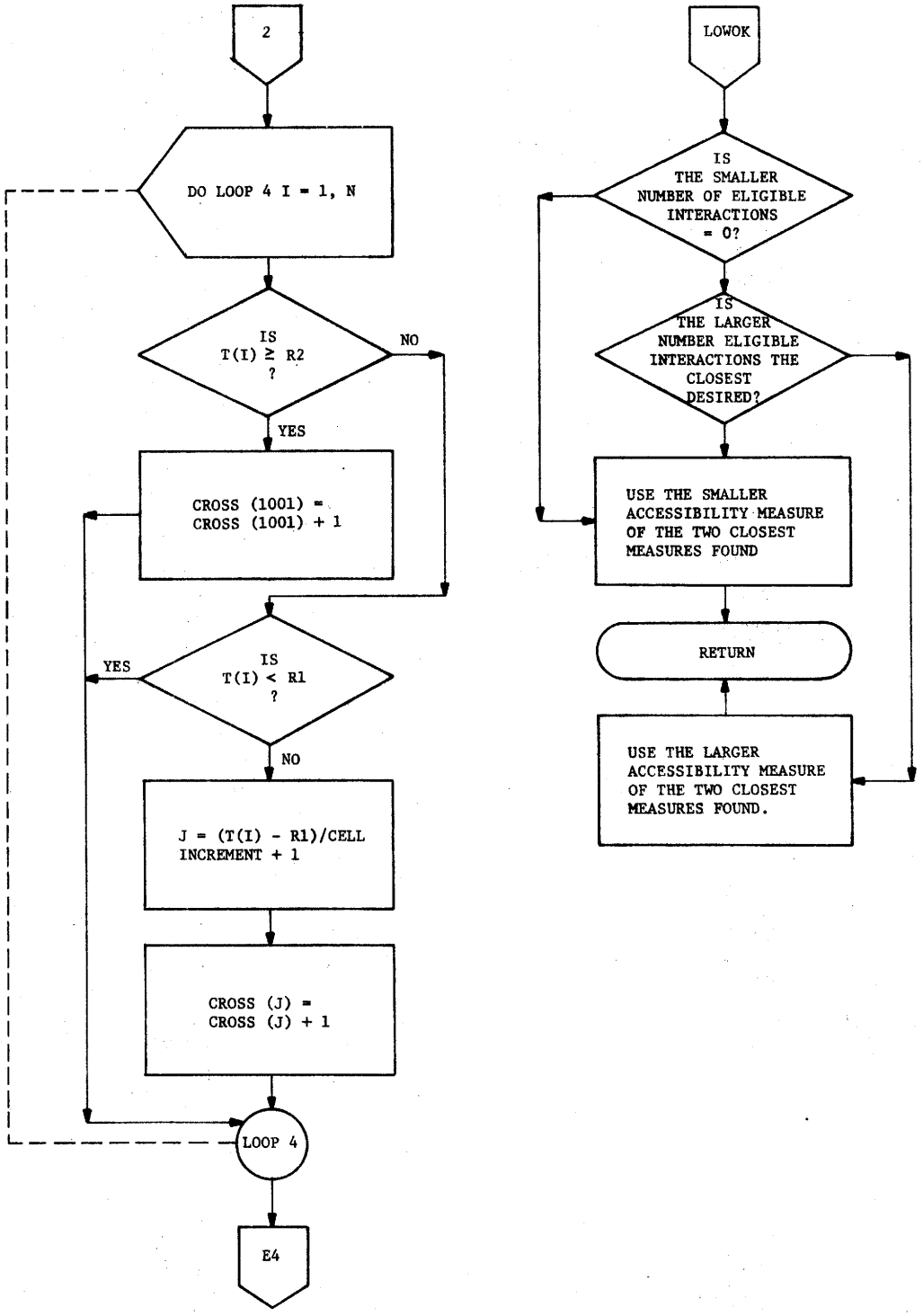

A 6
I = 20 GET
I = 21 MATCH
A 7
READ HEADER RECORD FROM UNIT MT[2] TO DEFINE N AND M
YES
N = 0 OR M = 0?
NO
CALL EQUATE TO ESTABLISH THE ZONE TO SECTOR EQUIVALENCES
NO
CHECK TO SEE IF EQUATE HAS BEEN EXECUTED
YES
EOD (DUMMY DATA SET)
READ HEADER RECORD FROM UNIT MS[4] TO DEFINE NF
YES
NF = 1?
NO
SET LOGICAL VARIABLE PLOT TO .FALSE.
CALL VERIFY TO INSURE THAT N AND NF HAVE NOT EXCEEDED THE CAPACITY OF THIS PACKAGE
CALL GET TO SUM INTERACTIONS, PRODUCTION, ATTRACTIONS, etc. SEE FLOW CHART FOR SET
CALL GRAPH TO PLOT 3 GRAPHS SEE FLOW CHART FOR GRAPH
.TRUE.
CHECK TO SEE IF PLOT IS WANTED?
.FALSE.
RETURN
N = 0?
YES
READ HEADER RECORD FROM UNIT NOWTRP[2] TO DEFINE N
NO
NF = 1?
YES
READ HEADER RECORD FROM UNIT NOWSEP[4] TO DEFINE NF
NO
CALL VERIFY TO INSURE THAT N AND NF HAVE NOT EXCEEDED THE CAPACITY OF THIS PACKAGE
CHECK TO SEE IF MAX. NUMBER OF SECTORS IS LESS THAN 320
YES
NO
CALL MATCH TO COMPARE SURVEY TRIPS WITH MODEL TRIPS, PERFORM ANALYSIS, AND PRINT RESULTS. SEE FLOW CHART FOR MATCH.
RETURN

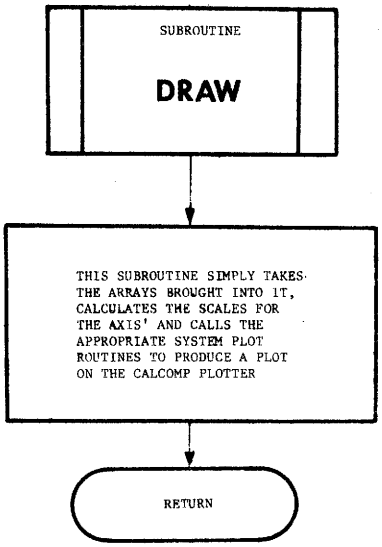

A 7
I = 22
SUM
I = 23
SWITCH
A 8
N = 0?
NO
YES
READ HEADER RECORD FROM UNIT ADD1[3] TO DEFINE N
CALL VERIFY TO INSURE THAT N HAS NOT EXCEEDED THE CAPACITY OF THIS PACKAGE
CALL SUM TO SUM UP TRIP MATRICES AND STORE ON UNIT SUMTRP[20]
RETURN
READ HEADER RECORD FROM UNIT MT[2] TO DEFINE N
N = 0?
YES
NO
CALL VERIFY TO INSURE THAT N HAS NOT EXCEEDED THE CAPACITY OF THIS PACKAGE
CALL SWITCH TO BEGIN PROCESS OF CONVERTING TRIP MATRIX ON UNIT MT[2] TO AN ORIGIN-DESTINATION TRIP MATRIX.
CALL SORT TO SORT THE RECORDS ON UNIT 9 AND PLACE SORTED RECORDS ON UNIT 10
CALL MERGE TO COMPLETE PROCESS OF CONVERTING TRIP MATRIX TO AN ORIGIN-DESTINATION TRIP MATRIX
RETURN

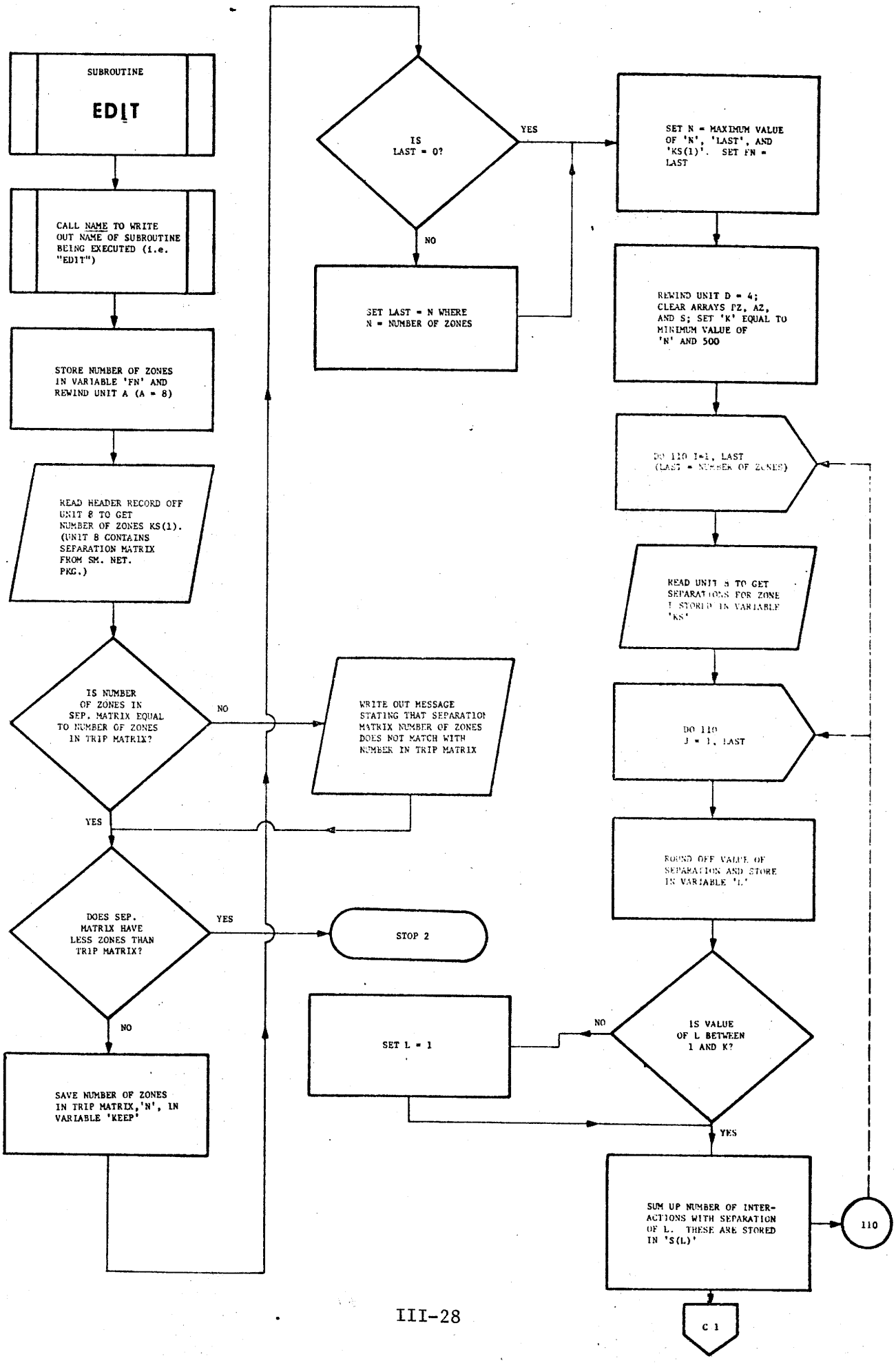

A5
I = 17
MODEL
I = 18
EXPAND
A6
CALL SUBROUTINE NAME TO PRINT 'MODEL'.
CALL EXPAND TO EXPAND TRIPS, RECALCULATE TRIPS, COMPUTE RESIDUALS AND STORE
SET POINTER MT EQUL TO MODTRP(3)
RETURN
IS FUTURE = .TRUE. ?
YES
NO
IS IMPSEP A DUMMY DATA SET?
YES
NO
MS = IMPSEP
CALL PRE TO SET THE MINIMUM ACCESSIBILITY LIMIT VALUES FOR EACH ZONE.
IS THE INTERACTION LIMIT CONSTRAINT TO BE APPLIED?
YES
NO
LOOP = 1
CALL MODEL TO COMPUTER TRIPS, ATTRACTIONS, ETC., AND COMPARE WITH ACTUAL VALUES FOR ITERATION LOOP
A10
WRITE PARAMETERS ON UNIT SV
IS DUMP = .TRUE. ?
YES
NO
CALL SUBROUTINE SAVE TO WRITE ARRAYS ON UNIT SA. REWIND SV.
IS I = 17 AND LOOP ≤ LIMIT ?
YES
NO
A12
A11
IS I = 26 AND LOOP ≤ LIMIT ?
YES
NO
IS FUTURE = .TRUE.
YES
NO
CALL PUNCH TO PUNCH OUT BIAS CARDS
RETURN

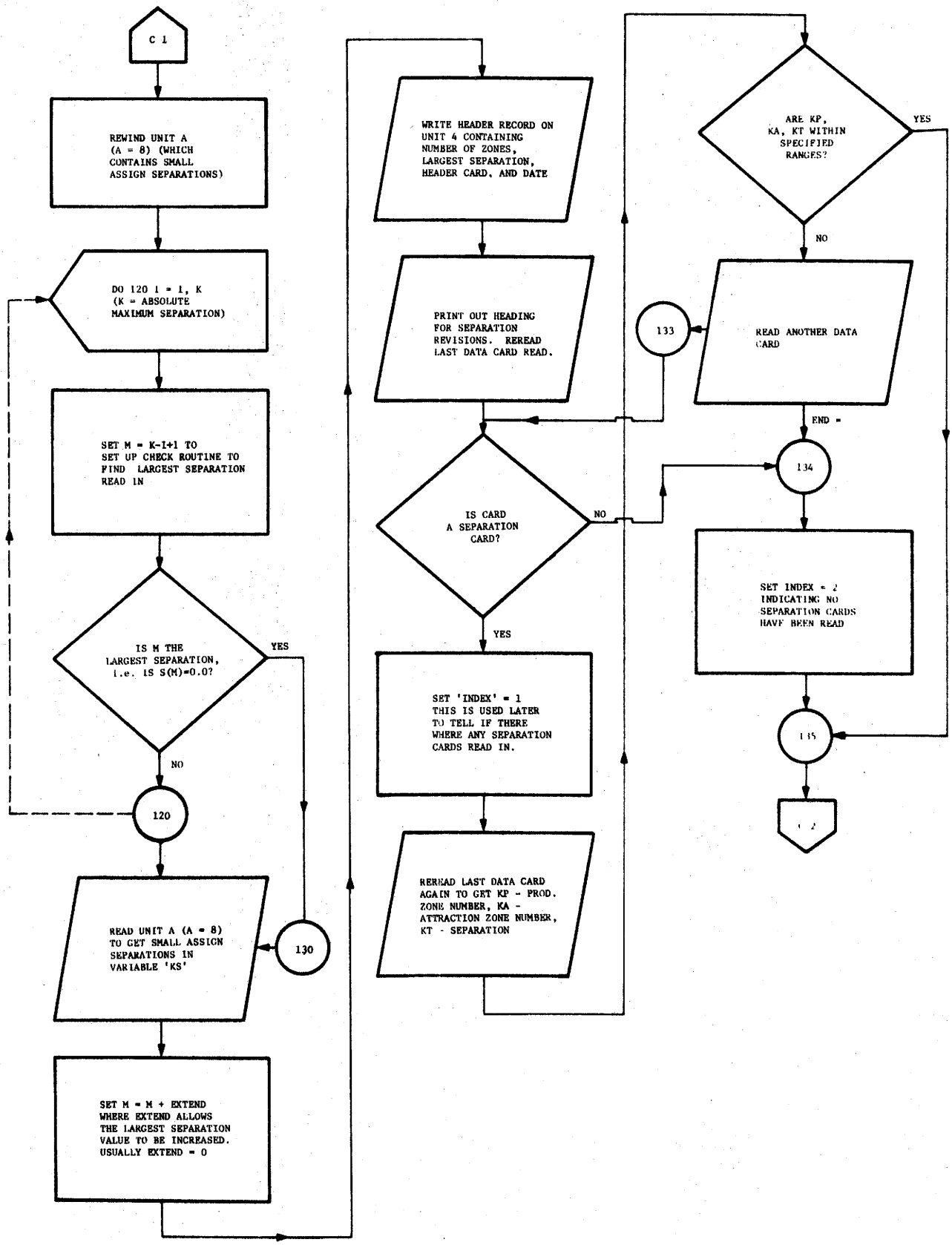


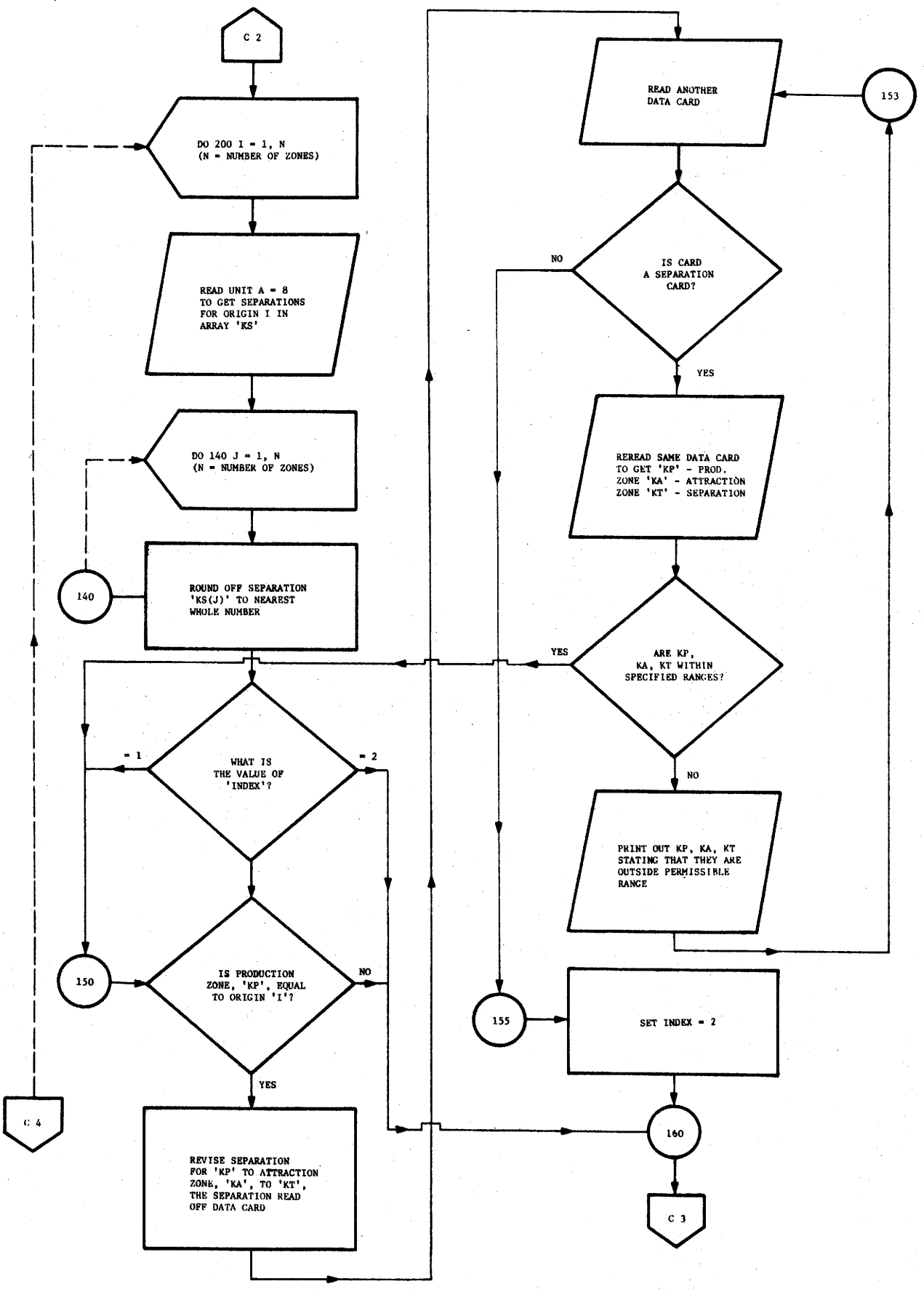
* AN ASSEMBLY LANGUAGE SUBROUTINE

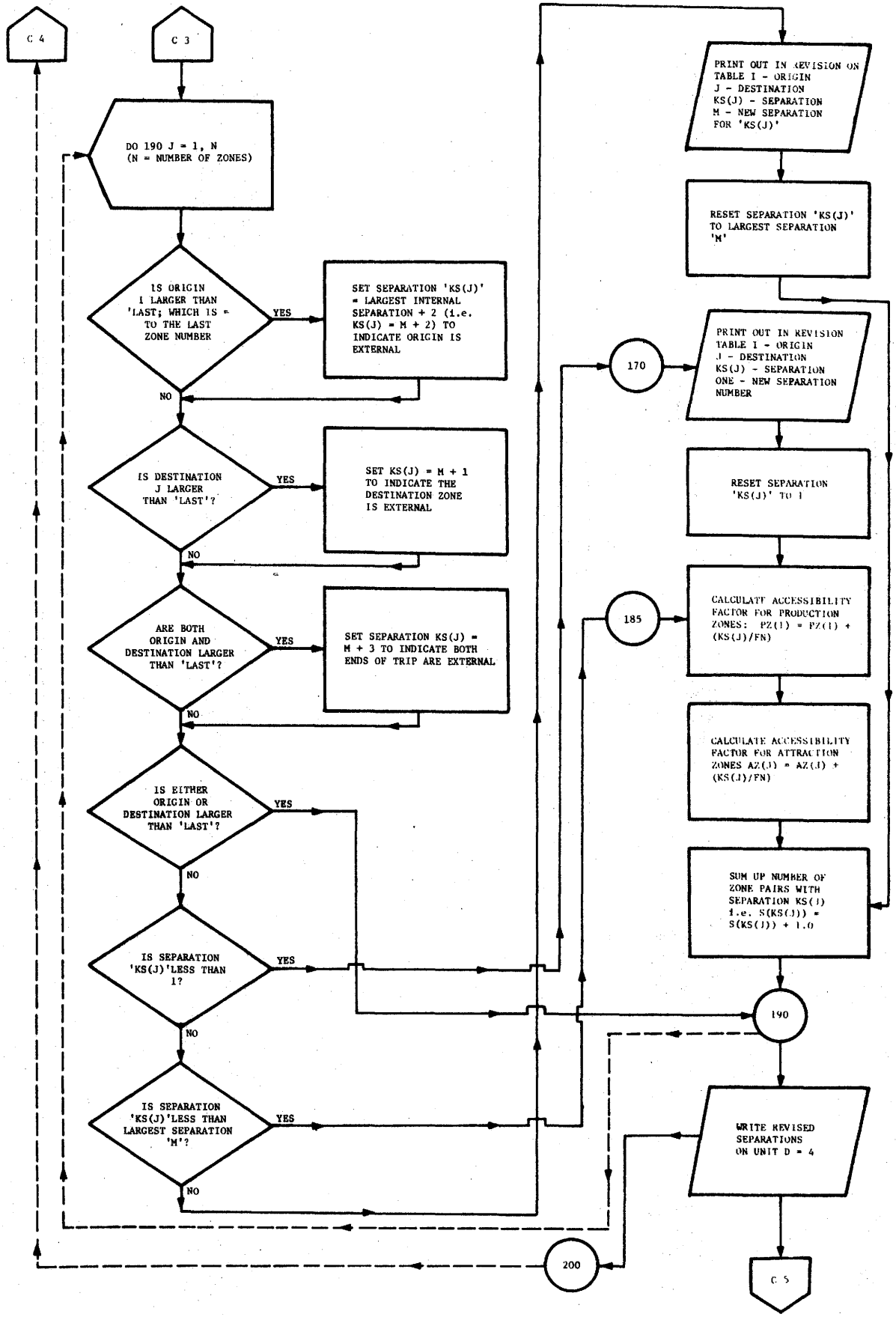

2
DO LOOP 4 I = 1, N
IS T(I) ≥ R2 ?
NO
YES
CROSS (1001) = CROSS (1001) + 1
IS T(I) < R1 ?
YES
NO
J = (T(I) - R1)/CELL INCREMENT + 1
CROSS (J) = CROSS (J) + 1
LOOP 4
E4
LOWOK
IS THE SMALLER NUMBER OF ELIGIBLE INTERACTIONS = 0?
IS THE LARGER NUMBER ELIGIBLE INTERACTIONS THE CLOSEST DESIRED?
USE THE SMALLER ACCESSIBILITY MEASURE OF THE TWO CLOSEST MEASURES FOUND
RETURN
USE THE LARGER ACCESSIBILITY MEASURE OF THE TWO CLOSEST MEASURES FOUND.

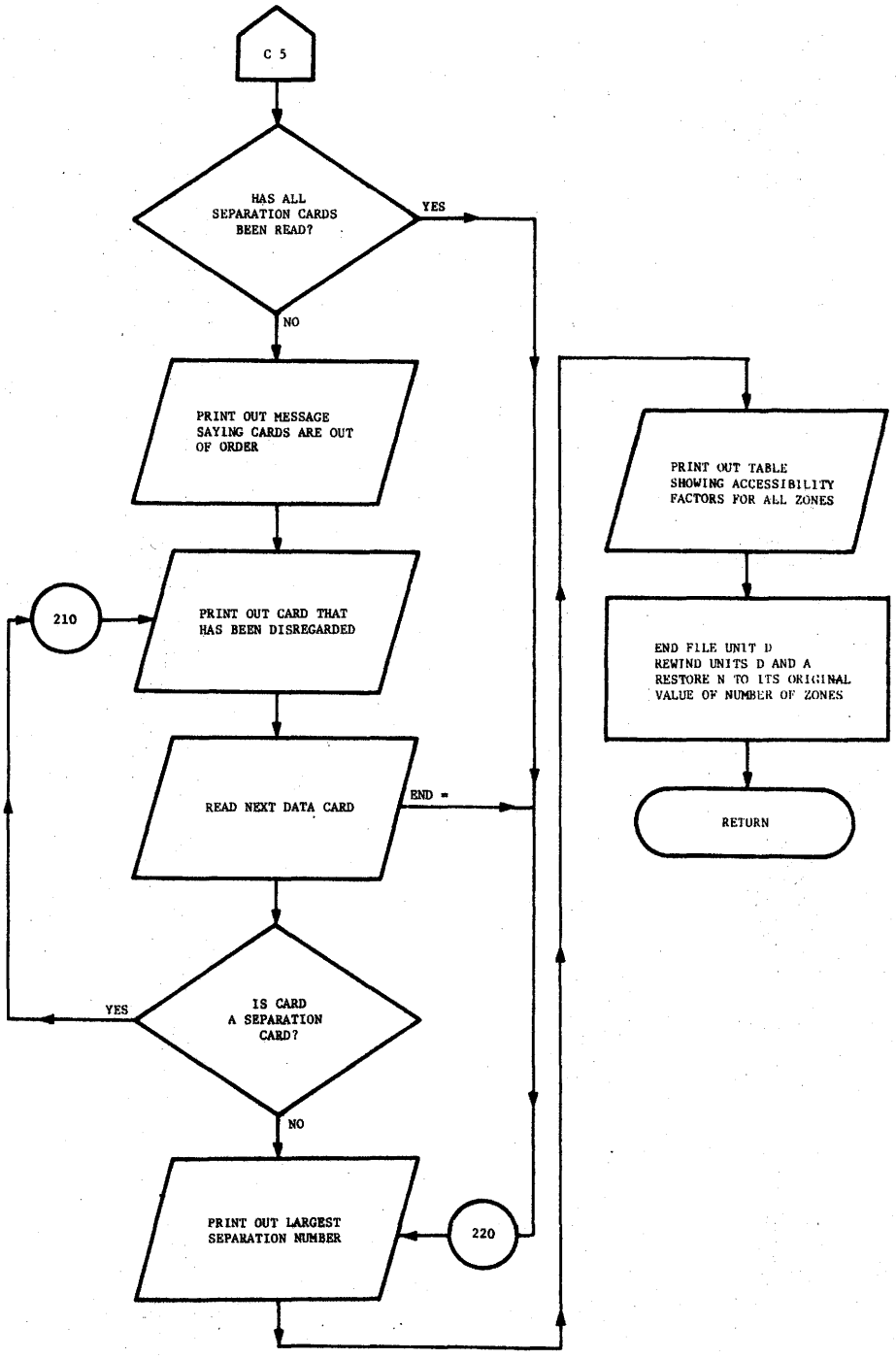

SUBROUTINE
DRAW
THIS SUBROUTINE SIMPLY TAKES THE ARRAYS BROUGHT INTO IT, CALCULATES THE SCALES FOR THE AXIS' AND CALLS THE APPROPRIATE SYSTEM PLOT ROUTINES TO PRODUCE A PLOT ON THE CALCOMP PLOTTER
RETURN

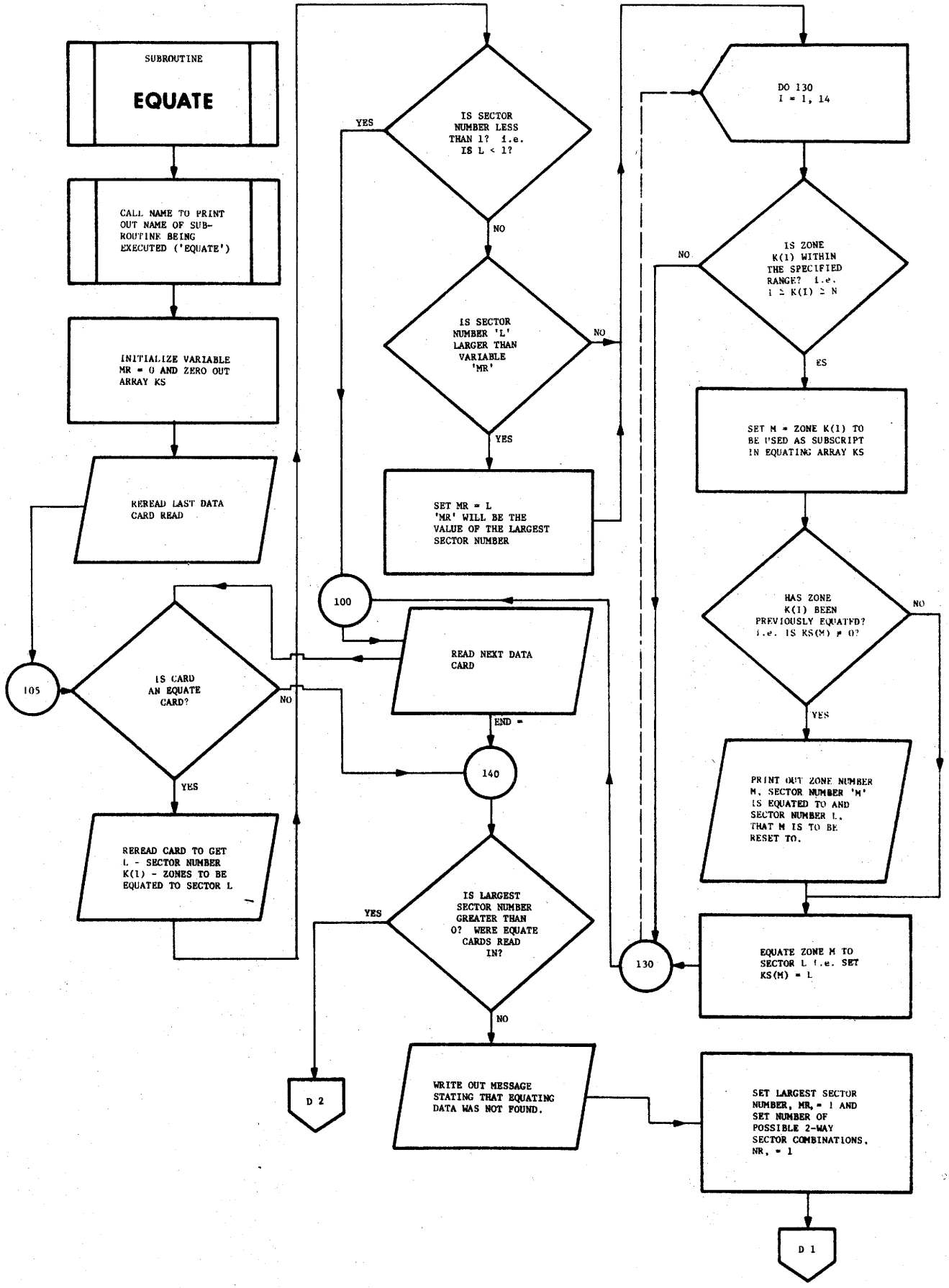

SUBROUTINE
EDIT
CALL NAME TO WRITE OUT NAME OF SUBROUTINE BEING EXECUTED (i.e. "EDIT")
STORE NUMBER OF ZONES IN VARIABLE 'FN' AND REWIND UNIT A (A = 8)
READ HEADER RECORD OFF UNIT 8 TO GET NUMBER OF ZONES KS(1). (UNIT 8 CONTAINS SEPARATION MATRIX FROM SM. NET. PKG.)
IS NUMBER OF ZONES IN SEP. MATRIX EQUAL TO NUMBER OF ZONES IN TRIP MATRIX?
NO
WRITE OUT MESSAGE STATING THAT SEPARATION MATRIX NUMBER OF ZONES DOES NOT MATCH WITH NUMBER IN TRIP MATRIX
YES
DOES SEP. MATRIX HAVE LESS ZONES THAN TRIP MATRIX?
YES
STOP 2
NO
SAVE NUMBER OF ZONES IN TRIP MATRIX,'N', IN VARIABLE 'KEEP'
IS LAST = 0?
YES
SET N = MAXIMUM VALUE OF 'N', 'LAST', AND 'KS(1)'. SET FN = LAST
NO
SET LAST = N WHERE N = NUMBER OF ZONES
REWIND UNIT D = 4; CLEAR ARRAYS PZ, AZ, AND S; SET 'K' EQUAL TO MINIMUM VALUE OF 'N' AND 500
DO 110 I=1, LAST (LAST = NUMBER OF ZONES)
READ UNIT 8 TO GET SEPARATIONS FOR ZONE I STORED IN VARIABLE 'KS'
DO 110 J = 1, LAST
ROUND OFF VALUE OF SEPARATION AND STORE IN VARIABLE 'L'
IS VALUE OF L BETWEEN 1 AND K?
NO
SET L = 1
YES
SUM UP NUMBER OF INTERACTIONS WITH SEPARATION OF L. THESE ARE STORED IN 'S(L)'
110
C 1

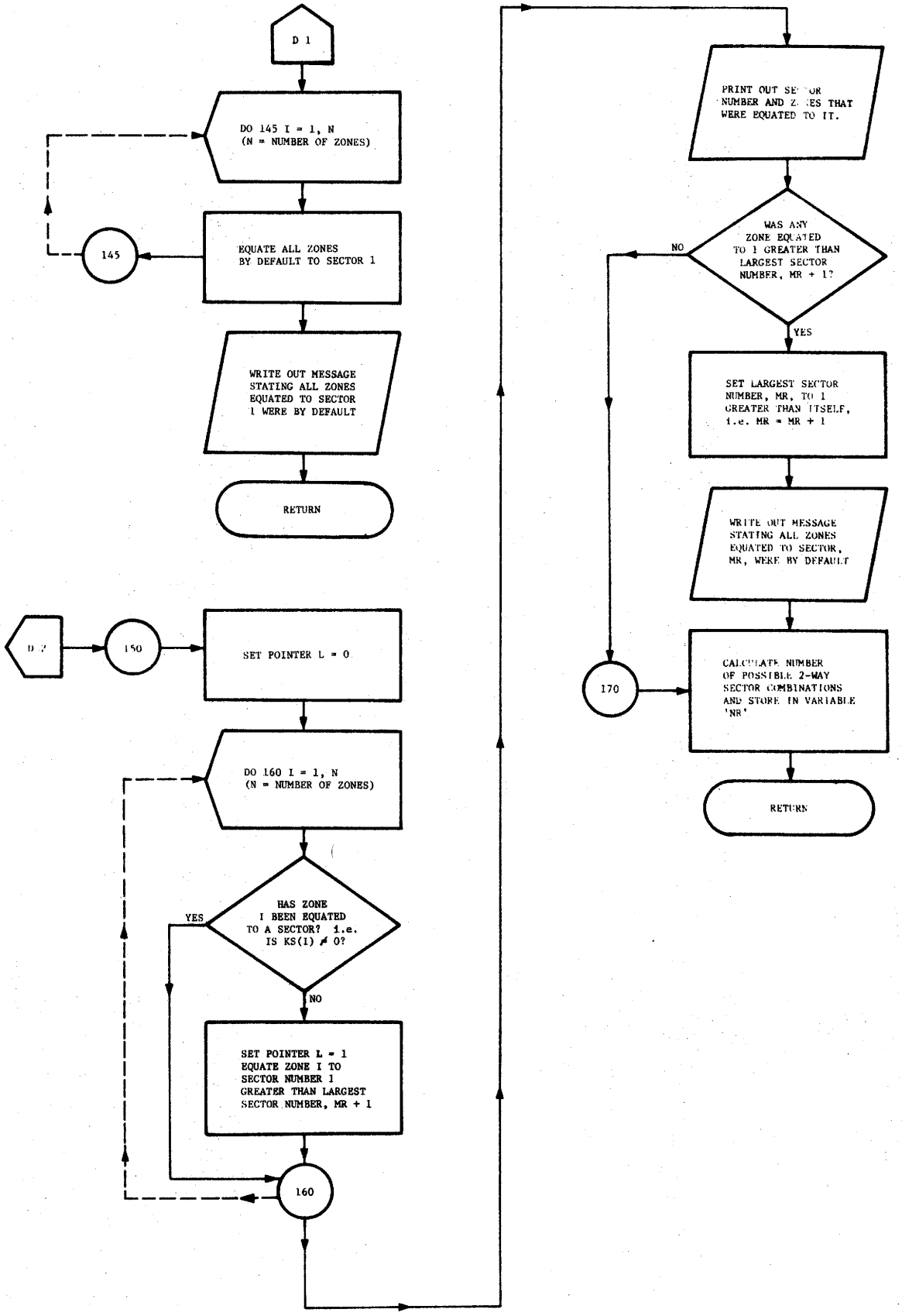

C 1
REWIND UNIT A (A = 8) (WHICH CONTAINS SMALL ASSIGN SEPARATIONS)
DO 120 I = 1, K (K = ABSOLUTE MAXIMUM SEPARATION)
SET M = K-I+1 TO SET UP CHECK ROUTINE TO FIND LARGEST SEPARATION READ IN
IS M THE LARGEST SEPARATION, i.e. IS S(M)=0.0?
YES
NO
120
READ UNIT A (A = 8) TO GET SMALL ASSIGN SEPARATIONS IN VARIABLE 'KS'
130
SET M = M + EXTEND WHERE EXTEND ALLOWS THE LARGEST SEPARATION VALUE TO BE INCREASED. USUALLY EXTEND = 0
WRITE HEADER RECORD ON UNIT 4 CONTAINING NUMBER OF ZONES, LARGEST SEPARATION, HEADER CARD, AND DATE
PRINT OUT HEADING FOR SEPARATION REVISIONS. REREAD LAST DATA CARD READ.
IS CARD A SEPARATION CARD?
NO
YES
SET 'INDEX' = 1 THIS IS USED LATER TO TELL IF THERE WHERE ANY SEPARATION CARDS READ IN.
REREAD LAST DATA CARD AGAIN TO GET KP - PROD. ZONE NUMBER, KA - ATTRACTION ZONE NUMBER, KT - SEPARATION
ARE KP, KA, KT WITHIN SPECIFIED RANGES?
YES
NO
133
READ ANOTHER DATA CARD
END =
134
SET INDEX = 2 INDICATING NO SEPARATION CARDS HAVE BEEN READ
135

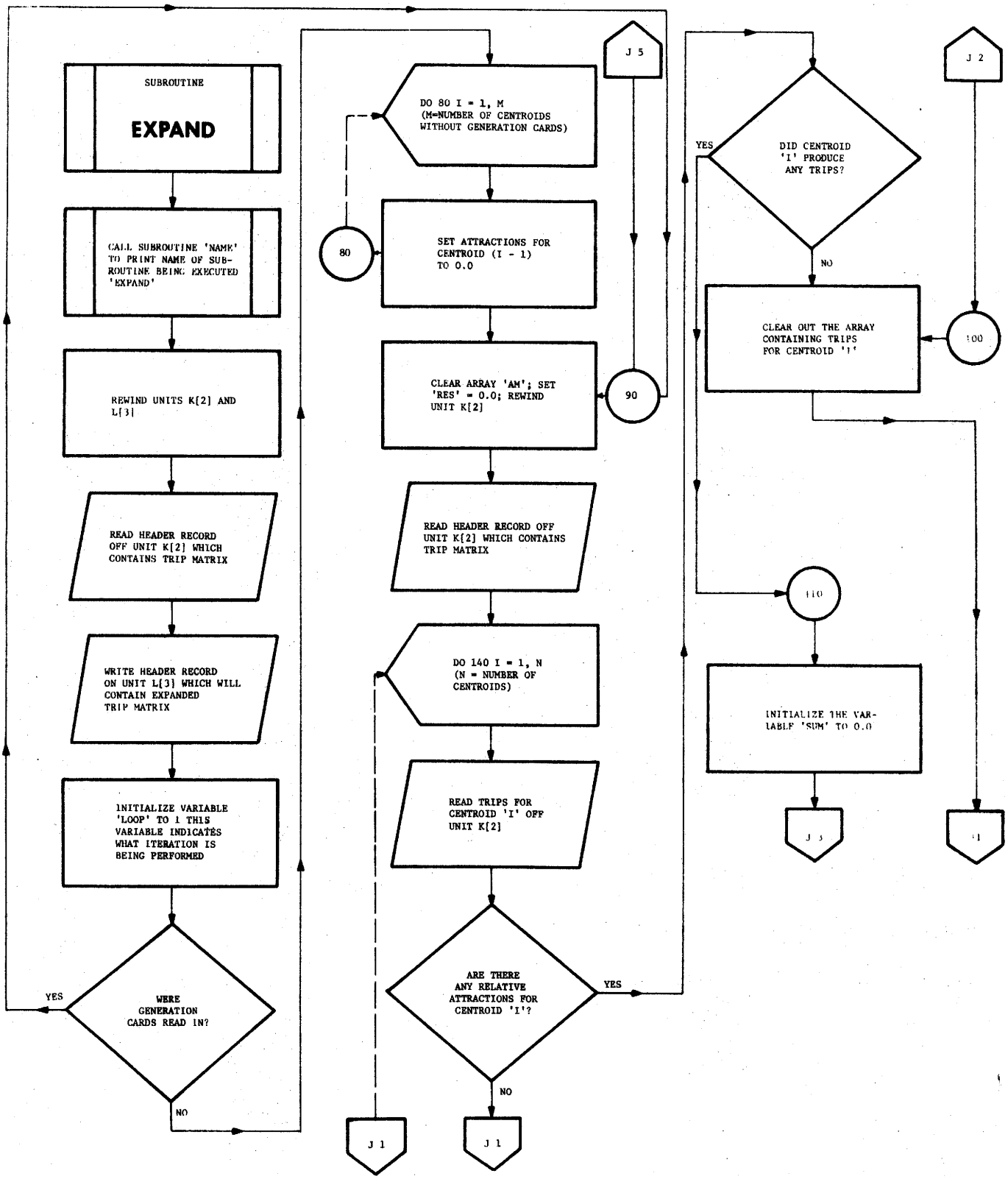

C 2
DO 200 I = 1, N
(N = NUMBER OF ZONES)
READ UNIT A = 8
TO GET SEPARATIONS
FOR ORIGIN I IN
ARRAY 'KS'
DO 140 J = 1, N
(N = NUMBER OF ZONES)
140
ROUND OFF SEPARATION
'KS(J)' TO NEAREST
WHOLE NUMBER
WHAT IS
THE VALUE OF
'INDEX'?
= 1
= 2
150
IS PRODUCTION
ZONE, 'KP', EQUAL
TO ORIGIN 'I'?
NO
YES
C 4
REVISE SEPARATION
FOR 'KP' TO ATTRACTION
ZONE, 'KA', TO 'KT',
THE SEPARATION READ
OFF DATA CARD
READ ANOTHER
DATA CARD
153
IS CARD
A SEPARATION
CARD?
NO
YES
REREAD SAME DATA CARD
TO GET 'KP' - PROD.
ZONE 'KA' - ATTRACTION
ZONE 'KT' - SEPARATION
ARE KP,
KA, KT WITHIN
SPECIFIED RANGES?
YES
NO
PRINT OUT KP, KA, KT
STATING THAT THEY ARE
OUTSIDE PERMISSIBLE
RANGE
155
SET INDEX = 2
160
C 3

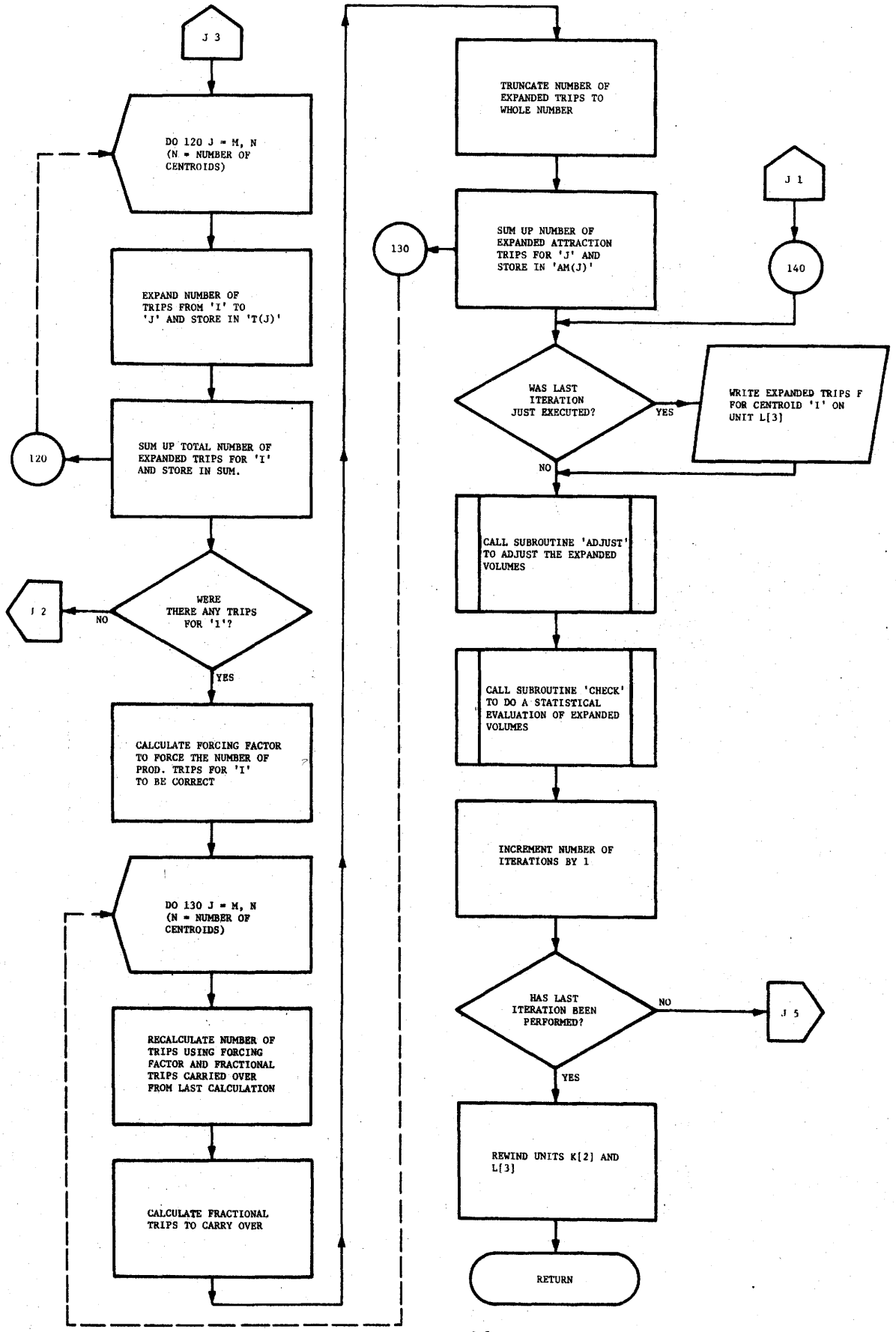

C 4
C 3
DO 190 J = 1, N
(N = NUMBER OF ZONES)
IS ORIGIN I LARGER THAN 'LAST; WHICH IS = TO THE LAST ZONE NUMBER
YES
SET SEPARATION 'KS(J)' = LARGEST INTERNAL SEPARATION + 2 (i.e. KS(J) = M + 2) TO INDICATE ORIGIN IS EXTERNAL
NO
IS DESTINATION J LARGER THAN 'LAST'?
YES
SET KS(J) = M + 1 TO INDICATE THE DESTINATION ZONE IS EXTERNAL
NO
ARE BOTH ORIGIN AND DESTINATION LARGER THAN 'LAST'?
YES
SET SEPARATION KS(J) = M + 3 TO INDICATE BOTH ENDS OF TRIP ARE EXTERNAL
NO
IS EITHER ORIGIN OR DESTINATION LARGER THAN 'LAST'?
YES
NO
IS SEPARATION 'KS(J)'LESS THAN 1?
YES
NO
IS SEPARATION 'KS(J)'LESS THAN LARGEST SEPARATION 'M'?
YES
NO
200
PRINT OUT IN REVISION ON TABLE I - ORIGIN
J - DESTINATION
KS(J) - SEPARATION
M - NEW SEPARATION FOR 'KS(J)'
RESET SEPARATION 'KS(J)' TO LARGEST SEPARATION 'M'
170
PRINT OUT IN REVISION TABLE I - ORIGIN
J - DESTINATION
KS(J) - SEPARATION
ONE - NEW SEPARATION NUMBER
RESET SEPARATION 'KS(J)' TO 1
185
CALCULATE ACCESSIBILITY FACTOR FOR PRODUCTION ZONES: PZ(I) = PZ(I) + (KS(J)/FN)
CALCULATE ACCESSIBILITY FACTOR FOR ATTRACTION ZONES AZ(J) = AZ(J) + (KS(J)/FN)
SUM UP NUMBER OF ZONE PAIRS WITH SEPARATION KS(J) i.e. S(KS(J)) = S(KS(J)) + 1.0
190
WRITE REVISED SEPARATIONS ON UNIT D = 4
C 5


C 5
HAS ALL SEPARATION CARDS BEEN READ?
YES
NO
PRINT OUT MESSAGE SAYING CARDS ARE OUT OF ORDER
210
PRINT OUT CARD THAT HAS BEEN DISREGARDED
READ NEXT DATA CARD
END =
IS CARD A SEPARATION CARD?
YES
NO
PRINT OUT LARGEST SEPARATION NUMBER
220
PRINT OUT TABLE SHOWING ACCESSIBILITY FACTORS FOR ALL ZONES
END FILE UNIT D
REWIND UNITS D AND A
RESTORE N TO ITS ORIGINAL VALUE OF NUMBER OF ZONES
RETURN

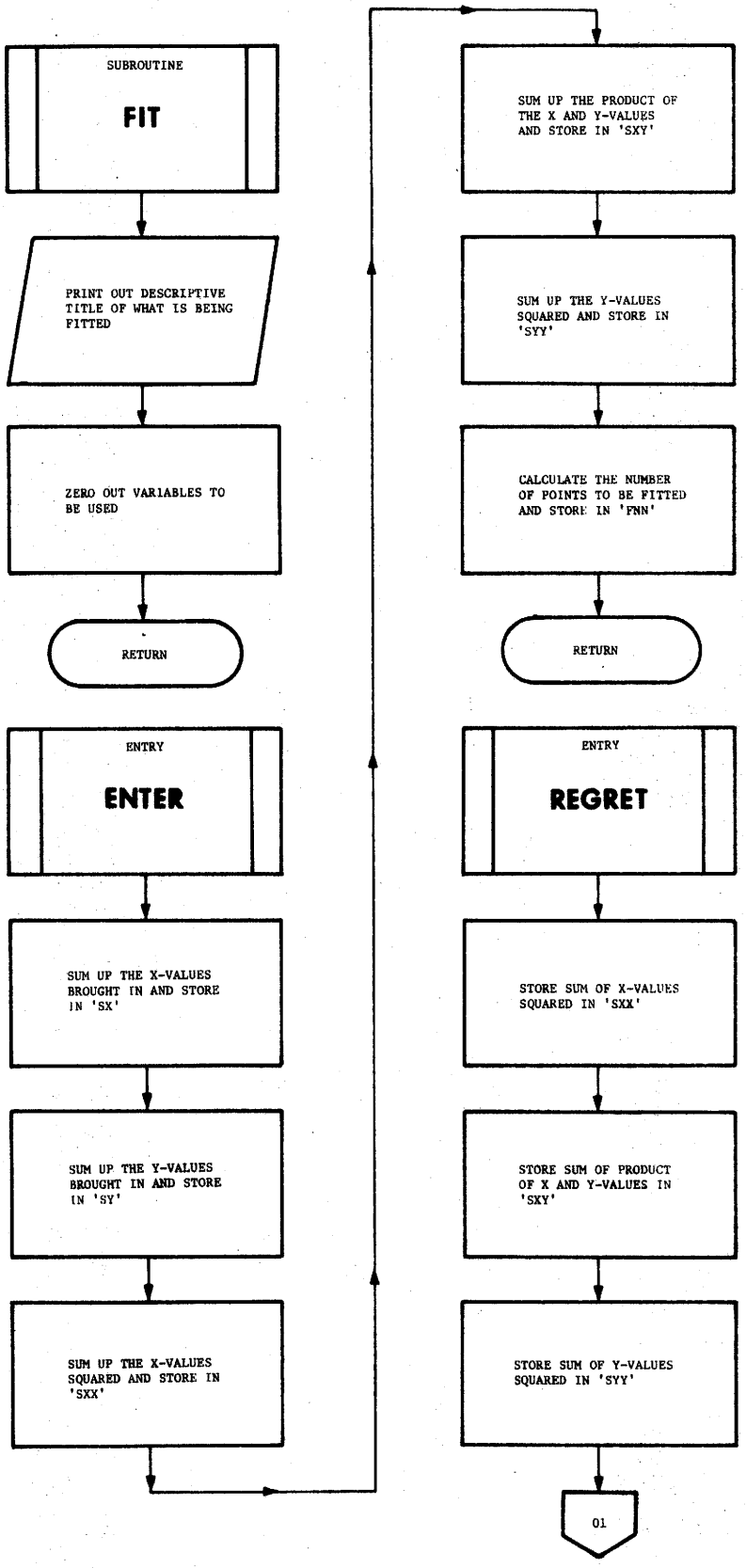

SUBROUTINE
EQUATE
CALL NAME TO PRINT OUT NAME OF SUB-ROUTINE BEING EXECUTED ('EQUATE')
INITIALIZE VARIABLE MR = 0 AND ZERO OUT ARRAY KS
REREAD LAST DATA CARD READ
105
IS CARD AN EQUATE CARD?
NO
YES
REREAD CARD TO GET L - SECTOR NUMBER K(1) - ZONES TO BE EQUATED TO SECTOR L
100
READ NEXT DATA CARD
END =
140
IS LARGEST SECTOR NUMBER GREATER THAN 0? WERE EQUATE CARDS READ IN?
YES
NO
D 2
WRITE OUT MESSAGE STATING THAT EQUATING DATA WAS NOT FOUND.
IS SECTOR NUMBER LESS THAN 1? i.e. IS L < 1?
YES
NO
IS SECTOR NUMBER 'L' LARGER THAN VARIABLE 'MR'
NO
YES
SET MR = L
'MR' WILL BE THE VALUE OF THE LARGEST SECTOR NUMBER
DO 130
I = 1, 14
IS ZONE K(1) WITHIN THE SPECIFIED RANGE? i.e. 1 ≤ K(I) ≤ N
NO
ES
SET M = ZONE K(1) TO BE USED AS SUBSCRIPT IN EQUATING ARRAY KS
HAS ZONE K(I) BEEN PREVIOUSLY EQUATED? i.e. IS KS(M) ≠ 0?
NO
YES
PRINT OUT ZONE NUMBER M, SECTOR NUMBER 'M' IS EQUATED TO AND SECTOR NUMBER L, THAT M IS TO BE RESET TO.
130
EQUATE ZONE M TO SECTOR L i.e. SET KS(M) = L
SET LARGEST SECTOR NUMBER, MR, = 1 AND SET NUMBER OF POSSIBLE 2-WAY SECTOR COMBINATIONS, NR, = 1
D 1

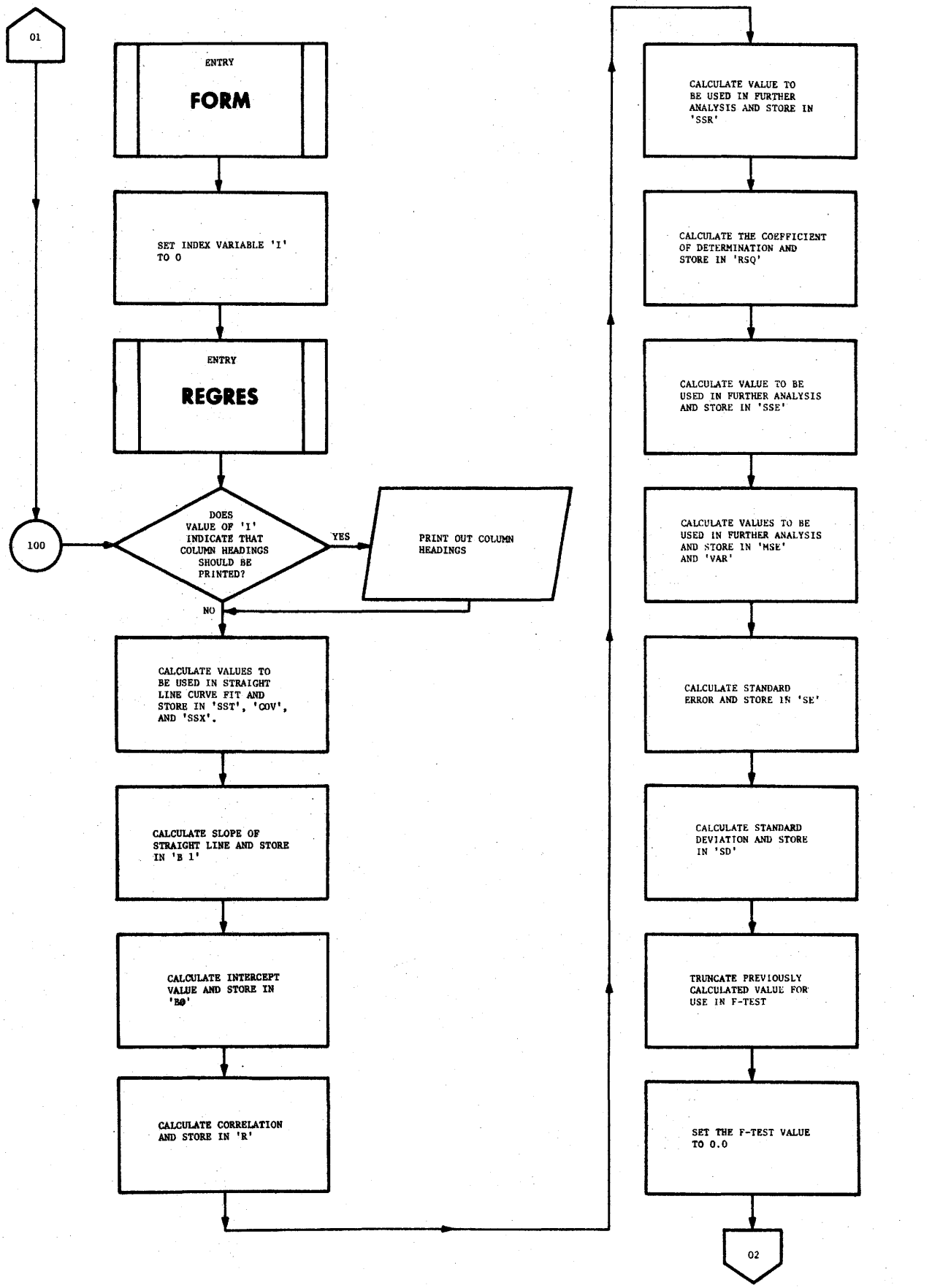

D 1
DO 145 I = 1, N
(N = NUMBER OF ZONES)
145
EQUATE ALL ZONES
BY DEFAULT TO SECTOR 1
WRITE OUT MESSAGE
STATING ALL ZONES
EQUATED TO SECTOR
1 WERE BY DEFAULT
RETURN
D 2
150
SET POINTER L = 0
DO 160 I = 1, N
(N = NUMBER OF ZONES)
HAS ZONE
I BEEN EQUATED
TO A SECTOR? i.e.
IS KS(I) ≠ 0?
YES
NO
SET POINTER L = 1
EQUATE ZONE I TO
SECTOR NUMBER 1
GREATER THAN LARGEST
SECTOR NUMBER, MR + 1
160
PRINT OUT SECTOR
NUMBER AND ZONES THAT
WERE EQUATED TO IT.
WAS ANY
ZONE EQUATED
TO 1 GREATER THAN
LARGEST SECTOR
NUMBER, MR + 1?
NO
YES
SET LARGEST SECTOR
NUMBER, MR, TO 1
GREATER THAN ITSELF,
i.e. MR = MR + 1
WRITE OUT MESSAGE
STATING ALL ZONES
EQUATED TO SECTOR,
MR, WERE BY DEFAULT
170
CALCULATE NUMBER
OF POSSIBLE 2-WAY
SECTOR COMBINATIONS
AND STORE IN VARIABLE
'NR'
RETURN

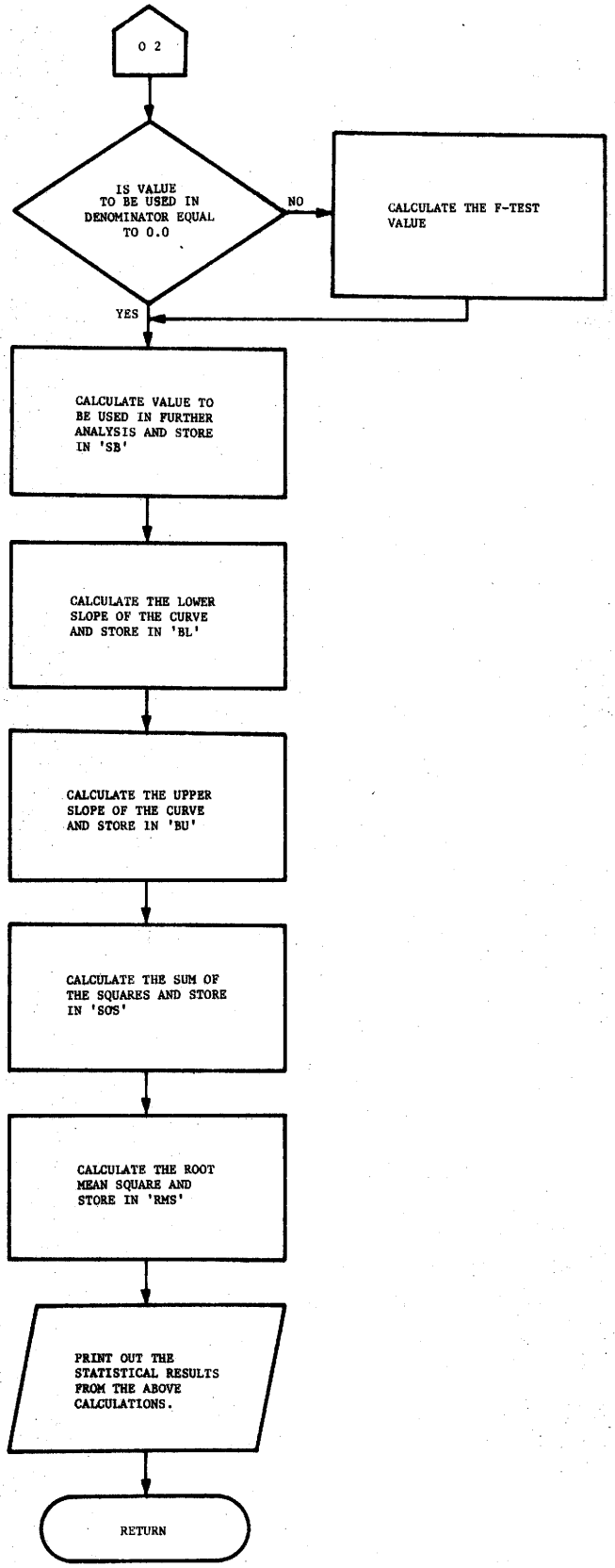

SUBROUTINE
EXPAND
CALL SUBROUTINE 'NAME' TO PRINT NAME OF SUB-ROUTINE BEING EXECUTED 'EXPAND'
REWIND UNITS K[2] AND L[3]
READ HEADER RECORD OFF UNIT K[2] WHICH CONTAINS TRIP MATRIX
WRITE HEADER RECORD ON UNIT L[3] WHICH WILL CONTAIN EXPANDED TRIP MATRIX
INITIALIZE VARIABLE 'LOOP' TO 1 THIS VARIABLE INDICATES WHAT ITERATION IS BEING PERFORMED
WERE GENERATION CARDS READ IN?
YES
NO
J 5
DO 80 I = 1, M (M=NUMBER OF CENTROIDS WITHOUT GENERATION CARDS)
80
SET ATTRACTIONS FOR CENTROID (I - 1) TO 0.0
CLEAR ARRAY 'AM'; SET 'RES' = 0.0; REWIND UNIT K[2]
90
READ HEADER RECORD OFF UNIT K[2] WHICH CONTAINS TRIP MATRIX
DO 140 I = 1, N (N = NUMBER OF CENTROIDS)
READ TRIPS FOR CENTROID 'I' OFF UNIT K[2]
ARE THERE ANY RELATIVE ATTRACTIONS FOR CENTROID 'I'?
YES
NO
J 1
J 1
J 2
DID CENTROID 'I' PRODUCE ANY TRIPS?
YES
NO
CLEAR OUT THE ARRAY CONTAINING TRIPS FOR CENTROID 'I'
100
110
INITIALIZE THE VARIABLE 'SUM' TO 0.0
J 3
1

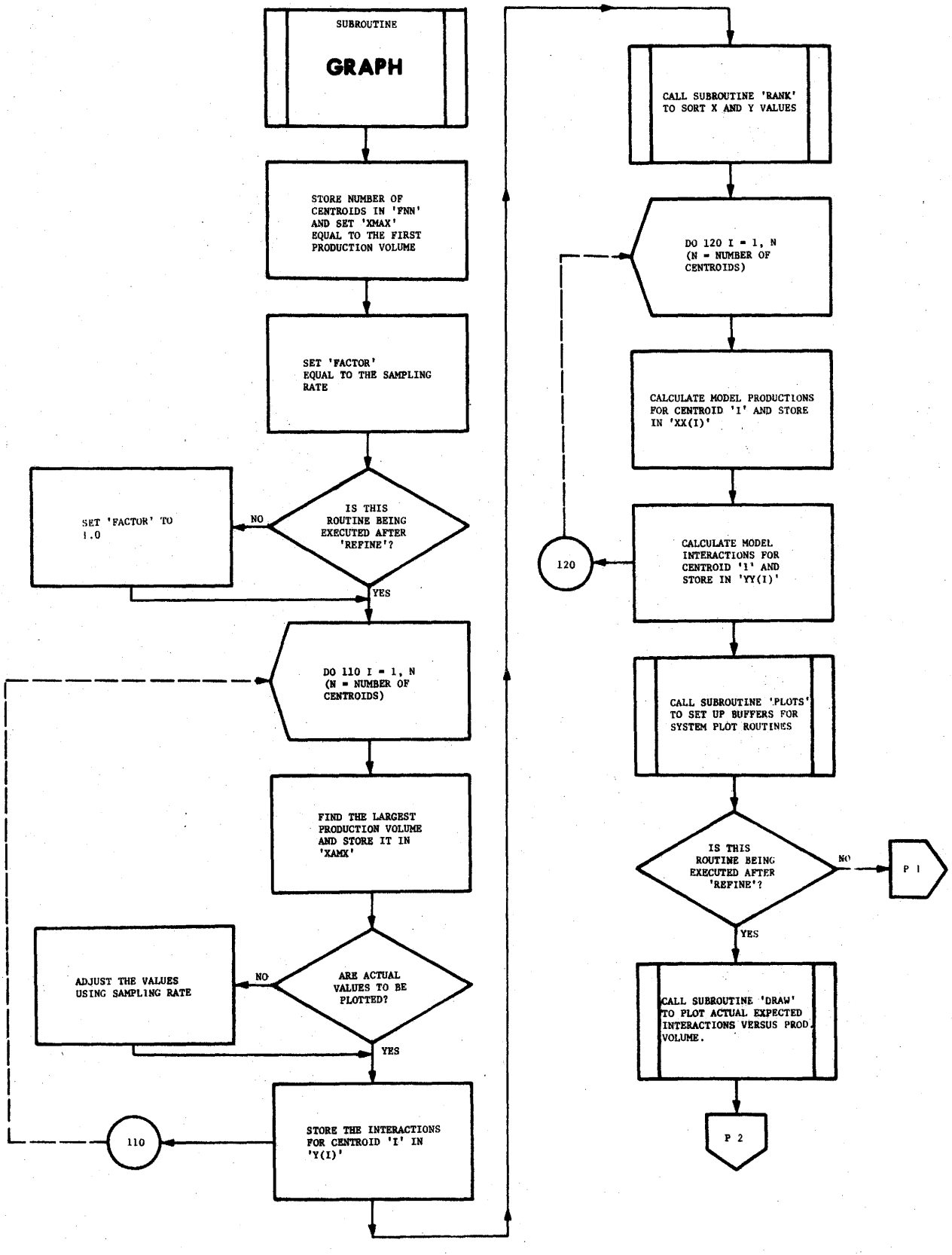

J 3
DO 120 J = M, N
(N = NUMBER OF
CENTROIDS)
EXPAND NUMBER OF
TRIPS FROM 'I' TO
'J' AND STORE IN 'T(J)'
SUM UP TOTAL NUMBER OF
EXPANDED TRIPS FOR 'I'
AND STORE IN SUM.
120
WERE
THERE ANY TRIPS
FOR 'I'?
NO
J 2
YES
CALCULATE FORCING FACTOR
TO FORCE THE NUMBER OF
PROD. TRIPS FOR 'I'
TO BE CORRECT
DO 130 J = M, N
(N = NUMBER OF
CENTROIDS)
RECALCULATE NUMBER OF
TRIPS USING FORCING
FACTOR AND FRACTIONAL
TRIPS CARRIED OVER
FROM LAST CALCULATION
CALCULATE FRACTIONAL
TRIPS TO CARRY OVER
TRUNCATE NUMBER OF
EXPANDED TRIPS TO
WHOLE NUMBER
SUM UP NUMBER OF
EXPANDED ATTRACTION
TRIPS FOR 'J' AND
STORE IN 'AM(J)'
130
J 1
140
WAS LAST
ITERATION
JUST EXECUTED?
YES
WRITE EXPANDED TRIPS F
FOR CENTROID 'I' ON
UNIT L[3]
NO
CALL SUBROUTINE 'ADJUST'
TO ADJUST THE EXPANDED
VOLUMES
CALL SUBROUTINE 'CHECK'
TO DO A STATISTICAL
EVALUATION OF EXPANDED
VOLUMES
INCREMENT NUMBER OF
ITERATIONS BY 1
HAS LAST
ITERATION BEEN
PERFORMED?
NO
J 5
YES
REWIND UNITS K[2] AND
L[3]
RETURN

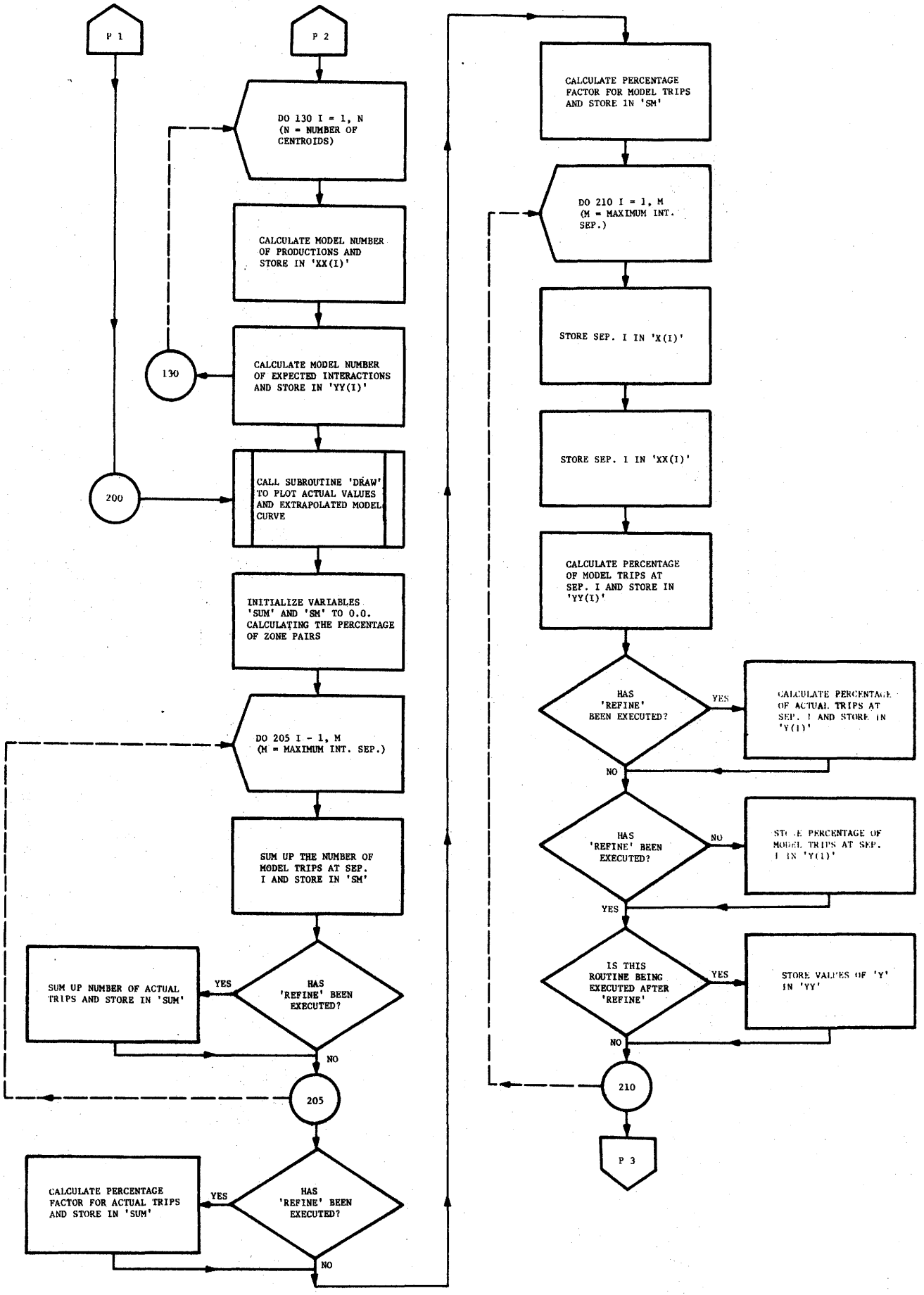
FIT
ENTER
REGRET
FORM
REGRE

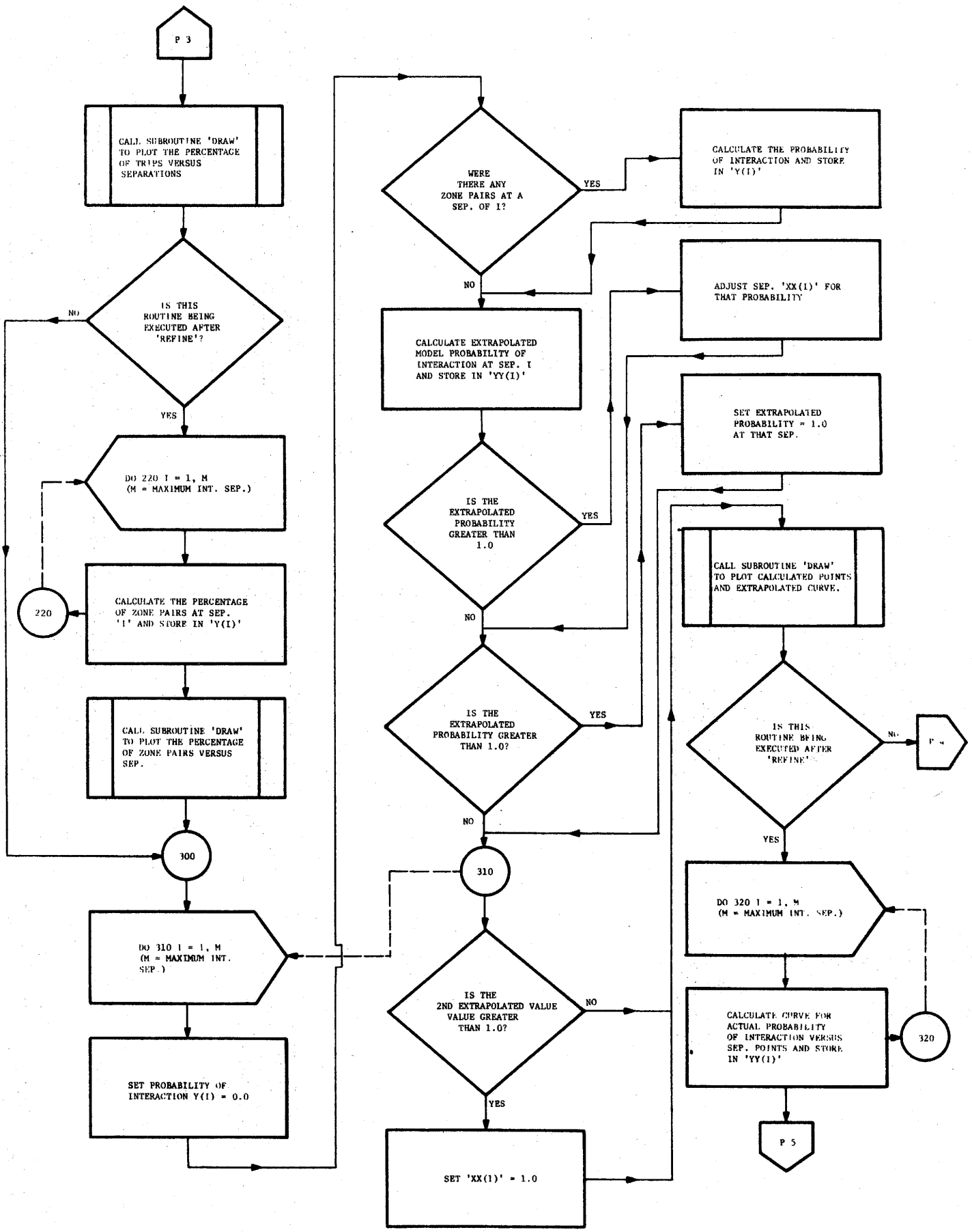


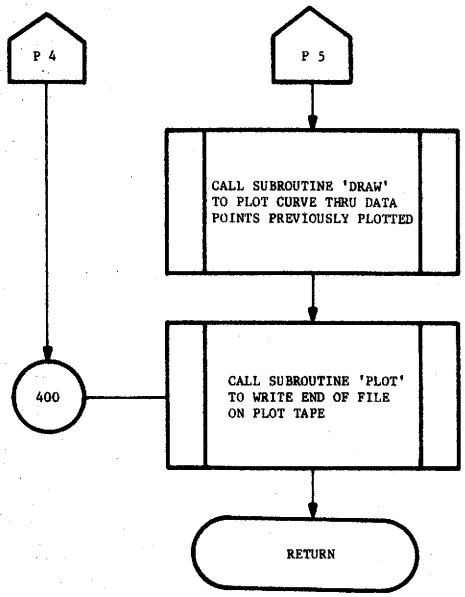

01
ENTRY
FORM
SET INDEX VARIABLE 'I' TO 0
ENTRY
REGRES
100
DOES VALUE OF 'I' INDICATE THAT COLUMN HEADINGS SHOULD BE PRINTED?
YES
PRINT OUT COLUMN HEADINGS
NO
CALCULATE VALUES TO BE USED IN STRAIGHT LINE CURVE FIT AND STORE IN 'SST', 'COV', AND 'SSX'.
CALCULATE SLOPE OF STRAIGHT LINE AND STORE IN 'B 1'
CALCULATE INTERCEPT VALUE AND STORE IN 'B0'
CALCULATE CORRELATION AND STORE IN 'R'
CALCULATE VALUE TO BE USED IN FURTHER ANALYSIS AND STORE IN 'SSR'
CALCULATE THE COEFFICIENT OF DETERMINATION AND STORE IN 'RSQ'
CALCULATE VALUE TO BE USED IN FURTHER ANALYSIS AND STORE IN 'SSE'
CALCULATE VALUES TO BE USED IN FURTHER ANALYSIS AND STORE IN 'MSE' AND 'VAR'
CALCULATE STANDARD ERROR AND STORE IN 'SE'
CALCULATE STANDARD DEVIATION AND STORE IN 'SD'
TRUNCATE PREVIOUSLY CALCULATED VALUE FOR USE IN F-TEST
SET THE F-TEST VALUE TO 0.0
02

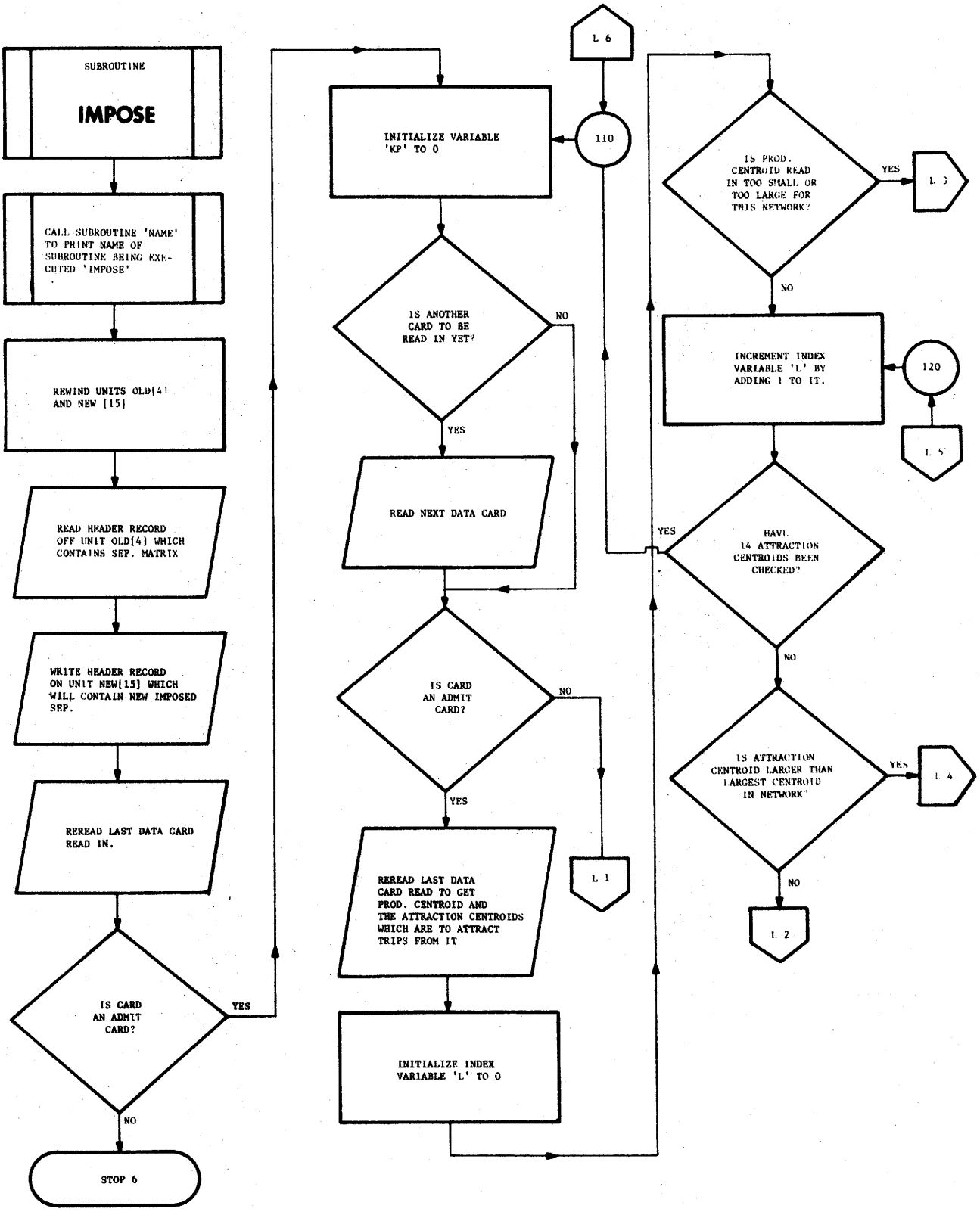

0 2
IS VALUE TO BE USED IN DENOMINATOR EQUAL TO 0.0
NO
CALCULATE THE F-TEST VALUE
YES
CALCULATE VALUE TO BE USED IN FURTHER ANALYSIS AND STORE IN 'SB'
CALCULATE THE LOWER SLOPE OF THE CURVE AND STORE IN 'BL'
CALCULATE THE UPPER SLOPE OF THE CURVE AND STORE IN 'BU'
CALCULATE THE SUM OF THE SQUARES AND STORE IN 'SOS'
CALCULATE THE ROOT MEAN SQUARE AND STORE IN 'RMS'
PRINT OUT THE STATISTICAL RESULTS FROM THE ABOVE CALCULATIONS.
RETURN

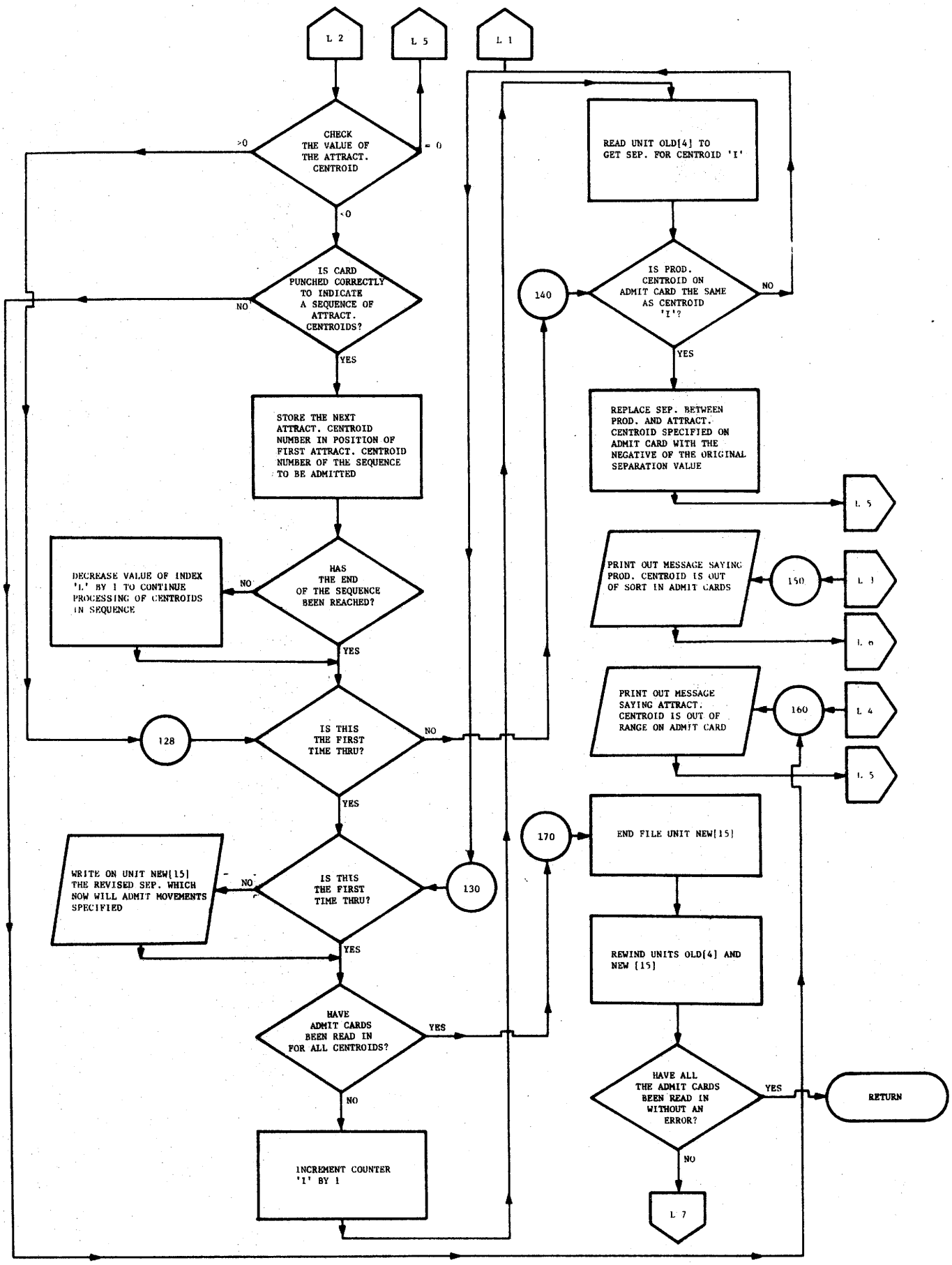

SUBROUTINE
GRAPH
STORE NUMBER OF CENTROIDS IN 'FNN' AND SET 'XMAX' EQUAL TO THE FIRST PRODUCTION VOLUME
SET 'FACTOR' EQUAL TO THE SAMPLING RATE
IS THIS ROUTINE BEING EXECUTED AFTER 'REFINE'?
NO
SET 'FACTOR' TO 1.0
YES
DO 110 I = 1, N (N = NUMBER OF CENTROIDS)
FIND THE LARGEST PRODUCTION VOLUME AND STORE IT IN 'XAMX'
ARE ACTUAL VALUES TO BE PLOTTED?
NO
ADJUST THE VALUES USING SAMPLING RATE
YES
STORE THE INTERACTIONS FOR CENTROID 'I' IN 'Y(I)'
110
CALL SUBROUTINE 'RANK' TO SORT X AND Y VALUES
DO 120 I = 1, N (N = NUMBER OF CENTROIDS)
CALCULATE MODEL PRODUCTIONS FOR CENTROID 'I' AND STORE IN 'XX(I)'
CALCULATE MODEL INTERACTIONS FOR CENTROID 'I' AND STORE IN 'YY(I)'
120
CALL SUBROUTINE 'PLOTS' TO SET UP BUFFERS FOR SYSTEM PLOT ROUTINES
IS THIS ROUTINE BEING EXECUTED AFTER 'REFINE'?
NO
P 1
YES
CALL SUBROUTINE 'DRAW' TO PLOT ACTUAL EXPECTED INTERACTIONS VERSUS PROD. VOLUME.
P 2

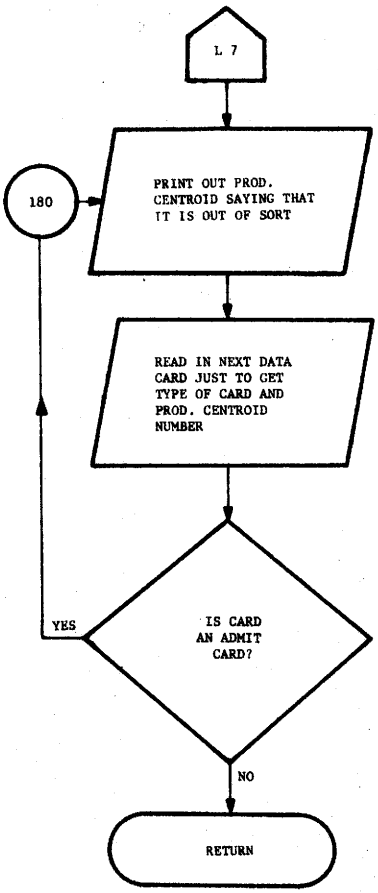

P 1
P 2
DO 130 I = 1, N (N = NUMBER OF CENTROIDS)
CALCULATE MODEL NUMBER OF PRODUCTIONS AND STORE IN 'XX(I)'
CALCULATE MODEL NUMBER OF EXPECTED INTERACTIONS AND STORE IN 'YY(I)'
130
200
CALL SUBROUTINE 'DRAW' TO PLOT ACTUAL VALUES AND EXTRAPOLATED MODEL CURVE
INITIALIZE VARIABLES 'SUM' AND 'SM' TO 0.0. CALCULATING THE PERCENTAGE OF ZONE PAIRS
DO 205 I = 1, M (M = MAXIMUM INT. SEP.)
SUM UP THE NUMBER OF MODEL TRIPS AT SEP. I AND STORE IN 'SM'
HAS 'REFINE' BEEN EXECUTED?
YES
SUM UP NUMBER OF ACTUAL TRIPS AND STORE IN 'SUM'
NO
205
HAS 'REFINE' BEEN EXECUTED?
YES
CALCULATE PERCENTAGE FACTOR FOR ACTUAL TRIPS AND STORE IN 'SUM'
NO
CALCULATE PERCENTAGE FACTOR FOR MODEL TRIPS AND STORE IN 'SM'
DO 210 I = 1, M (M = MAXIMUM INT. SEP.)
STORE SEP. I IN 'X(I)'
STORE SEP. I IN 'XX(I)'
CALCULATE PERCENTAGE OF MODEL TRIPS AT SEP. I AND STORE IN 'YY(I)'
HAS 'REFINE' BEEN EXECUTED?
YES
CALCULATE PERCENTAGE OF ACTUAL TRIPS AT SEP. I AND STORE IN 'Y(I)'
NO
HAS 'REFINE' BEEN EXECUTED?
NO
STORE PERCENTAGE OF MODEL TRIPS AT SEP. I IN 'Y(I)'
YES
IS THIS ROUTINE BEING EXECUTED AFTER 'REFINE'
YES
STORE VALUES OF 'Y' IN 'YY'
NO
210
P 3

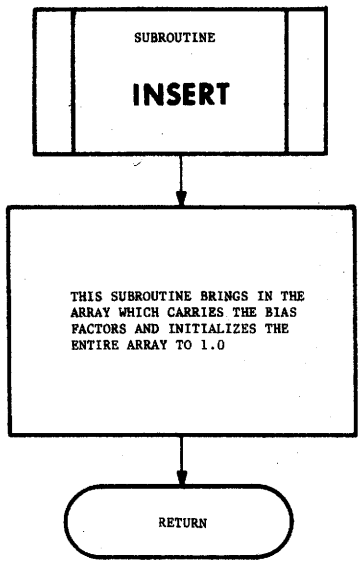

P 3
CALL SUBROUTINE 'DRAW' TO PLOT THE PERCENTAGE OF TRIPS VERSUS SEPARATIONS
IS THIS ROUTINE BEING EXECUTED AFTER 'REFINE'?
NO
YES
DO 220 I = 1, M (M = MAXIMUM INT. SEP.)
220
CALCULATE THE PERCENTAGE OF ZONE PAIRS AT SEP. 'I' AND STORE IN 'Y(I)'
CALL SUBROUTINE 'DRAW' TO PLOT THE PERCENTAGE OF ZONE PAIRS VERSUS SEP.
300
DO 310 I = 1, M (M = MAXIMUM INT. SEP.)
SET PROBABILITY OF INTERACTION Y(I) = 0.0
WERE THERE ANY ZONE PAIRS AT A SEP. OF I?
YES
NO
CALCULATE THE PROBABILITY OF INTERACTION AND STORE IN 'Y(I)'
ADJUST SEP. 'XX(I)' FOR THAT PROBABILITY
CALCULATE EXTRAPOLATED MODEL PROBABILITY OF INTERACTION AT SEP. I AND STORE IN 'YY(I)'
IS THE EXTRAPOLATED PROBABILITY GREATER THAN 1.0
YES
NO
SET EXTRAPOLATED PROBABILITY = 1.0 AT THAT SEP.
IS THE EXTRAPOLATED PROBABILITY GREATER THAN 1.0?
YES
NO
310
IS THE 2ND EXTRAPOLATED VALUE VALUE GREATER THAN 1.0?
NO
YES
SET 'XX(1)' = 1.0
CALL SUBROUTINE 'DRAW' TO PLOT CALCULATED POINTS AND EXTRAPOLATED CURVE.
IS THIS ROUTINE BEING EXECUTED AFTER 'REFINE'
NO
YES
DO 320 I = 1, M (M = MAXIMUM INT. SEP.)
CALCULATE CURVE FOR ACTUAL PROBABILITY OF INTERACTION VERSUS SEP. POINTS AND STORE IN 'YY(I)'
320
P 5

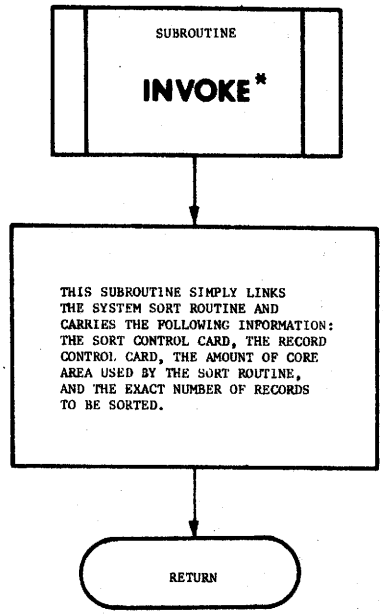

P 4
P 5
CALL SUBROUTINE 'DRAW'
TO PLOT CURVE THRU DATA
POINTS PREVIOUSLY PLOTTED
400
CALL SUBROUTINE 'PLOT'
TO WRITE END OF FILE
ON PLOT TAPE
RETURN


SUBROUTINE
IMPOSE
CALL SUBROUTINE 'NAME' TO PRINT NAME OF SUBROUTINE BEING EXECUTED 'IMPOSE'
REWIND UNITS OLD[4] AND NEW [15]
READ HEADER RECORD OFF UNIT OLD[4] WHICH CONTAINS SEP. MATRIX
WRITE HEADER RECORD ON UNIT NEW[15] WHICH WILL CONTAIN NEW IMPOSED SEP.
REREAD LAST DATA CARD READ IN.
IS CARD AN ADMIT CARD?
YES
NO
STOP 6
L 6
110
INITIALIZE VARIABLE 'KP' TO 0
IS ANOTHER CARD TO BE READ IN YET?
NO
YES
READ NEXT DATA CARD
IS CARD AN ADMIT CARD?
NO
YES
REREAD LAST DATA CARD READ TO GET PROD. CENTROID AND THE ATTRACTION CENTROIDS WHICH ARE TO ATTRACT TRIPS FROM IT
L 1
INITIALIZE INDEX VARIABLE 'L' TO 0
IS PROD. CENTROID READ IN TOO SMALL OR TOO LARGE FOR THIS NETWORK?
YES
L 3
NO
INCREMENT INDEX VARIABLE 'L' BY ADDING 1 TO IT.
120
L 5
HAVE 14 ATTRACTION CENTROIDS BEEN CHECKED?
YES
NO
IS ATTRACTION CENTROID LARGER THAN LARGEST CENTROID IN NETWORK?
YES
L 4
NO
L 2

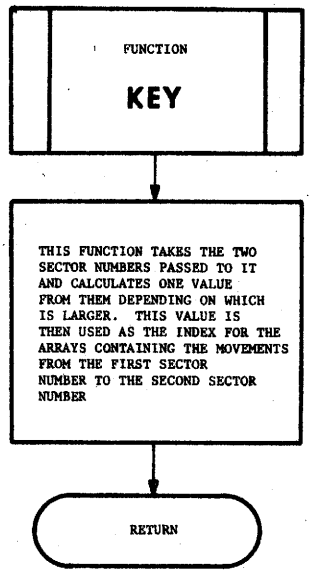

L 2
L 5
L 1
CHECK THE VALUE OF THE ATTRACT. CENTROID
>0
= 0
<0
IS CARD PUNCHED CORRECTLY TO INDICATE A SEQUENCE OF ATTRACT. CENTROIDS?
NO
YES
STORE THE NEXT ATTRACT. CENTROID NUMBER IN POSITION OF FIRST ATTRACT. CENTROID NUMBER OF THE SEQUENCE TO BE ADMITTED
DECREASE VALUE OF INDEX 'L' BY 1 TO CONTINUE PROCESSING OF CENTROIDS IN SEQUENCE
HAS THE END OF THE SEQUENCE BEEN REACHED?
NO
YES
128
IS THIS THE FIRST TIME THRU?
NO
YES
WRITE ON UNIT NEW[15] THE REVISED SEP. WHICH NOW WILL ADMIT MOVEMENTS SPECIFIED
IS THIS THE FIRST TIME THRU?
NO
YES
130
HAVE ADMIT CARDS BEEN READ IN FOR ALL CENTROIDS?
YES
NO
INCREMENT COUNTER 'I' BY 1
READ UNIT OLD[4] TO GET SEP. FOR CENTROID 'I'
140
IS PROD. CENTROID ON ADMIT CARD THE SAME AS CENTROID 'I'?
NO
YES
REPLACE SEP. BETWEEN PROD. AND ATTRACT. CENTROID SPECIFIED ON ADMIT CARD WITH THE NEGATIVE OF THE ORIGINAL SEPARATION VALUE
L 5
PRINT OUT MESSAGE SAYING PROD. CENTROID IS OUT OF SORT IN ADMIT CARDS
150
L 3
L 6
PRINT OUT MESSAGE SAYING ATTRACT. CENTROID IS OUT OF RANGE ON ADMIT CARD
160
L 4
L 5
170
END FILE UNIT NEW[15]
REWIND UNITS OLD[4] AND NEW [15]
HAVE ALL THE ADMIT CARDS BEEN READ IN WITHOUT AN ERROR?
YES
RETURN
NO
L 7

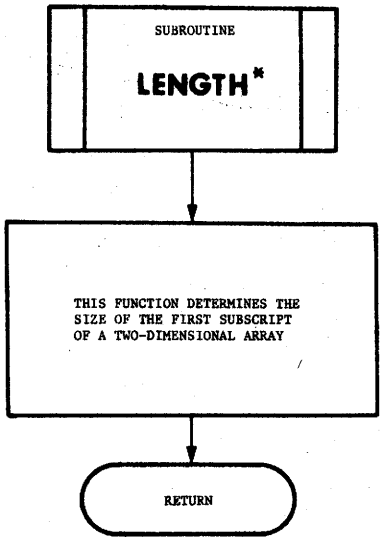

L 7
180
PRINT OUT PROD. CENTROID SAYING THAT IT IS OUT OF SORT
READ IN NEXT DATA CARD JUST TO GET TYPE OF CARD AND PROD. CENTROID NUMBER
IS CARD AN ADMIT CARD?
YES
NO
RETURN


SUBROUTINE
INSERT
THIS SUBROUTINE BRINGS IN THE ARRAY WHICH CARRIES THE BIAS FACTORS AND INITIALIZES THE ENTIRE ARRAY TO 1.0
RETURN

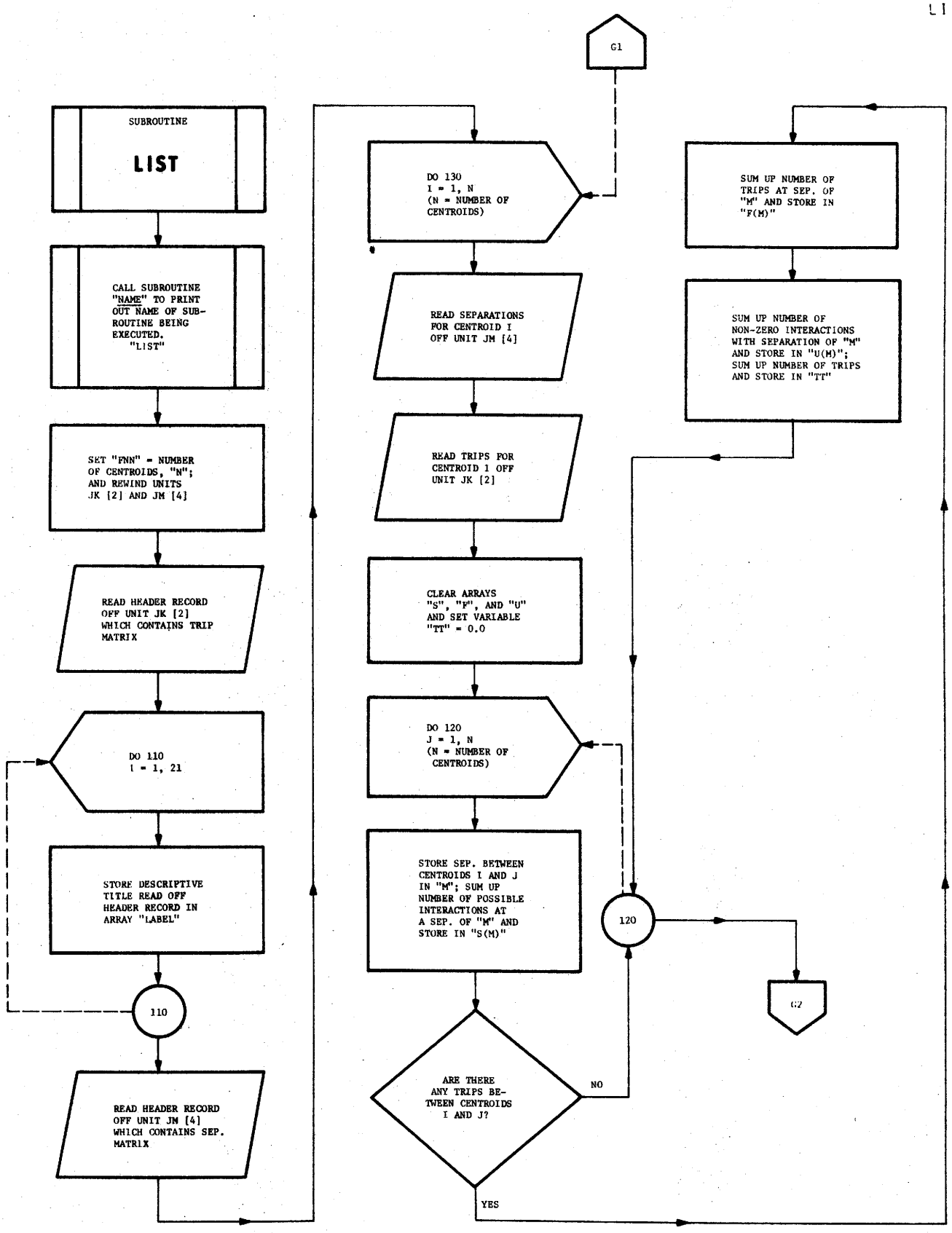


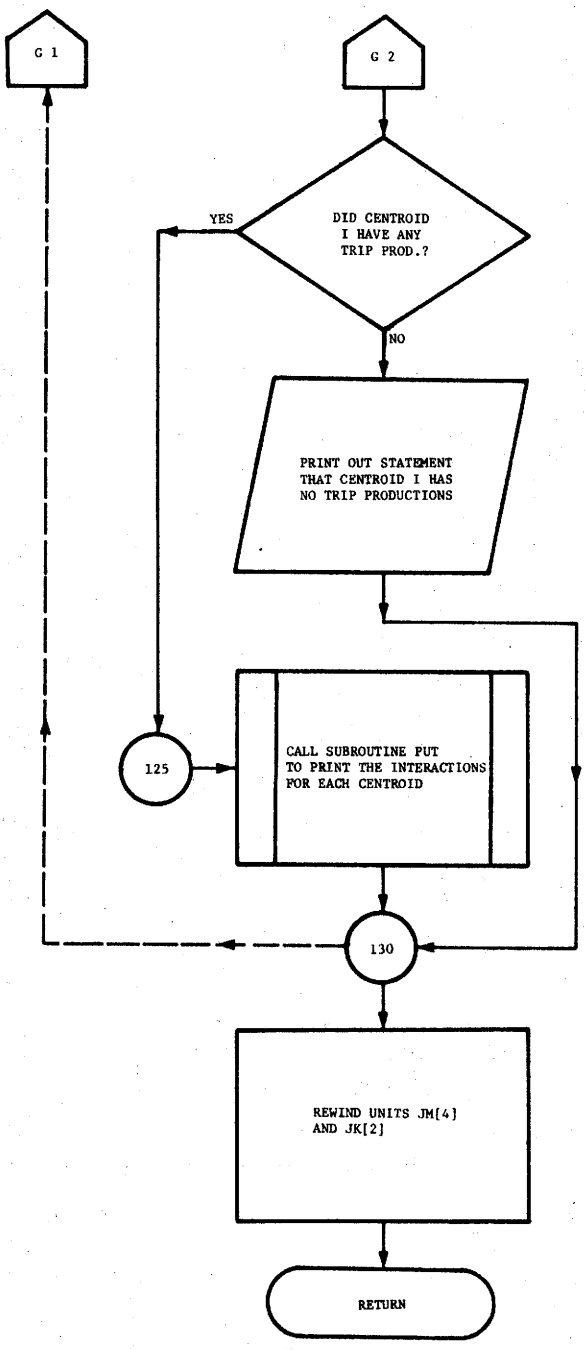
* AN ASSEMBLY LANGUAGE SUBPROGRAM

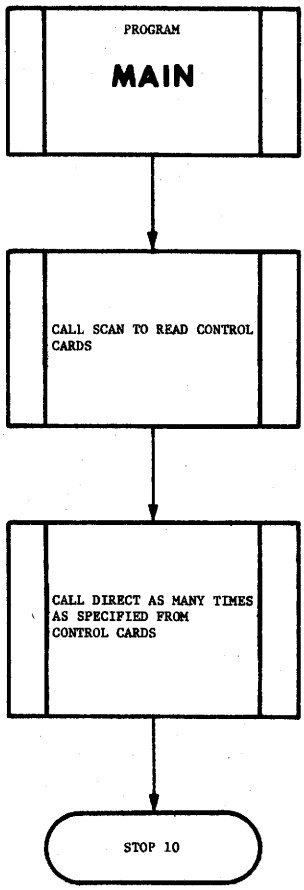

FUNCTION
KEY
THIS FUNCTION TAKES THE TWO SECTOR NUMBERS PASSED TO IT AND CALCULATES ONE VALUE FROM THEM DEPENDING ON WHICH IS LARGER. THIS VALUE IS THEN USED AS THE INDEX FOR THE ARRAYS CONTAINING THE MOVEMENTS FROM THE FIRST SECTOR NUMBER TO THE SECOND SECTOR NUMBER
RETURN

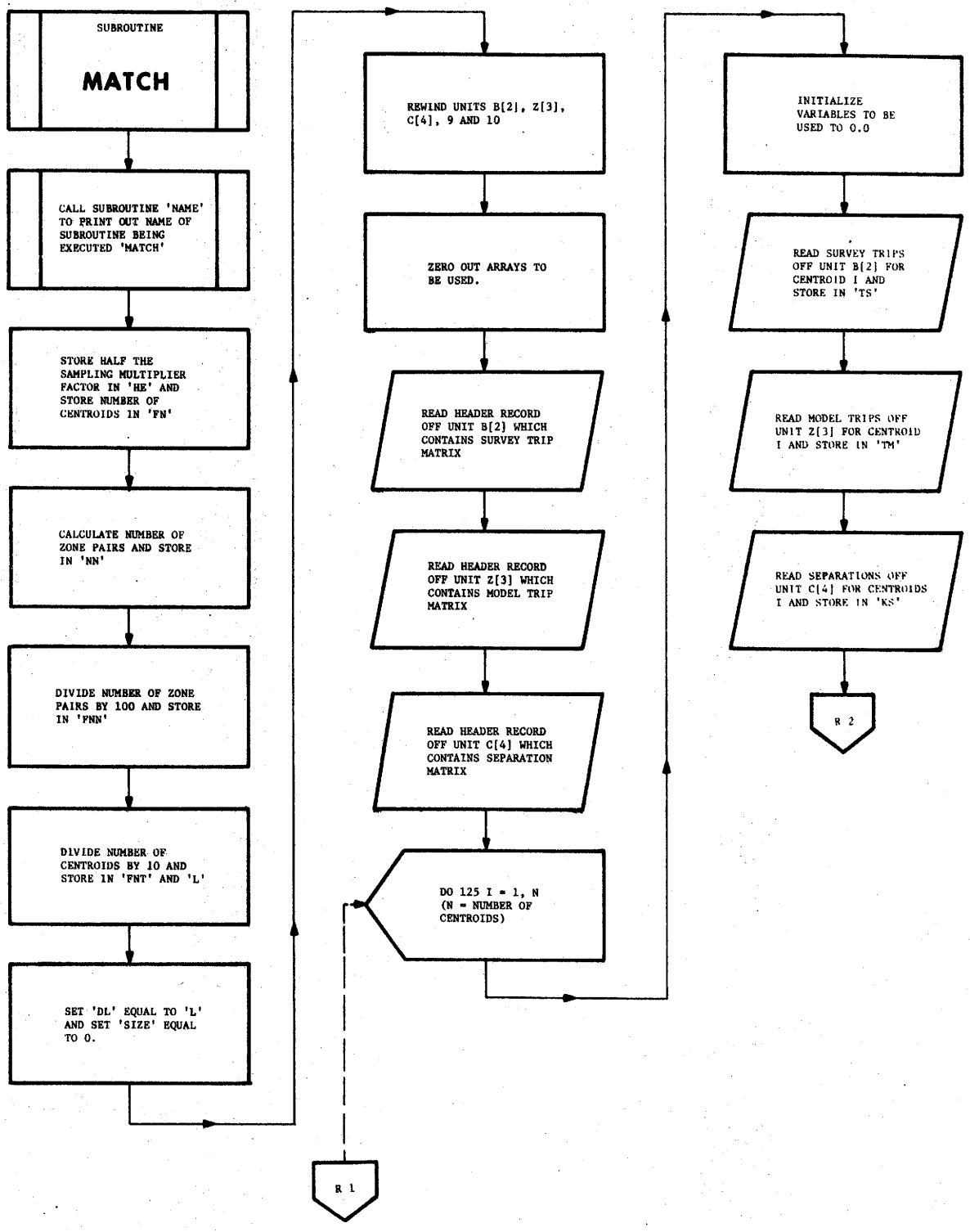


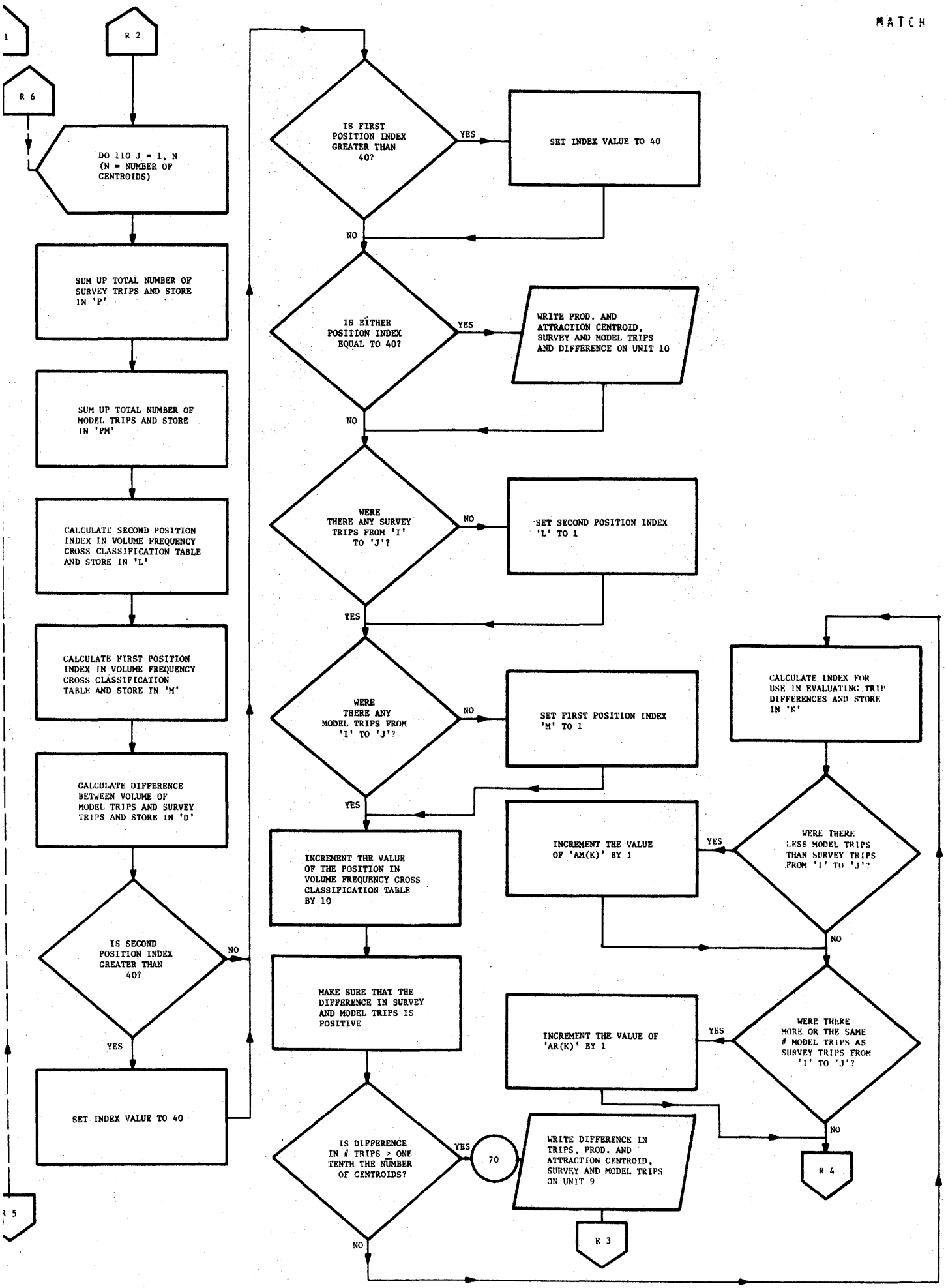
* AN ASSEMBLY LANGUAGE SUBPROGRAM

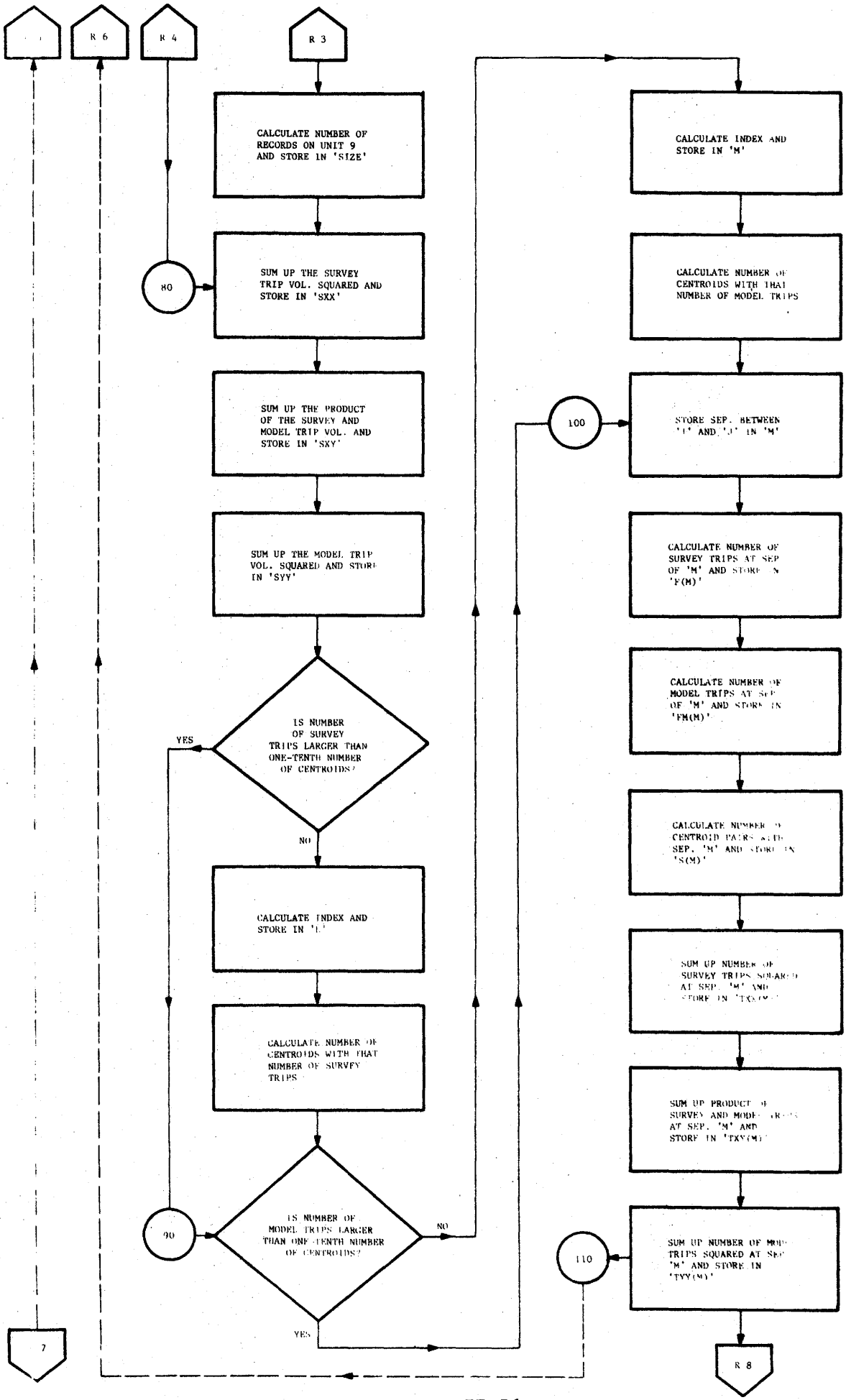

SUBROUTINE
LIST
CALL SUBROUTINE "NAME" TO PRINT OUT NAME OF SUB-ROUTINE BEING EXECUTED. "LIST"
SET "FNN" = NUMBER OF CENTROIDS, "N"; AND REWIND UNITS JK [2] AND JM [4]
READ HEADER RECORD OFF UNIT JK [2] WHICH CONTAINS TRIP MATRIX
DO 110 I = 1, 21
STORE DESCRIPTIVE TITLE READ OFF HEADER RECORD IN ARRAY "LABEL"
110
READ HEADER RECORD OFF UNIT JM [4] WHICH CONTAINS SEP. MATRIX
G1
DO 130 I = 1, N (N = NUMBER OF CENTROIDS)
READ SEPARATIONS FOR CENTROID I OFF UNIT JM [4]
READ TRIPS FOR CENTROID I OFF UNIT JK [2]
CLEAR ARRAYS "S", "F", AND "U" AND SET VARIABLE "TT" = 0.0
DO 120 J = 1, N (N = NUMBER OF CENTROIDS)
STORE SEP. BETWEEN CENTROIDS I AND J IN "M"; SUM UP NUMBER OF POSSIBLE INTERACTIONS AT A SEP. OF "M" AND STORE IN "S(M)"
ARE THERE ANY TRIPS BE-TWEEN CENTROIDS I AND J?
NO
YES
120
SUM UP NUMBER OF TRIPS AT SEP. OF "M" AND STORE IN "F(M)"
SUM UP NUMBER OF NON-ZERO INTERACTIONS WITH SEPARATION OF "M" AND STORE IN "U(M)"; SUM UP NUMBER OF TRIPS AND STORE IN "TT"
G2

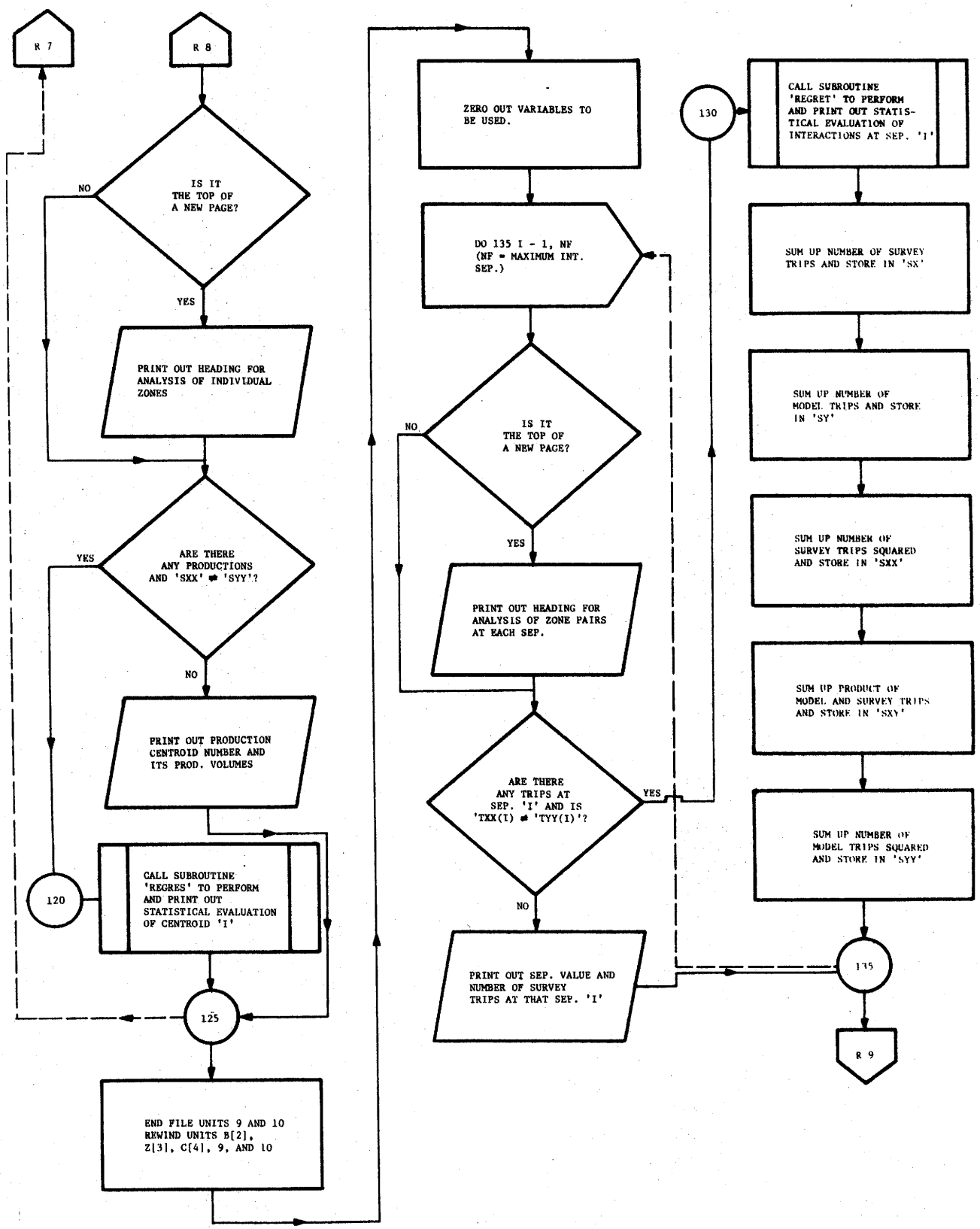

G 1
G 2
DID CENTROID I HAVE ANY TRIP PROD.?
YES
NO
PRINT OUT STATEMENT THAT CENTROID I HAS NO TRIP PRODUCTIONS
125
CALL SUBROUTINE PUT TO PRINT THE INTERACTIONS FOR EACH CENTROID
130
REWIND UNITS JM[4] AND JK[2]
RETURN

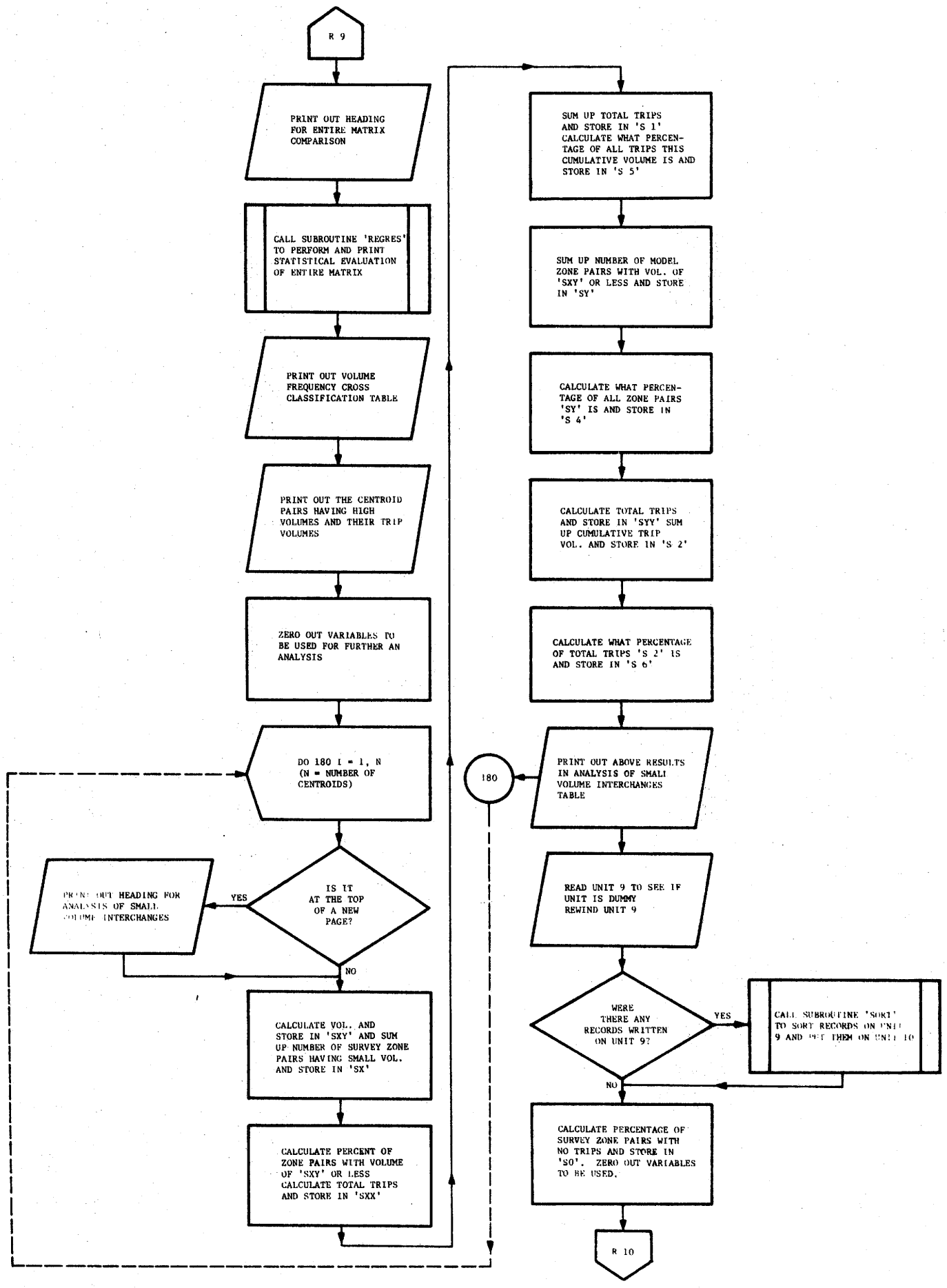

PROGRAM
MAIN
CALL SCAN TO READ CONTROL CARDS
CALL DIRECT AS MANY TIMES AS SPECIFIED FROM CONTROL CARDS
STOP 10

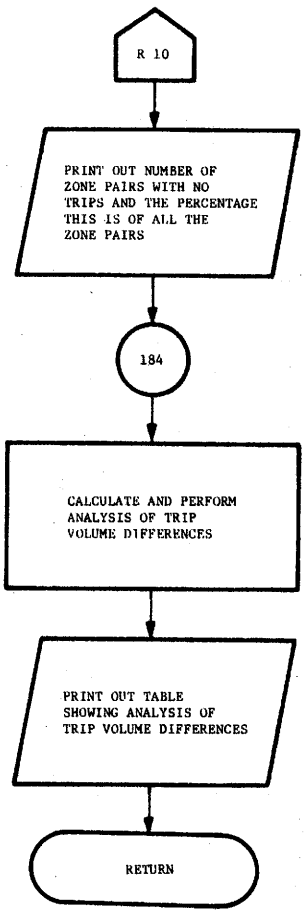

SUBROUTINE
MATCH
CALL SUBROUTINE 'NAME' TO PRINT OUT NAME OF SUBROUTINE BEING EXECUTED 'MATCH'
STORE HALF THE SAMPLING MULTIPLIER FACTOR IN 'HE' AND STORE NUMBER OF CENTROIDS IN 'FN'
CALCULATE NUMBER OF ZONE PAIRS AND STORE IN 'NN'
DIVIDE NUMBER OF ZONE PAIRS BY 100 AND STORE IN 'FNN'
DIVIDE NUMBER OF CENTROIDS BY 10 AND STORE IN 'FNT' AND 'L'
SET 'DL' EQUAL TO 'L' AND SET 'SIZE' EQUAL TO 0.
R 1
REWIND UNITS B[2], Z[3], C[4], 9 AND 10
ZERO OUT ARRAYS TO BE USED.
READ HEADER RECORD OFF UNIT B[2] WHICH CONTAINS SURVEY TRIP MATRIX
READ HEADER RECORD OFF UNIT Z[3] WHICH CONTAINS MODEL TRIP MATRIX
READ HEADER RECORD OFF UNIT C[4] WHICH CONTAINS SEPARATION MATRIX
DO 125 I = 1, N (N = NUMBER OF CENTROIDS)
INITIALIZE VARIABLES TO BE USED TO 0.0
READ SURVEY TRIPS OFF UNIT B[2] FOR CENTROID I AND STORE IN 'TS'
READ MODEL TRIPS OFF UNIT Z[3] FOR CENTROID I AND STORE IN 'TM'
READ SEPARATIONS OFF UNIT C[4] FOR CENTROIDS I AND STORE IN 'KS'
R 2

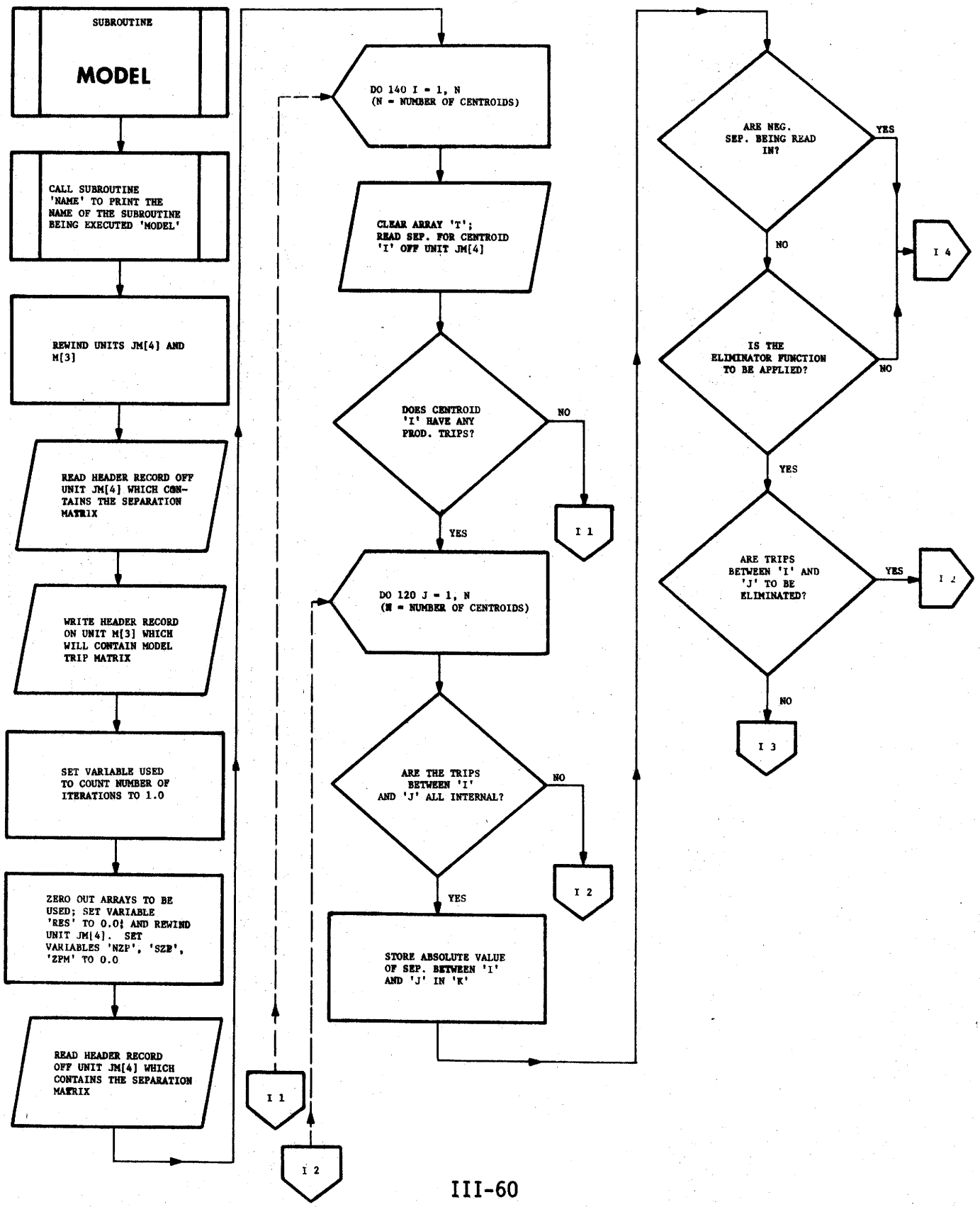

1
R 2
R 6
DO 110 J = 1, N
(N = NUMBER OF
CENTROIDS)
SUM UP TOTAL NUMBER OF
SURVEY TRIPS AND STORE
IN 'P'
SUM UP TOTAL NUMBER OF
MODEL TRIPS AND STORE
IN 'PM'
CALCULATE SECOND POSITION
INDEX IN VOLUME FREQUENCY
CROSS CLASSIFICATION TABLE
AND STORE IN 'L'
CALCULATE FIRST POSITION
INDEX IN VOLUME FREQUENCY
CROSS CLASSIFICATION
TABLE AND STORE IN 'M'
CALCULATE DIFFERENCE
BETWEEN VOLUME OF
MODEL TRIPS AND SURVEY
TRIPS AND STORE IN 'D'
IS SECOND
POSITION INDEX
GREATER THAN
40?
NO
YES
SET INDEX VALUE TO 40
R 5
IS FIRST
POSITION INDEX
GREATER THAN
40?
YES
SET INDEX VALUE TO 40
NO
IS EITHER
POSITION INDEX
EQUAL TO 40?
YES
WRITE PROD. AND
ATTRACTION CENTROID,
SURVEY AND MODEL TRIPS
AND DIFFERENCE ON UNIT 10
NO
WERE
THERE ANY SURVEY
TRIPS FROM 'I'
TO 'J'?
NO
SET SECOND POSITION INDEX
'L' TO 1
YES
WERE
THERE ANY
MODEL TRIPS FROM
'I' TO 'J'?
NO
SET FIRST POSITION INDEX
'M' TO 1
YES
INCREMENT THE VALUE
OF THE POSITION IN
VOLUME FREQUENCY CROSS
CLASSIFICATION TABLE
BY 10
MAKE SURE THAT THE
DIFFERENCE IN SURVEY
AND MODEL TRIPS IS
POSITIVE
IS DIFFERENCE
IN # TRIPS > ONE
TENTH THE NUMBER
OF CENTROIDS?
YES
70
WRITE DIFFERENCE IN
TRIPS, PROD. AND
ATTRACTION CENTROID,
SURVEY AND MODEL TRIPS
ON UNIT 9
R 3
NO
INCREMENT THE VALUE
OF 'AM(K)' BY 1
YES
WERE THERE
LESS MODEL TRIPS
THAN SURVEY TRIPS
FROM 'I' TO 'J'?
NO
INCREMENT THE VALUE OF
'AR(K)' BY 1
YES
WERE THERE
MORE OR THE SAME
MODEL TRIPS AS
SURVEY TRIPS FROM
'I' TO 'J'?
NO
R 4
CALCULATE INDEX FOR
USE IN EVALUATING TRIP
DIFFERENCES AND STORE
IN 'K'

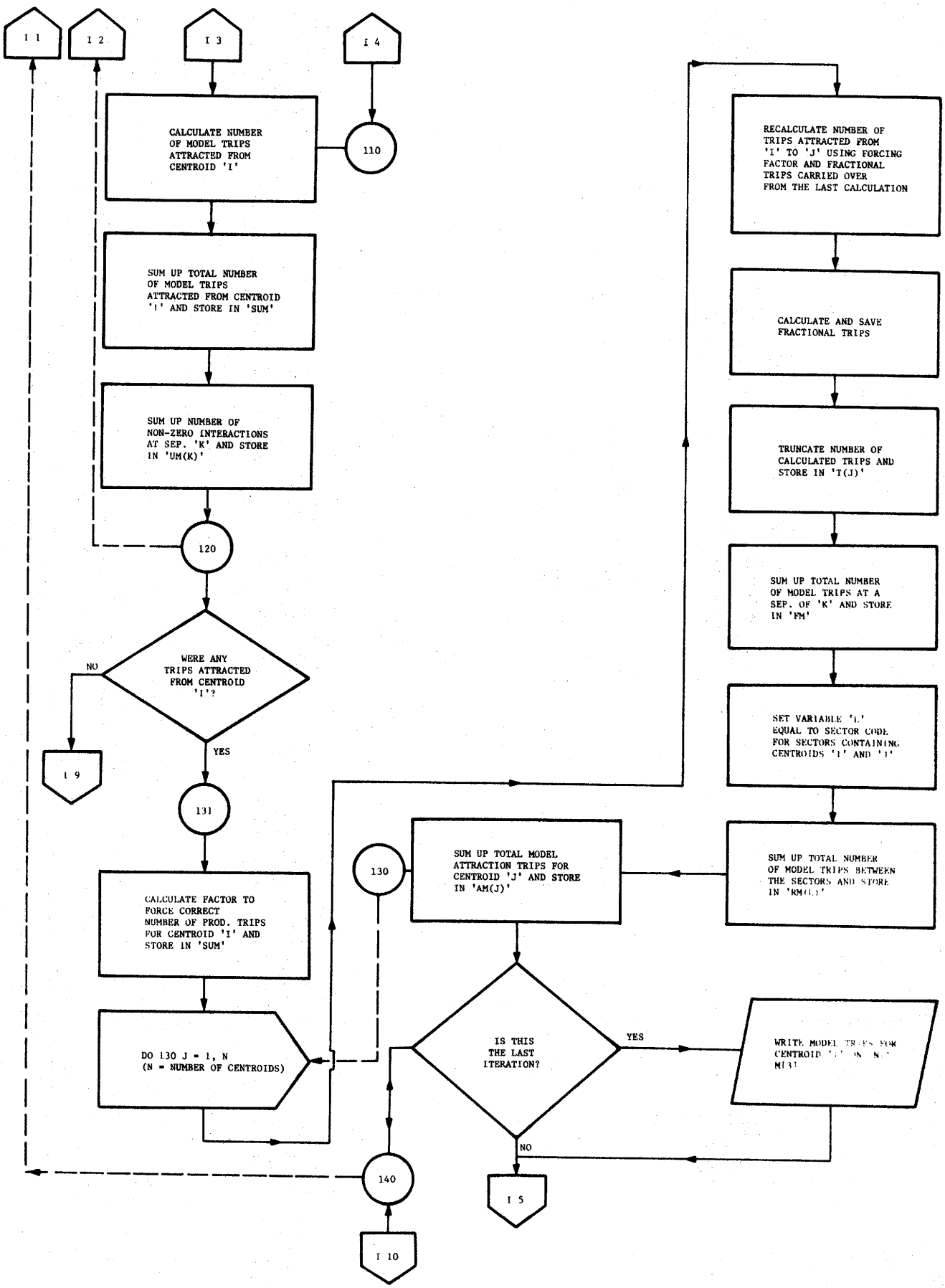

R 6
R 4
R 3
CALCULATE NUMBER OF RECORDS ON UNIT 9 AND STORE IN 'SIZE'
80
SUM UP THE SURVEY TRIP VOL. SQUARED AND STORE IN 'SXX'
SUM UP THE PRODUCT OF THE SURVEY AND MODEL TRIP VOL. AND STORE IN 'SXY'
SUM UP THE MODEL TRIP VOL. SQUARED AND STORE IN 'SYY'
IS NUMBER OF SURVEY TRIPS LARGER THAN ONE-TENTH NUMBER OF CENTROIDS?
YES
NO
CALCULATE INDEX AND STORE IN 'L'
CALCULATE NUMBER OF CENTROIDS WITH THAT NUMBER OF SURVEY TRIPS
90
IS NUMBER OF MODEL TRIPS LARGER THAN ONE-TENTH NUMBER OF CENTROIDS?
NO
YES
7
CALCULATE INDEX AND STORE IN 'M'
CALCULATE NUMBER OF CENTROIDS WITH THAT NUMBER OF MODEL TRIPS
100
STORE SEP. BETWEEN 'I' AND 'J' IN 'M'
CALCULATE NUMBER OF SURVEY TRIPS AT SEP OF 'M' AND STORE IN 'F(M)'
CALCULATE NUMBER OF MODEL TRIPS AT SEP OF 'M' AND STORE IN 'FM(M)'
CALCULATE NUMBER OF CENTROID PAIRS WITH SEP. 'M' AND STORE IN 'S(M)'
SUM UP NUMBER OF SURVEY TRIPS SQUARED AT SEP. 'M' AND STORE IN 'TXX(M)'
SUM UP PRODUCT OF SURVEY AND MODEL TRIPS AT SEP. 'M' AND STORE IN 'TXY(M)'
110
SUM UP NUMBER OF MOD TRIPS SQUARED AT SEP 'M' AND STORE IN 'TYY(M)'
R 8

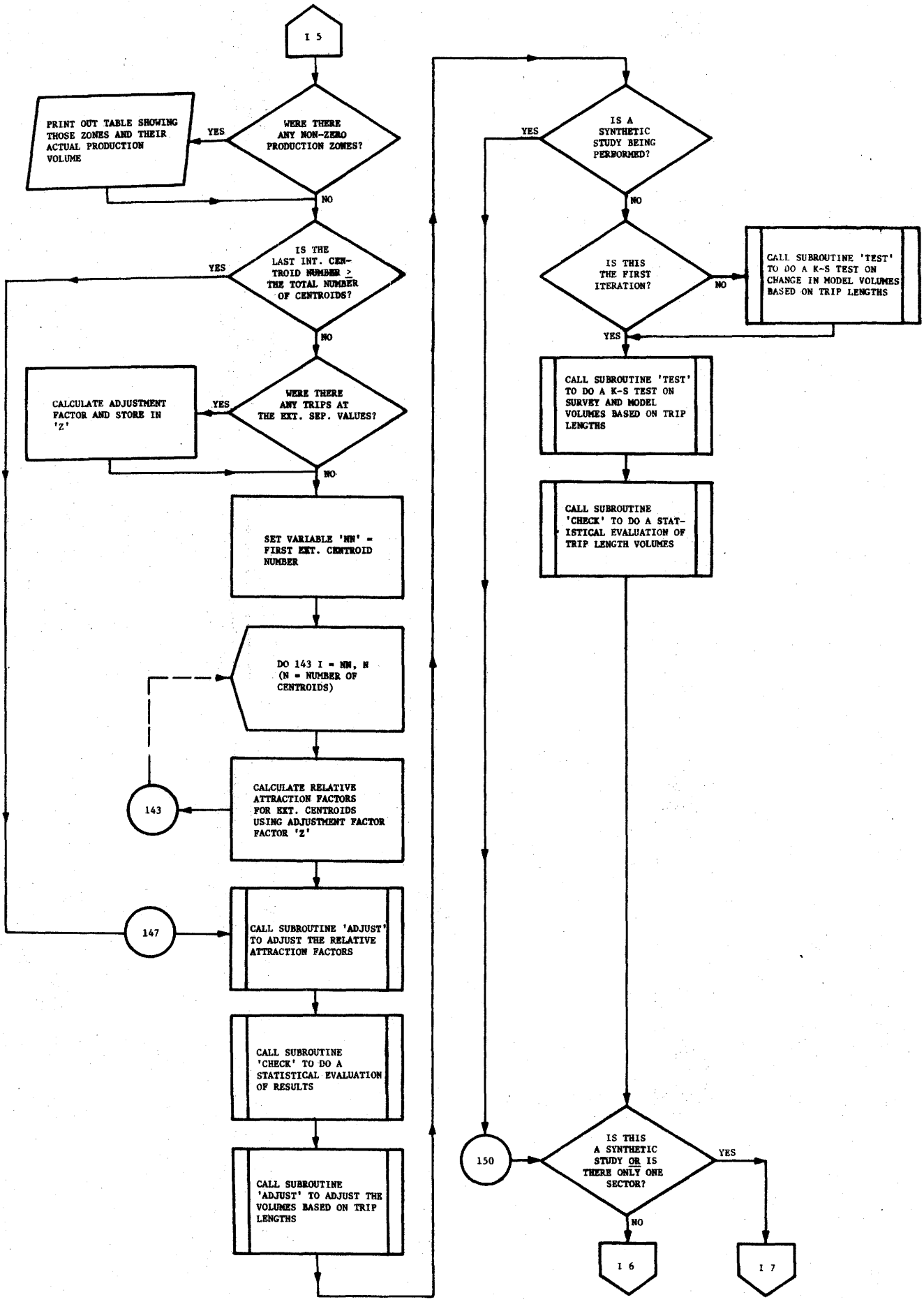

R 7
R 8
IS IT THE TOP OF A NEW PAGE?
NO
YES
PRINT OUT HEADING FOR ANALYSIS OF INDIVIDUAL ZONES
ARE THERE ANY PRODUCTIONS AND 'SXX' ≠ 'SYY'?
YES
NO
PRINT OUT PRODUCTION CENTROID NUMBER AND ITS PROD. VOLUMES
120
CALL SUBROUTINE 'REGRES' TO PERFORM AND PRINT OUT STATISTICAL EVALUATION OF CENTROID 'I'
125
END FILE UNITS 9 AND 10 REWIND UNITS B[2], Z[3], C[4], 9, AND 10
ZERO OUT VARIABLES TO BE USED.
DO 135 I = 1, NF (NF = MAXIMUM INT. SEP.)
IS IT THE TOP OF A NEW PAGE?
NO
YES
PRINT OUT HEADING FOR ANALYSIS OF ZONE PAIRS AT EACH SEP.
ARE THERE ANY TRIPS AT SEP. 'I' AND IS 'TXX(I) ≠ 'TYY(I)'?
YES
NO
PRINT OUT SEP. VALUE AND NUMBER OF SURVEY TRIPS AT THAT SEP. 'I'
130
CALL SUBROUTINE 'REGRET' TO PERFORM AND PRINT OUT STATISTICAL EVALUATION OF INTERACTIONS AT SEP. 'I'
SUM UP NUMBER OF SURVEY TRIPS AND STORE IN 'SX'
SUM UP NUMBER OF MODEL TRIPS AND STORE IN 'SY'
SUM UP NUMBER OF SURVEY TRIPS SQUARED AND STORE IN 'SXX'
SUM UP PRODUCT OF MODEL AND SURVEY TRIPS AND STORE IN 'SXY'
SUM UP NUMBER OF MODEL TRIPS SQUARED AND STORE IN 'SYY'
135
R 9

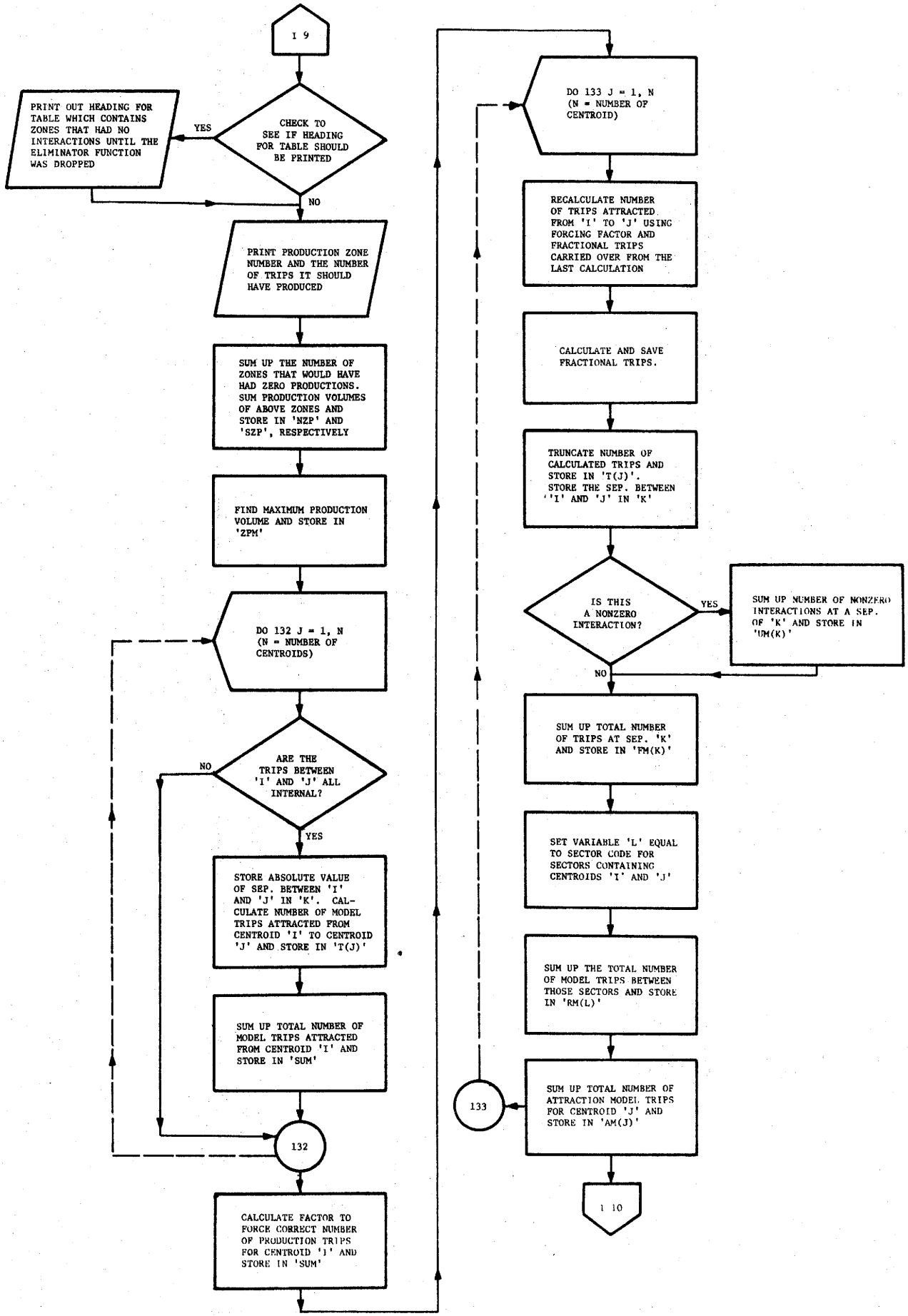

R 9
PRINT OUT HEADING FOR ENTIRE MATRIX COMPARISON
CALL SUBROUTINE 'REGRES' TO PERFORM AND PRINT STATISTICAL EVALUATION OF ENTIRE MATRIX
PRINT OUT VOLUME FREQUENCY CROSS CLASSIFICATION TABLE
PRINT OUT THE CENTROID PAIRS HAVING HIGH VOLUMES AND THEIR TRIP VOLUMES
ZERO OUT VARIABLES TO BE USED FOR FURTHER AN ANALYSIS
DO 180 I = 1, N (N = NUMBER OF CENTROIDS)
IS IT AT THE TOP OF A NEW PAGE?
YES
PRINT OUT HEADING FOR ANALYSIS OF SMALL VOLUME INTERCHANGES
NO
CALCULATE VOL. AND STORE IN 'SXY' AND SUM UP NUMBER OF SURVEY ZONE PAIRS HAVING SMALL VOL. AND STORE IN 'SX'
CALCULATE PERCENT OF ZONE PAIRS WITH VOLUME OF 'SXY' OR LESS CALCULATE TOTAL TRIPS AND STORE IN 'SXX'
SUM UP TOTAL TRIPS AND STORE IN 'S 1' CALCULATE WHAT PERCENTAGE OF ALL TRIPS THIS CUMULATIVE VOLUME IS AND STORE IN 'S 5'
SUM UP NUMBER OF MODEL ZONE PAIRS WITH VOL. OF 'SXY' OR LESS AND STORE IN 'SY'
CALCULATE WHAT PERCENTAGE OF ALL ZONE PAIRS 'SY' IS AND STORE IN 'S 4'
CALCULATE TOTAL TRIPS AND STORE IN 'SYY' SUM UP CUMULATIVE TRIP VOL. AND STORE IN 'S 2'
CALCULATE WHAT PERCENTAGE OF TOTAL TRIPS 'S 2' IS AND STORE IN 'S 6'
180
PRINT OUT ABOVE RESULTS IN ANALYSIS OF SMALL VOLUME INTERCHANGES TABLE
READ UNIT 9 TO SEE IF UNIT IS DUMMY REWIND UNIT 9
WERE THERE ANY RECORDS WRITTEN ON UNIT 9?
YES
CALL SUBROUTINE 'SORT' TO SORT RECORDS ON UNIT 9 AND PUT THEM ON UNIT 10
NO
CALCULATE PERCENTAGE OF SURVEY ZONE PAIRS WITH NO TRIPS AND STORE IN 'SO'. ZERO OUT VARIABLES TO BE USED.
R 10

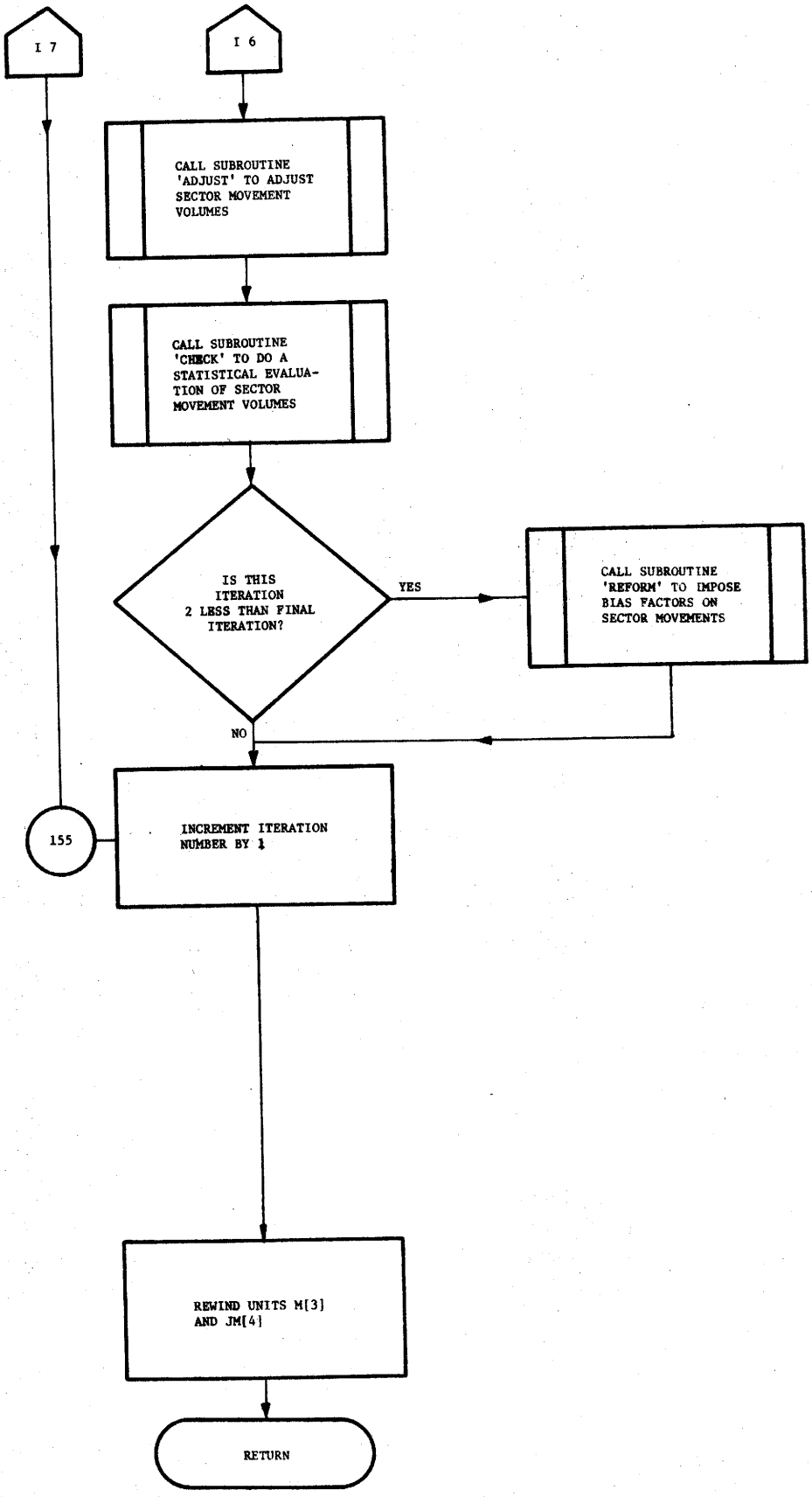

R 10
PRINT OUT NUMBER OF ZONE PAIRS WITH NO TRIPS AND THE PERCENTAGE THIS IS OF ALL THE ZONE PAIRS
184
CALCULATE AND PERFORM ANALYSIS OF TRIP VOLUME DIFFERENCES
PRINT OUT TABLE SHOWING ANALYSIS OF TRIP VOLUME DIFFERENCES
RETURN

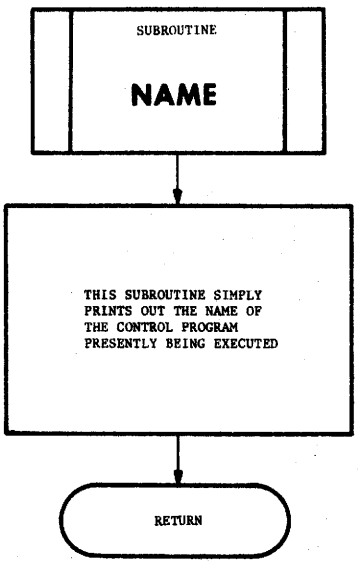

SUBROUTINE
MODEL
CALL SUBROUTINE 'NAME' TO PRINT THE NAME OF THE SUBROUTINE BEING EXECUTED 'MODEL'
REWIND UNITS JM[4] AND M[3]
READ HEADER RECORD OFF UNIT JM[4] WHICH CONTAINS THE SEPARATION MATRIX
WRITE HEADER RECORD ON UNIT M[3] WHICH WILL CONTAIN MODEL TRIP MATRIX
SET VARIABLE USED TO COUNT NUMBER OF ITERATIONS TO 1.0
ZERO OUT ARRAYS TO BE USED; SET VARIABLE 'RES' TO 0.0; AND REWIND UNIT JM[4]. SET VARIABLES 'NZP', 'SZP', 'ZPM' TO 0.0
READ HEADER RECORD OFF UNIT JM[4] WHICH CONTAINS THE SEPARATION MATRIX
I 1
I 2
DO 140 I = 1, N
(N = NUMBER OF CENTROIDS)
CLEAR ARRAY 'T'; READ SEP. FOR CENTROID 'I' OFF UNIT JM[4]
DOES CENTROID 'I' HAVE ANY PROD. TRIPS?
NO
YES
I 1
DO 120 J = 1, N
(N = NUMBER OF CENTROIDS)
ARE THE TRIPS BETWEEN 'I' AND 'J' ALL INTERNAL?
NO
YES
I 2
STORE ABSOLUTE VALUE OF SEP. BETWEEN 'I' AND 'J' IN 'K'
ARE NEG. SEP. BEING READ IN?
YES
NO
I 4
IS THE ELIMINATOR FUNCTION TO BE APPLIED?
NO
YES
ARE TRIPS BETWEEN 'I' AND 'J' TO BE ELIMINATED?
YES
I 3
NO

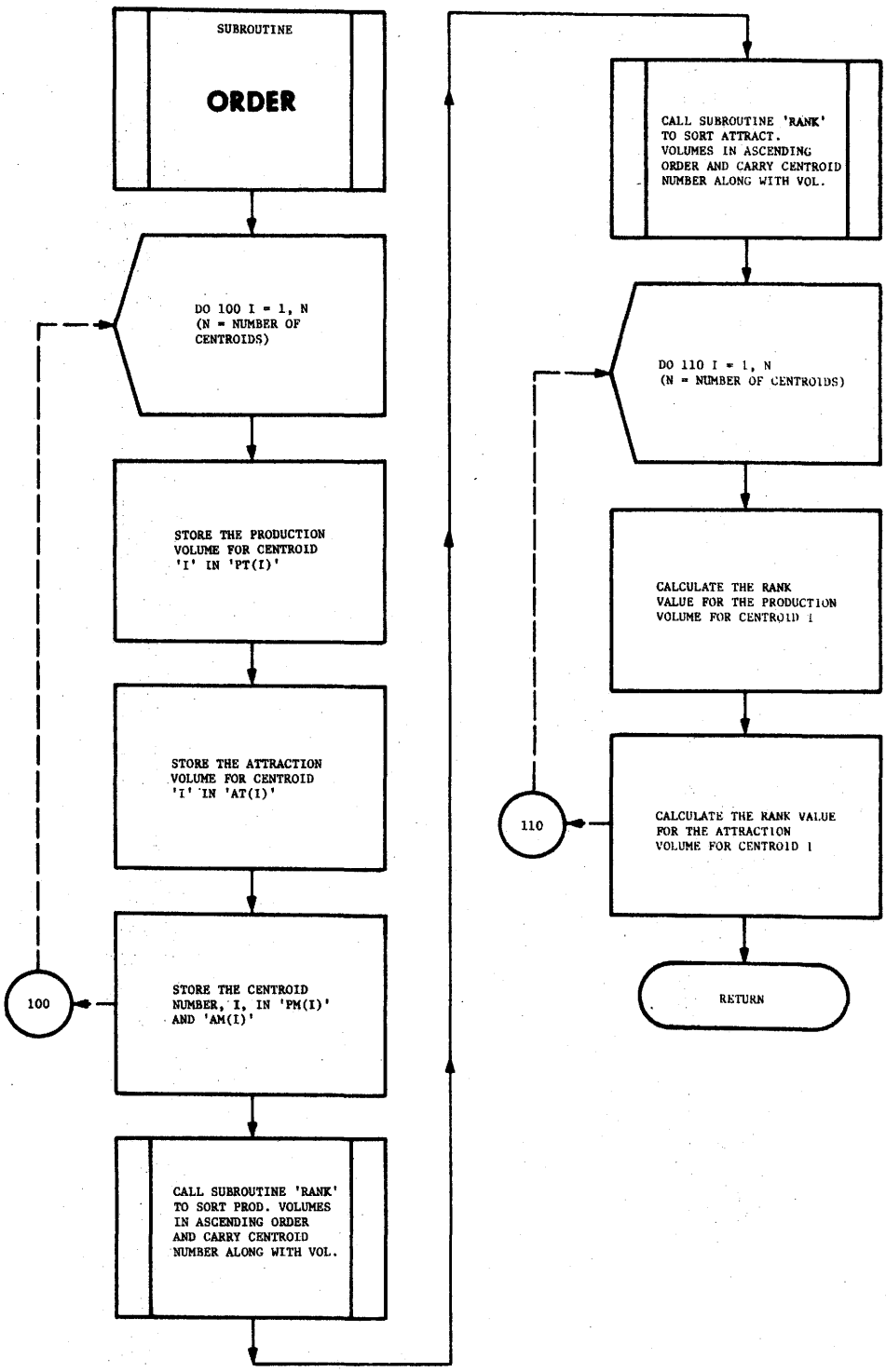

I 1
I 2
I 3
I 4
CALCULATE NUMBER OF MODEL TRIPS ATTRACTED FROM CENTROID 'I'
110
SUM UP TOTAL NUMBER OF MODEL TRIPS ATTRACTED FROM CENTROID 'I' AND STORE IN 'SUM'
SUM UP NUMBER OF NON-ZERO INTERACTIONS AT SEP. 'K' AND STORE IN 'UM(K)'
120
WERE ANY TRIPS ATTRACTED FROM CENTROID 'I'?
NO
YES
I 9
131
CALCULATE FACTOR TO FORCE CORRECT NUMBER OF PROD. TRIPS FOR CENTROID 'I' AND STORE IN 'SUM'
DO 130 J = 1, N
(N = NUMBER OF CENTROIDS)
130
SUM UP TOTAL MODEL ATTRACTION TRIPS FOR CENTROID 'J' AND STORE IN 'AM(J)'
IS THIS THE LAST ITERATION?
YES
NO
WRITE MODEL TRIPS FOR CENTROID
I 5
140
I 10
SUM UP TOTAL NUMBER OF MODEL TRIPS BETWEEN THE SECTORS AND STORE IN 'RM(L)'
SET VARIABLE 'L' EQUAL TO SECTOR CODE FOR SECTORS CONTAINING CENTROIDS 'I' AND 'J'
SUM UP TOTAL NUMBER OF MODEL TRIPS AT A SEP. OF 'K' AND STORE IN 'FM'
TRUNCATE NUMBER OF CALCULATED TRIPS AND STORE IN 'T(J)'
CALCULATE AND SAVE FRACTIONAL TRIPS
RECALCULATE NUMBER OF TRIPS ATTRACTED FROM 'I' TO 'J' USING FORCING FACTOR AND FRACTIONAL TRIPS CARRIED OVER FROM THE LAST CALCULATION

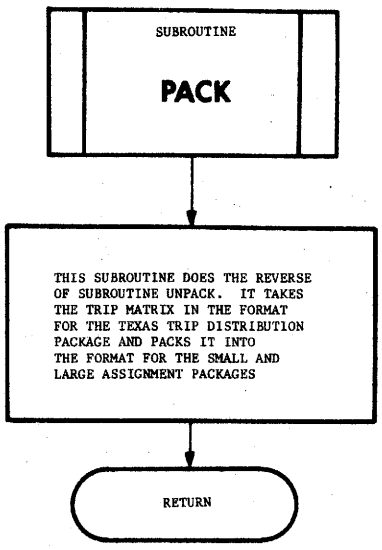

I 5
WERE THERE ANY NON-ZERO PRODUCTION ZONES?
YES
PRINT OUT TABLE SHOWING THOSE ZONES AND THEIR ACTUAL PRODUCTION VOLUME
NO
IS THE LAST INT. CENTROID NUMBER ≥ THE TOTAL NUMBER OF CENTROIDS?
YES
NO
WERE THERE ANY TRIPS AT THE EXT. SEP. VALUES?
YES
CALCULATE ADJUSTMENT FACTOR AND STORE IN 'Z'
NO
SET VARIABLE 'NN' = FIRST EXT. CENTROID NUMBER
DO 143 I = NN, N (N = NUMBER OF CENTROIDS)
143
CALCULATE RELATIVE ATTRACTION FACTORS FOR EXT. CENTROIDS USING ADJUSTMENT FACTOR FACTOR 'Z'
147
CALL SUBROUTINE 'ADJUST' TO ADJUST THE RELATIVE ATTRACTION FACTORS
CALL SUBROUTINE 'CHECK' TO DO A STATISTICAL EVALUATION OF RESULTS
CALL SUBROUTINE 'ADJUST' TO ADJUST THE VOLUMES BASED ON TRIP LENGTHS
IS A SYNTHETIC STUDY BEING PERFORMED?
YES
NO
IS THIS THE FIRST ITERATION?
NO
CALL SUBROUTINE 'TEST' TO DO A K-S TEST ON CHANGE IN MODEL VOLUMES BASED ON TRIP LENGTHS
YES
CALL SUBROUTINE 'TEST' TO DO A K-S TEST ON SURVEY AND MODEL VOLUMES BASED ON TRIP LENGTHS
CALL SUBROUTINE 'CHECK' TO DO A STATISTICAL EVALUATION OF TRIP LENGTH VOLUMES
150
IS THIS A SYNTHETIC STUDY OR IS THERE ONLY ONE SECTOR?
YES
NO
I 6
I 7

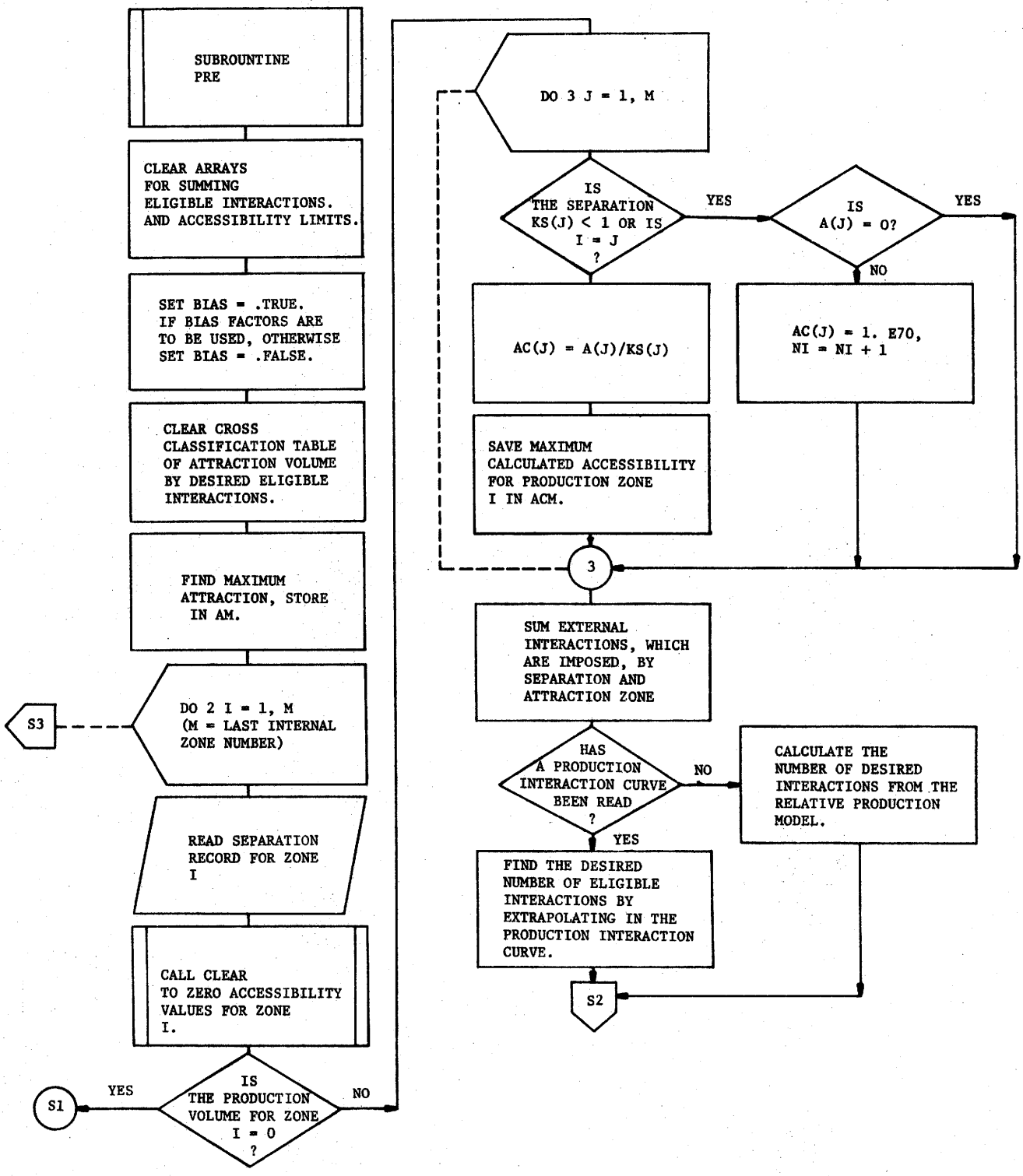

I 9
CHECK TO SEE IF HEADING FOR TABLE SHOULD BE PRINTED
YES
PRINT OUT HEADING FOR TABLE WHICH CONTAINS ZONES THAT HAD NO INTERACTIONS UNTIL THE ELIMINATOR FUNCTION WAS DROPPED
NO
PRINT PRODUCTION ZONE NUMBER AND THE NUMBER OF TRIPS IT SHOULD HAVE PRODUCED
SUM UP THE NUMBER OF ZONES THAT WOULD HAVE HAD ZERO PRODUCTIONS. SUM PRODUCTION VOLUMES OF ABOVE ZONES AND STORE IN 'NZP' AND 'SZP', RESPECTIVELY
FIND MAXIMUM PRODUCTION VOLUME AND STORE IN 'ZPM'
DO 132 J = 1, N (N = NUMBER OF CENTROIDS)
ARE THE TRIPS BETWEEN 'I' AND 'J' ALL INTERNAL?
NO
YES
STORE ABSOLUTE VALUE OF SEP. BETWEEN 'I' AND 'J' IN 'K'. CALCULATE NUMBER OF MODEL TRIPS ATTRACTED FROM CENTROID 'I' TO CENTROID 'J' AND STORE IN 'T(J)'
SUM UP TOTAL NUMBER OF MODEL TRIPS ATTRACTED FROM CENTROID 'I' AND STORE IN 'SUM'
132
CALCULATE FACTOR TO FORCE CORRECT NUMBER OF PRODUCTION TRIPS FOR CENTROID 'I' AND STORE IN 'SUM'
DO 133 J = 1, N (N = NUMBER OF CENTROID)
RECALCULATE NUMBER OF TRIPS ATTRACTED FROM 'I' TO 'J' USING FORCING FACTOR AND FRACTIONAL TRIPS CARRIED OVER FROM THE LAST CALCULATION
CALCULATE AND SAVE FRACTIONAL TRIPS.
TRUNCATE NUMBER OF CALCULATED TRIPS AND STORE IN 'T(J)'. STORE THE SEP. BETWEEN 'I' AND 'J' IN 'K'
IS THIS A NONZERO INTERACTION?
YES
SUM UP NUMBER OF NONZERO INTERACTIONS AT A SEP. OF 'K' AND STORE IN 'UM(K)'
NO
SUM UP TOTAL NUMBER OF TRIPS AT SEP. 'K' AND STORE IN 'FM(K)'
SET VARIABLE 'L' EQUAL TO SECTOR CODE FOR SECTORS CONTAINING CENTROIDS 'I' AND 'J'
SUM UP THE TOTAL NUMBER OF MODEL TRIPS BETWEEN THOSE SECTORS AND STORE IN 'RM(L)'
SUM UP TOTAL NUMBER OF ATTRACTION MODEL TRIPS FOR CENTROID 'J' AND STORE IN 'AM(J)'
133
I 10

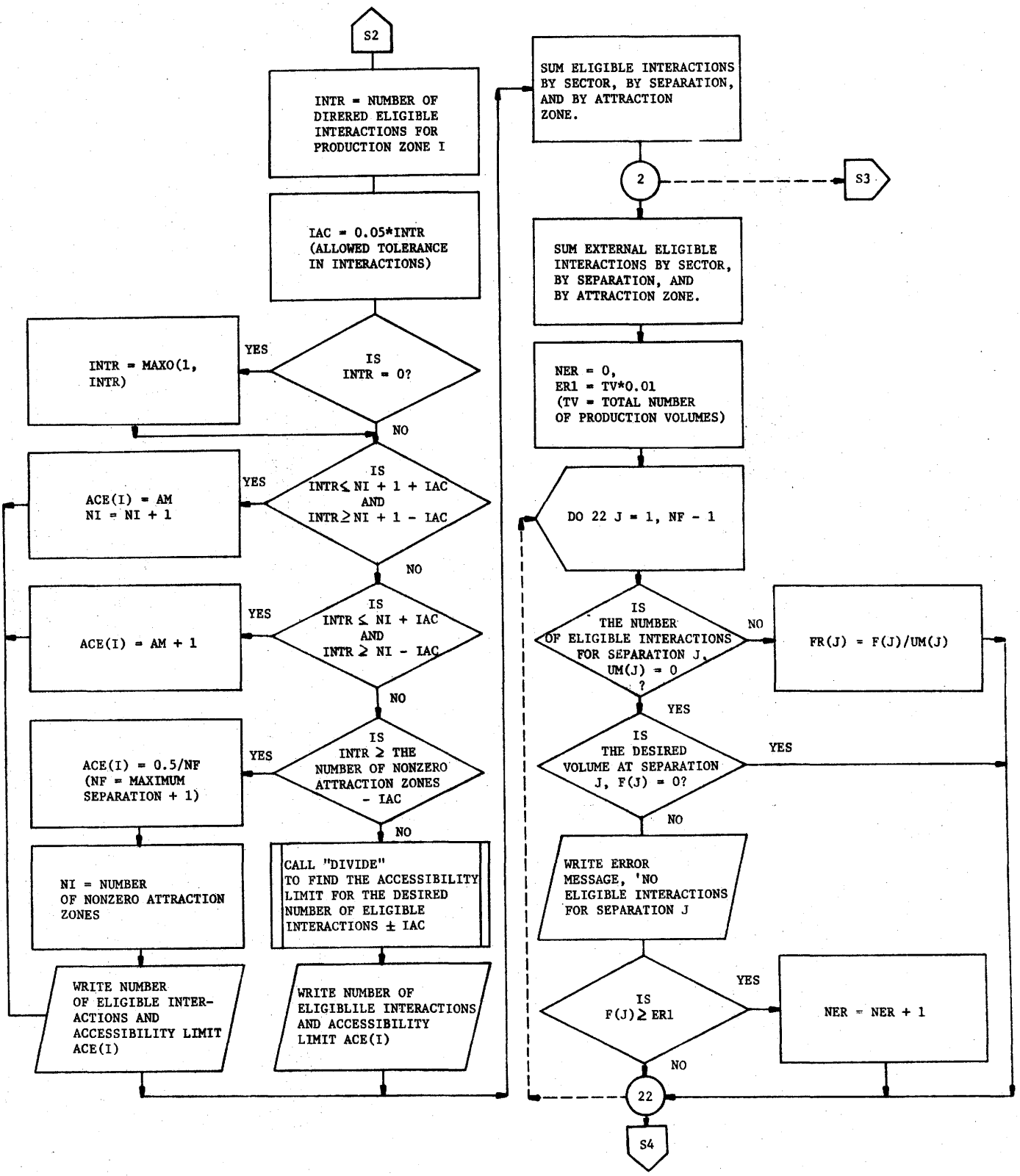

I 7
I 6
CALL SUBROUTINE 'ADJUST' TO ADJUST SECTOR MOVEMENT VOLUMES
CALL SUBROUTINE 'CHECK' TO DO A STATISTICAL EVALUATION OF SECTOR MOVEMENT VOLUMES
IS THIS ITERATION 2 LESS THAN FINAL ITERATION?
YES
CALL SUBROUTINE 'REFORM' TO IMPOSE BIAS FACTORS ON SECTOR MOVEMENTS
NO
155
INCREMENT ITERATION NUMBER BY 1
REWIND UNITS M[3] AND JM[4]
RETURN

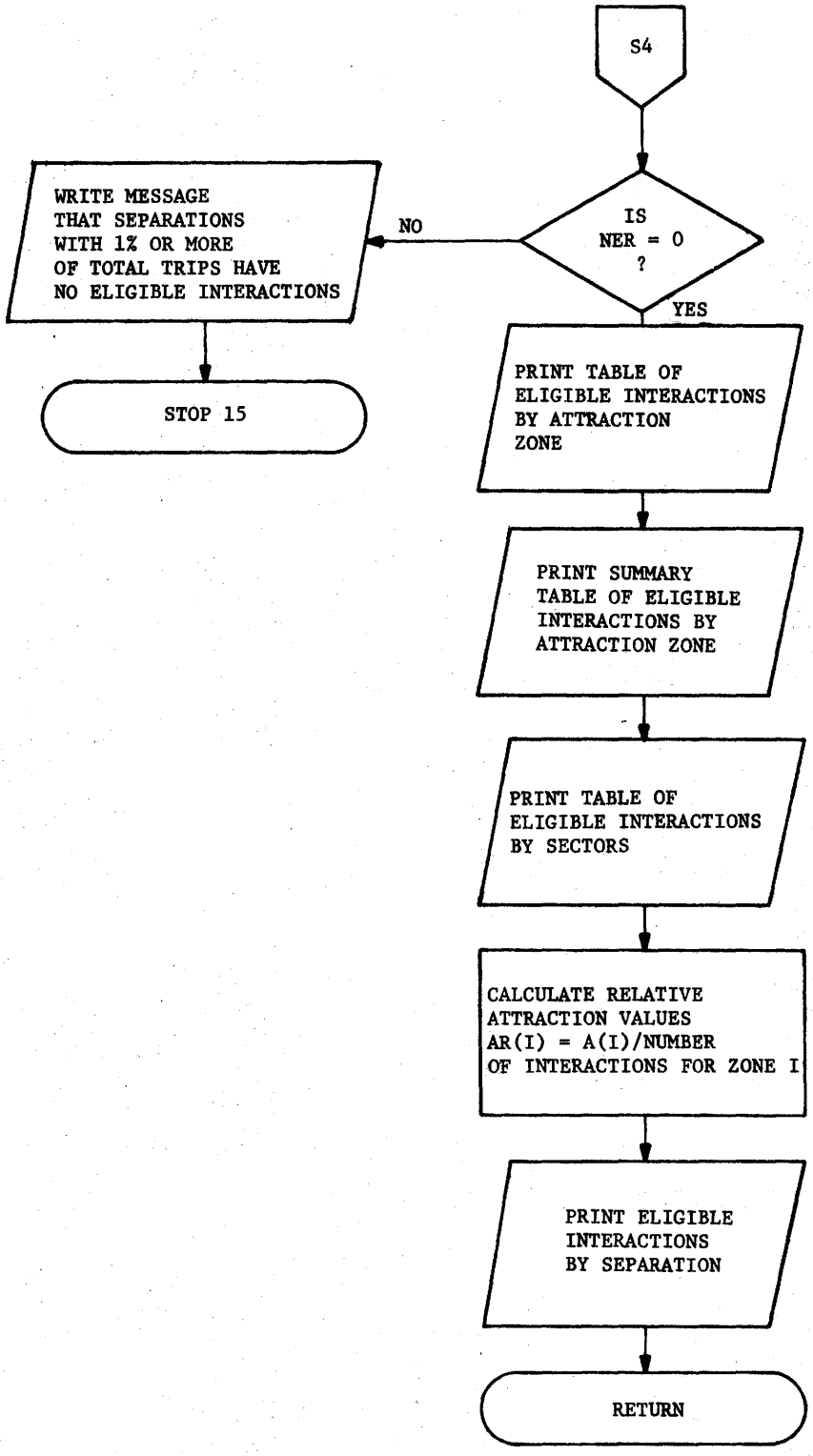

SUBROUTINE
NAME
THIS SUBROUTINE SIMPLY
PRINTS OUT THE NAME OF
THE CONTROL PROGRAM
PRESENTLY BEING EXECUTED
RETURN

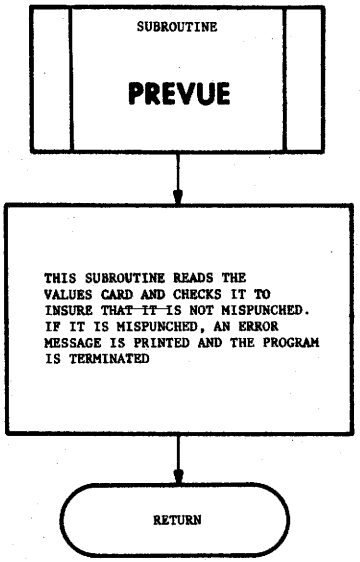

SUBROUTINE
ORDER
DO 100 I = 1, N
(N = NUMBER OF
CENTROIDS)
STORE THE PRODUCTION
VOLUME FOR CENTROID
'I' IN 'PT(I)'
STORE THE ATTRACTION
VOLUME FOR CENTROID
'I' IN 'AT(I)'
STORE THE CENTROID
NUMBER, I, IN 'PM(I)'
AND 'AM(I)'
100
CALL SUBROUTINE 'RANK'
TO SORT PROD. VOLUMES
IN ASCENDING ORDER
AND CARRY CENTROID
NUMBER ALONG WITH VOL.
CALL SUBROUTINE 'RANK'
TO SORT ATTRACT.
VOLUMES IN ASCENDING
ORDER AND CARRY CENTROID
NUMBER ALONG WITH VOL.
DO 110 I = 1, N
(N = NUMBER OF CENTROIDS)
CALCULATE THE RANK
VALUE FOR THE PRODUCTION
VOLUME FOR CENTROID I
CALCULATE THE RANK VALUE
FOR THE ATTRACTION
VOLUME FOR CENTROID I
110
RETURN

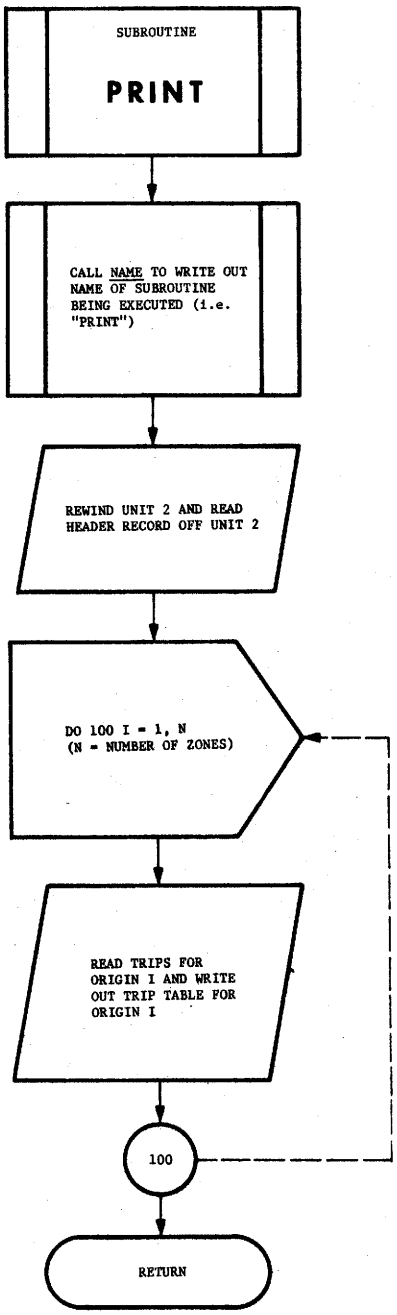

SUBROUTINE
PACK
THIS SUBROUTINE DOES THE REVERSE OF SUBROUTINE UNPACK. IT TAKES THE TRIP MATRIX IN THE FORMAT FOR THE TEXAS TRIP DISTRIBUTION PACKAGE AND PACKS IT INTO THE FORMAT FOR THE SMALL AND LARGE ASSIGNMENT PACKAGES
RETURN

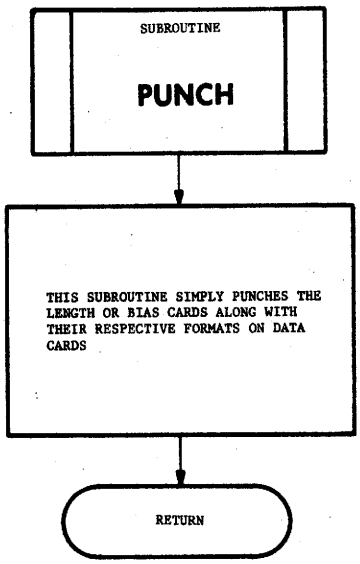

SUBROUNTINE
PRE
CLEAR ARRAYS
FOR SUMMING
ELIGIBLE INTERACTIONS.
AND ACCESSIBILITY LIMITS.
SET BIAS = .TRUE.
IF BIAS FACTORS ARE
TO BE USED, OTHERWISE
SET BIAS = .FALSE.
CLEAR CROSS
CLASSIFICATION TABLE
OF ATTRACTION VOLUME
BY DESIRED ELIGIBLE
INTERACTIONS.
FIND MAXIMUM
ATTRACTION, STORE
IN AM.
S3
DO 2 I = 1, M
(M = LAST INTERNAL
ZONE NUMBER)
READ SEPARATION
RECORD FOR ZONE
I
CALL CLEAR
TO ZERO ACCESSIBILITY
VALUES FOR ZONE
I.
S1
YES
IS
THE PRODUCTION
VOLUME FOR ZONE
I = 0
?
NO
DO 3 J = 1, M
IS
THE SEPARATION
KS(J) < 1 OR IS
I = J
?
YES
IS
A(J) = 0?
YES
NO
AC(J) = A(J)/KS(J)
AC(J) = 1. E70,
NI = NI + 1
SAVE MAXIMUM
CALCULATED ACCESSIBILITY
FOR PRODUCTION ZONE
I IN ACM.
3
SUM EXTERNAL
INTERACTIONS, WHICH
ARE IMPOSED, BY
SEPARATION AND
ATTRACTION ZONE
HAS
A PRODUCTION
INTERACTION CURVE
BEEN READ
?
NO
YES
CALCULATE THE
NUMBER OF DESIRED
INTERACTIONS FROM THE
RELATIVE PRODUCTION
MODEL.
FIND THE DESIRED
NUMBER OF ELIGIBLE
INTERACTIONS BY
EXTRAPOLATING IN THE
PRODUCTION INTERACTION
CURVE.
S2

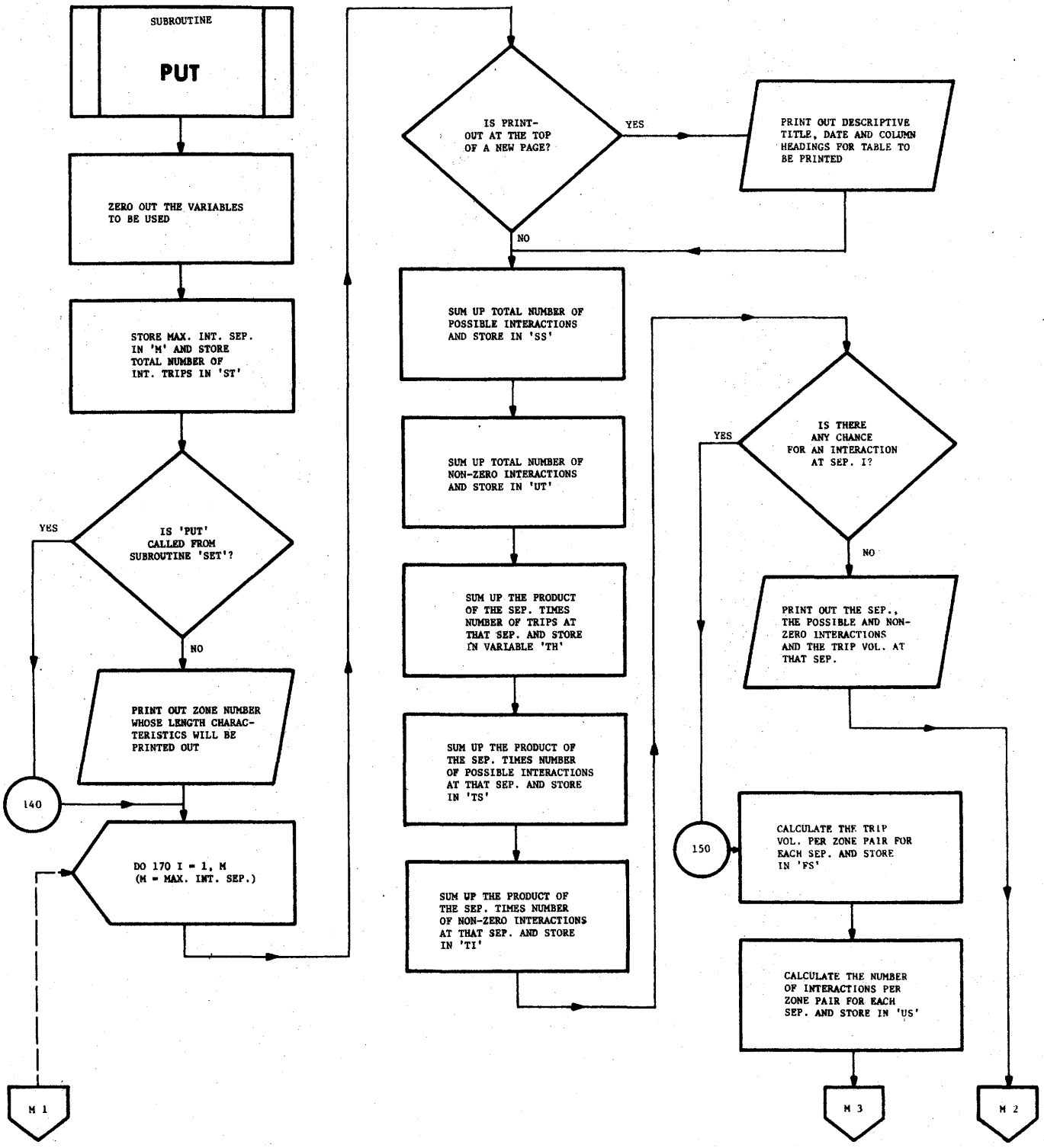

S2
INTR = NUMBER OF DIRERED ELIGIBLE INTERACTIONS FOR PRODUCTION ZONE I
IAC = 0.05*INTR (ALLOWED TOLERANCE IN INTERACTIONS)
IS INTR = 0?
YES
INTR = MAXO(1, INTR)
NO
IS INTR≤ NI + 1 + IAC AND INTR≥ NI + 1 - IAC
YES
ACE(I) = AM
NI = NI + 1
NO
IS INTR ≤ NI + IAC AND INTR ≥ NI - IAC
YES
ACE(I) = AM + 1
NO
IS INTR ≥ THE NUMBER OF NONZERO ATTRACTION ZONES - IAC
YES
ACE(I) = 0.5/NF (NF = MAXIMUM SEPARATION + 1)
NO
NI = NUMBER OF NONZERO ATTRACTION ZONES
CALL "DIVIDE" TO FIND THE ACCESSIBILITY LIMIT FOR THE DESIRED NUMBER OF ELIGIBLE INTERACTIONS ± IAC
WRITE NUMBER OF ELIGIBLE INTER-ACTIONS AND ACCESSIBILITY LIMIT ACE(I)
WRITE NUMBER OF ELIGIBLILE INTERACTIONS AND ACCESSIBILITY LIMIT ACE(I)
SUM ELIGIBLE INTERACTIONS BY SECTOR, BY SEPARATION, AND BY ATTRACTION ZONE.
2
S3
SUM EXTERNAL ELIGIBLE INTERACTIONS BY SECTOR, BY SEPARATION, AND BY ATTRACTION ZONE.
NER = 0,
ER1 = TV*0.01
(TV = TOTAL NUMBER OF PRODUCTION VOLUMES)
DO 22 J = 1, NF - 1
IS THE NUMBER OF ELIGIBLE INTERACTIONS FOR SEPARATION J, UM(J) = 0 ?
NO
FR(J) = F(J)/UM(J)
YES
IS THE DESIRED VOLUME AT SEPARATION J, F(J) = 0?
YES
NO
WRITE ERROR MESSAGE, 'NO ELIGIBLE INTERACTIONS FOR SEPARATION J
IS F(J)≥ ER1
YES
NER = NER + 1
NO
22
S4

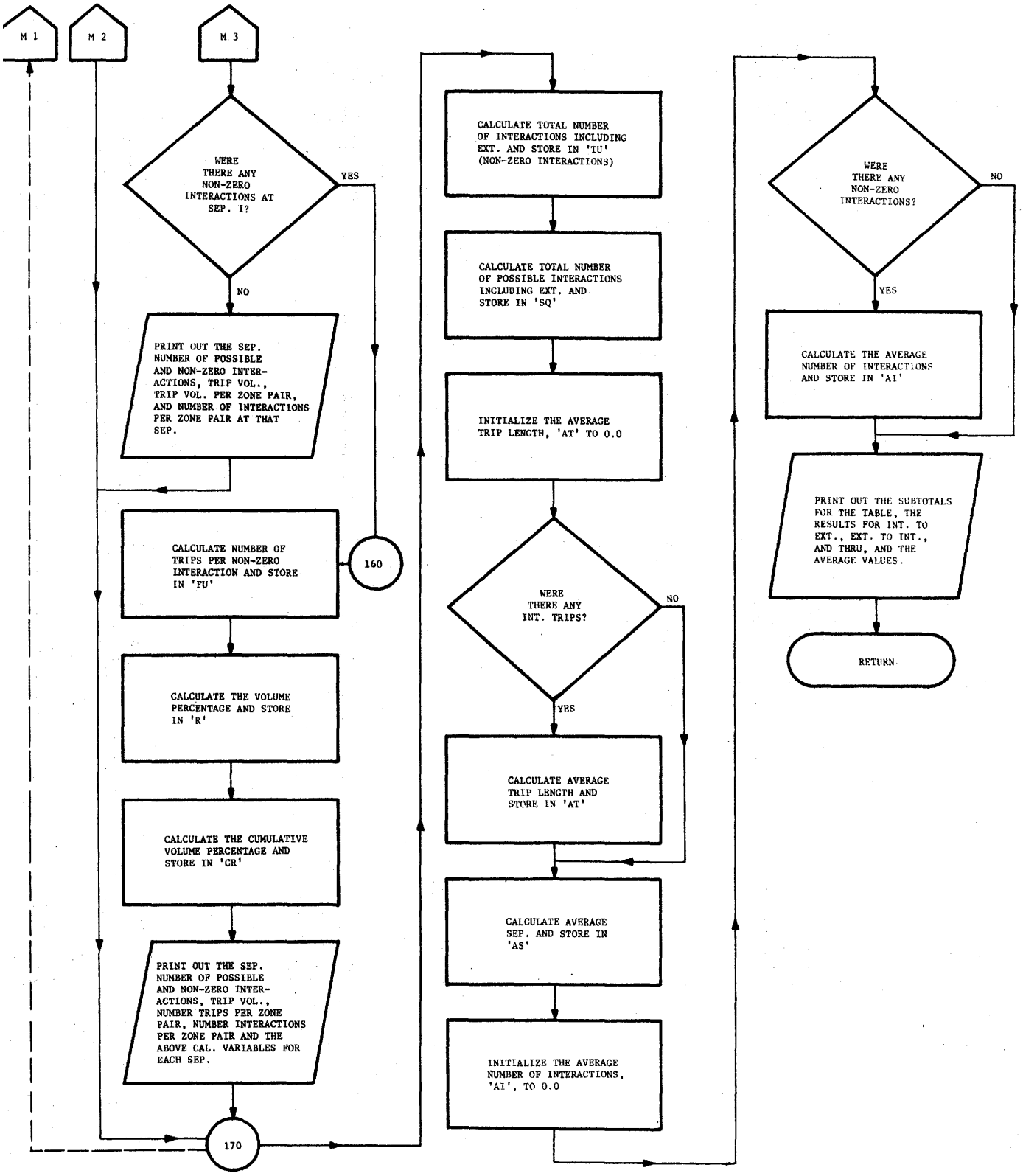

S4
IS
NER = 0
?
NO
YES
WRITE MESSAGE
THAT SEPARATIONS
WITH 1% OR MORE
OF TOTAL TRIPS HAVE
NO ELIGIBLE INTERACTIONS
STOP 15
PRINT TABLE OF
ELIGIBLE INTERACTIONS
BY ATTRACTION
ZONE
PRINT SUMMARY
TABLE OF ELIGIBLE
INTERACTIONS BY
ATTRACTION ZONE
PRINT TABLE OF
ELIGIBLE INTERACTIONS
BY SECTORS
CALCULATE RELATIVE
ATTRACTION VALUES
AR(I) = A(I)/NUMBER
OF INTERACTIONS FOR ZONE I
PRINT ELIGIBLE
INTERACTIONS
BY SEPARATION
RETURN

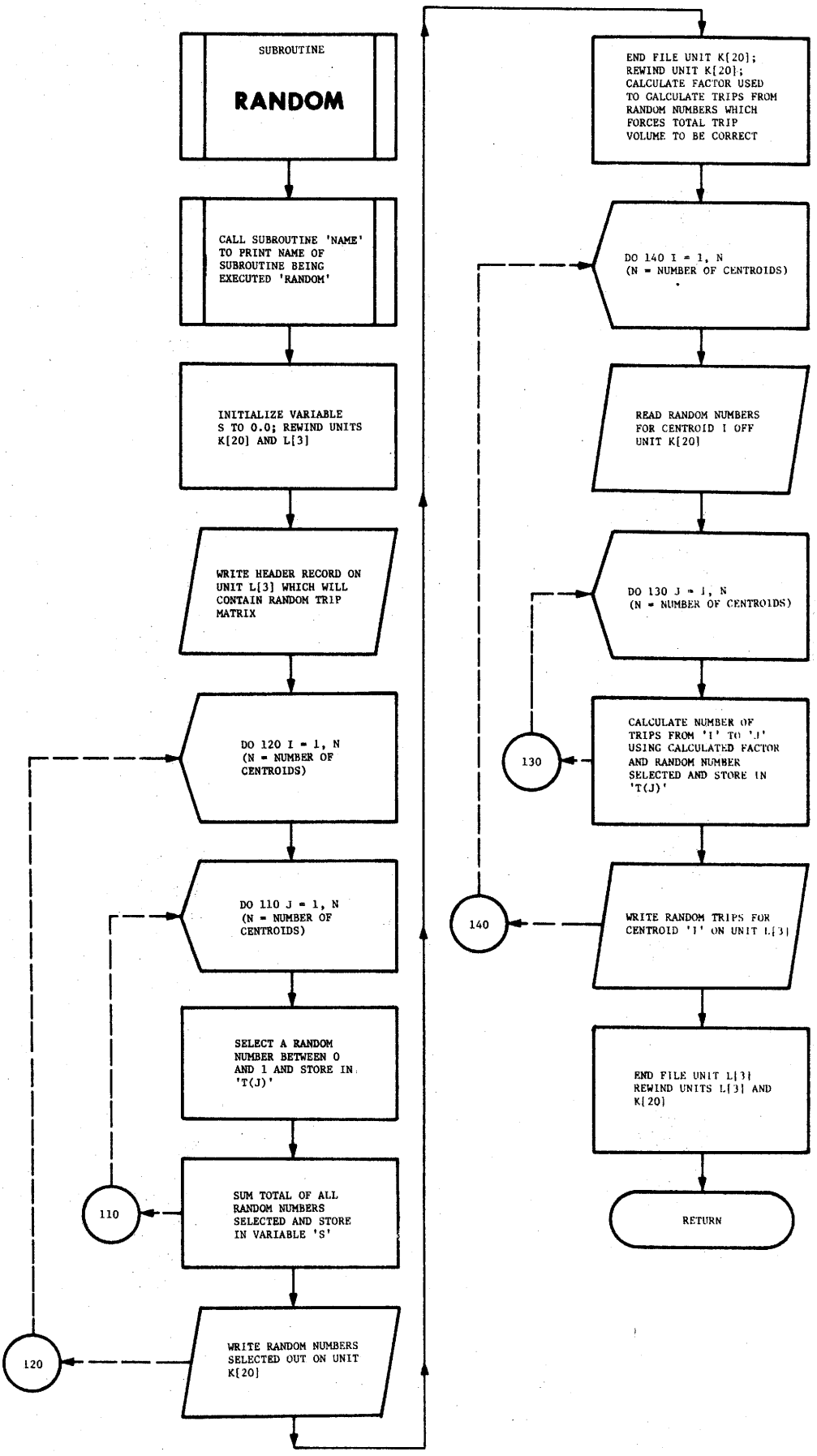

SUBROUTINE
PREVUE
THIS SUBROUTINE READS THE VALUES CARD AND CHECKS IT TO INSURE THAT IT IS NOT MISPUNCHED. IF IT IS MISPUNCHED, AN ERROR MESSAGE IS PRINTED AND THE PROGRAM IS TERMINATED
RETURN

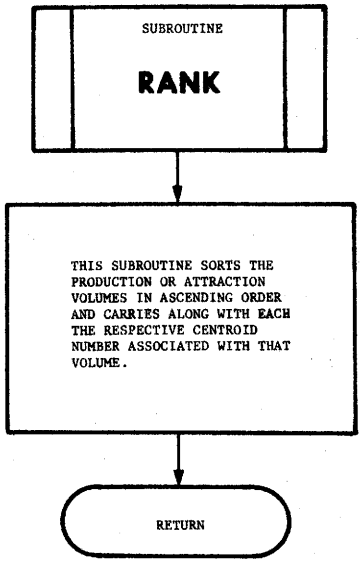

SUBROUTINE
PRINT
CALL NAME TO WRITE OUT NAME OF SUBROUTINE BEING EXECUTED (i.e. "PRINT")
REWIND UNIT 2 AND READ HEADER RECORD OFF UNIT 2
DO 100 I = 1, N
(N = NUMBER OF ZONES)
READ TRIPS FOR ORIGIN I AND WRITE OUT TRIP TABLE FOR ORIGIN I
100
RETURN

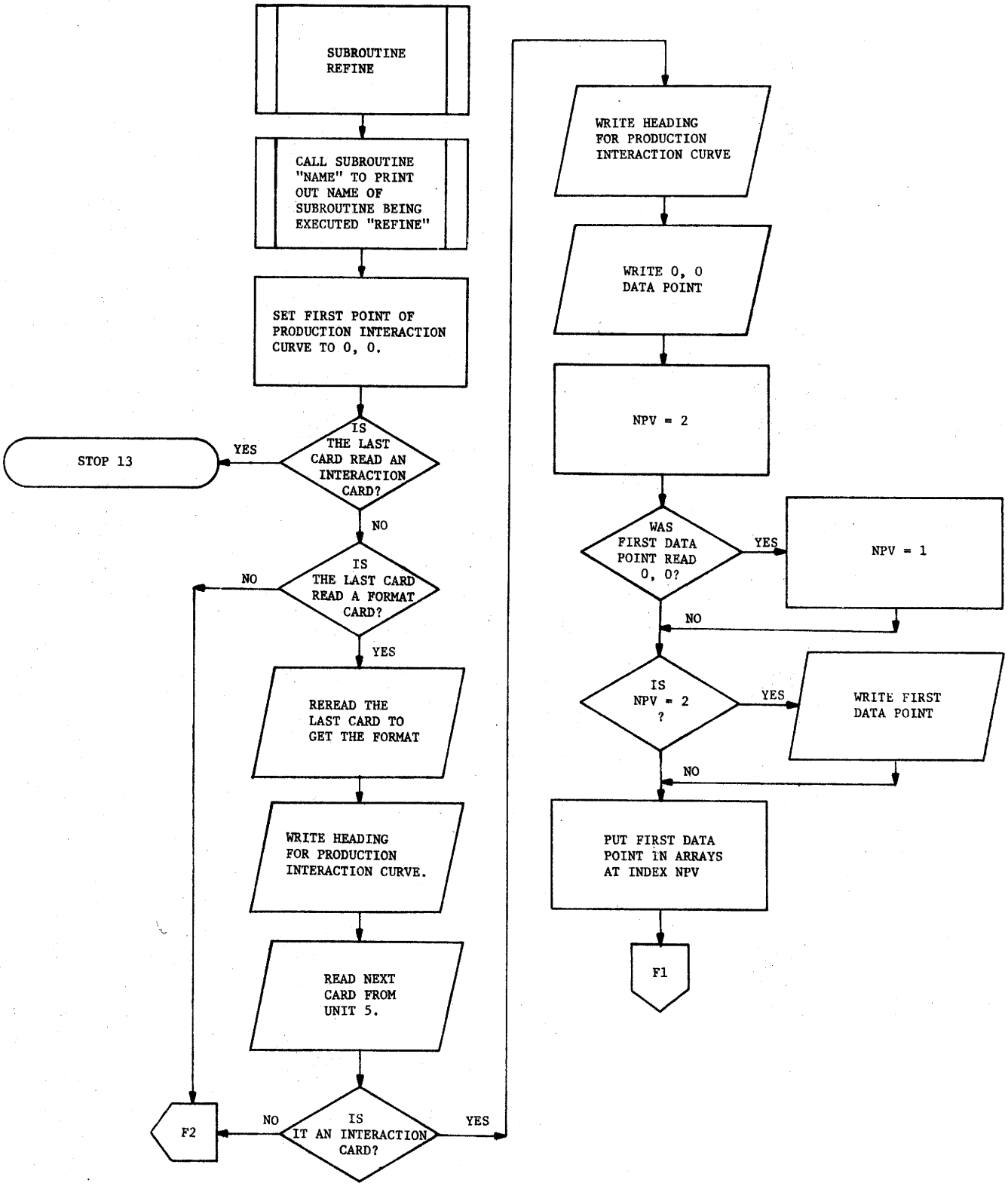

SUBROUTINE
PUNCH
THIS SUBROUTINE SIMPLY PUNCHES THE LENGTH OR BIAS CARDS ALONG WITH THEIR RESPECTIVE FORMATS ON DATA CARDS
RETURN

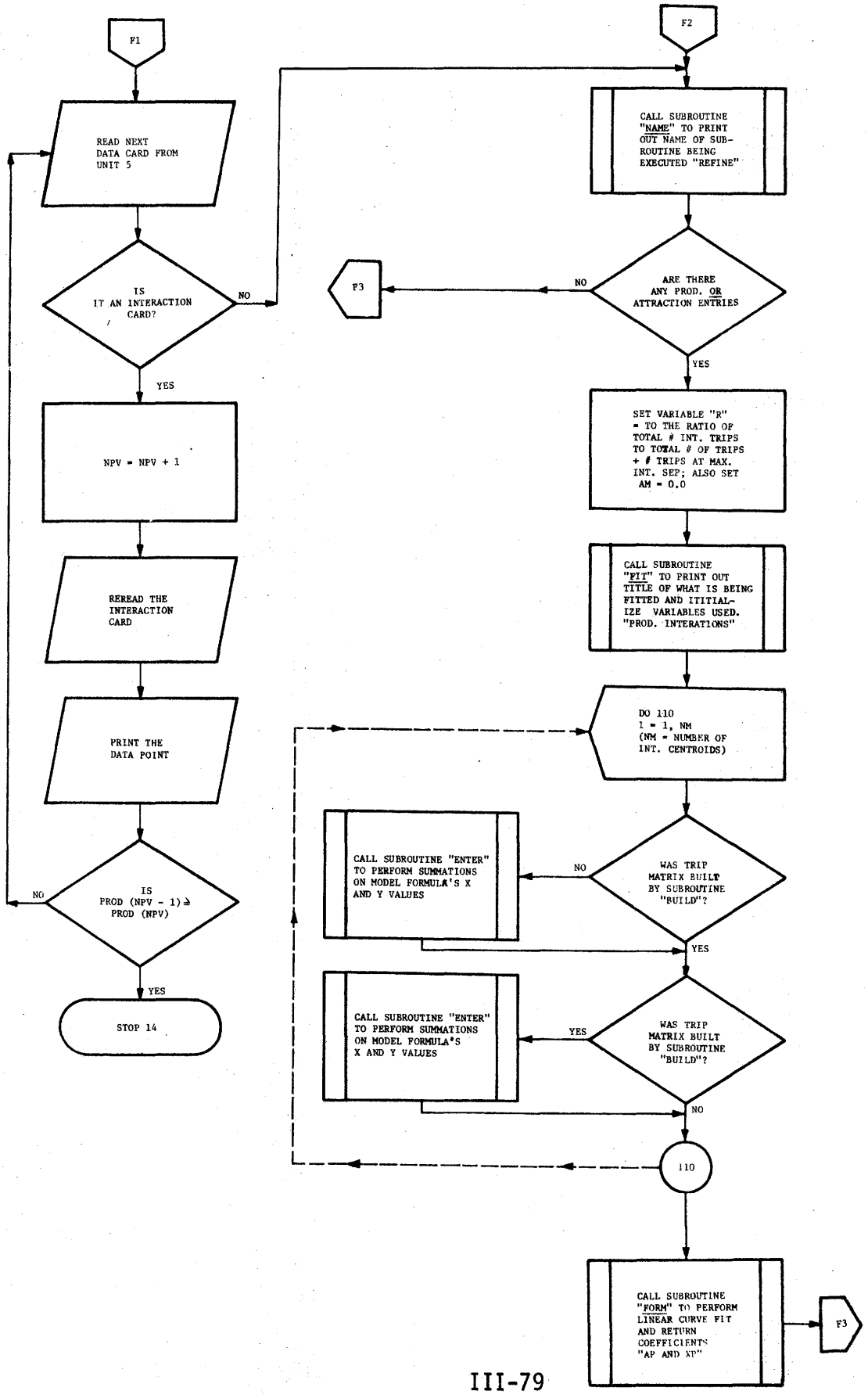

SUBROUTINE
PUT
ZERO OUT THE VARIABLES TO BE USED
STORE MAX. INT. SEP. IN 'M' AND STORE TOTAL NUMBER OF INT. TRIPS IN 'ST'
IS 'PUT' CALLED FROM SUBROUTINE 'SET'?
YES
NO
PRINT OUT ZONE NUMBER WHOSE LENGTH CHARACTERISTICS WILL BE PRINTED OUT
140
DO 170 I = 1, M (M = MAX. INT. SEP.)
M 1
IS PRINT-OUT AT THE TOP OF A NEW PAGE?
YES
PRINT OUT DESCRIPTIVE TITLE, DATE AND COLUMN HEADINGS FOR TABLE TO BE PRINTED
NO
SUM UP TOTAL NUMBER OF POSSIBLE INTERACTIONS AND STORE IN 'SS'
SUM UP TOTAL NUMBER OF NON-ZERO INTERACTIONS AND STORE IN 'UT'
SUM UP THE PRODUCT OF THE SEP. TIMES NUMBER OF TRIPS AT THAT SEP. AND STORE IN VARIABLE 'TH'
SUM UP THE PRODUCT OF THE SEP. TIMES NUMBER OF POSSIBLE INTERACTIONS AT THAT SEP. AND STORE IN 'TS'
SUM UP THE PRODUCT OF THE SEP. TIMES NUMBER OF NON-ZERO INTERACTIONS AT THAT SEP. AND STORE IN 'TI'
IS THERE ANY CHANCE FOR AN INTERACTION AT SEP. I?
YES
NO
PRINT OUT THE SEP., THE POSSIBLE AND NON-ZERO INTERACTIONS AND THE TRIP VOL. AT THAT SEP.
150
CALCULATE THE TRIP VOL. PER ZONE PAIR FOR EACH SEP. AND STORE IN 'FS'
CALCULATE THE NUMBER OF INTERACTIONS PER ZONE PAIR FOR EACH SEP. AND STORE IN 'US'
M 3
M 2

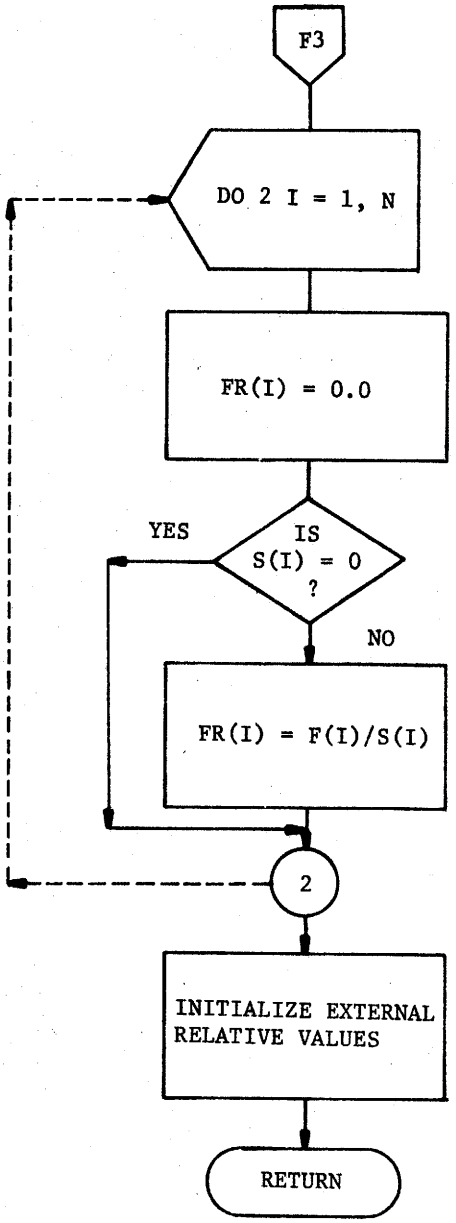

M 1
M 2
M 3
WERE THERE ANY NON-ZERO INTERACTIONS AT SEP. I?
YES
NO
PRINT OUT THE SEP. NUMBER OF POSSIBLE AND NON-ZERO INTER-ACTIONS, TRIP VOL., TRIP VOL. PER ZONE PAIR, AND NUMBER OF INTERACTIONS PER ZONE PAIR AT THAT SEP.
CALCULATE NUMBER OF TRIPS PER NON-ZERO INTERACTION AND STORE IN 'FU'
160
CALCULATE THE VOLUME PERCENTAGE AND STORE IN 'R'
CALCULATE THE CUMULATIVE VOLUME PERCENTAGE AND STORE IN 'CR'
PRINT OUT THE SEP. NUMBER OF POSSIBLE AND NON-ZERO INTER-ACTIONS, TRIP VOL., NUMBER TRIPS PER ZONE PAIR, NUMBER INTERACTIONS PER ZONE PAIR AND THE ABOVE CAL. VARIABLES FOR EACH SEP.
170
CALCULATE TOTAL NUMBER OF INTERACTIONS INCLUDING EXT. AND STORE IN 'TU' (NON-ZERO INTERACTIONS)
CALCULATE TOTAL NUMBER OF POSSIBLE INTERACTIONS INCLUDING EXT. AND STORE IN 'SQ'
INITIALIZE THE AVERAGE TRIP LENGTH, 'AT' TO 0.0
WERE THERE ANY INT. TRIPS?
NO
YES
CALCULATE AVERAGE TRIP LENGTH AND STORE IN 'AT'
CALCULATE AVERAGE SEP. AND STORE IN 'AS'
INITIALIZE THE AVERAGE NUMBER OF INTERACTIONS, 'AI', TO 0.0
WERE THERE ANY NON-ZERO INTERACTIONS?
NO
YES
CALCULATE THE AVERAGE NUMBER OF INTERACTIONS AND STORE IN 'AI'
PRINT OUT THE SUBTOTALS FOR THE TABLE, THE RESULTS FOR INT. TO EXT., EXT. TO INT., AND THRU, AND THE AVERAGE VALUES.
RETURN

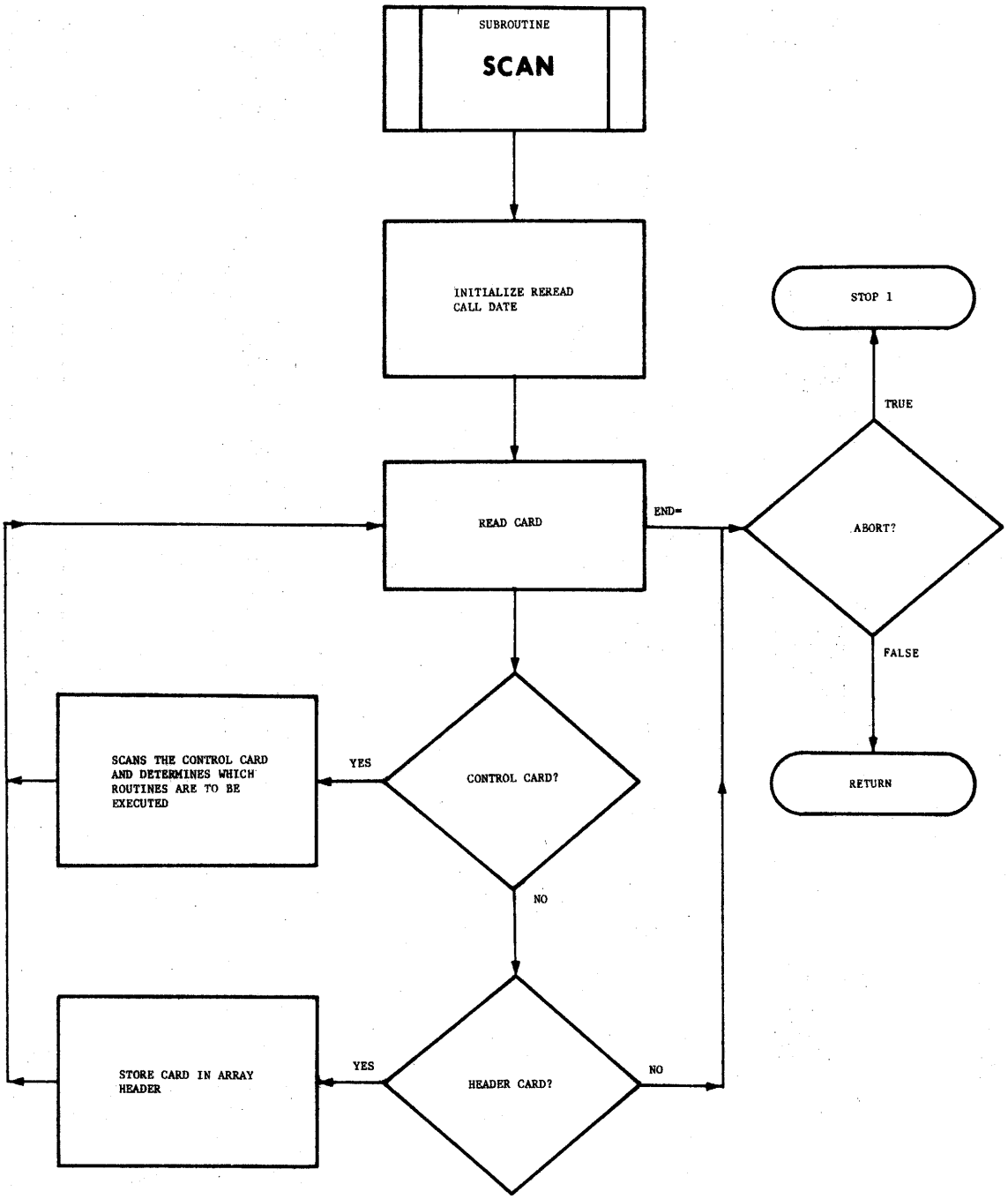

SUBROUTINE
RANDOM
CALL SUBROUTINE 'NAME' TO PRINT NAME OF SUBROUTINE BEING EXECUTED 'RANDOM'
INITIALIZE VARIABLE S TO 0.0; REWIND UNITS K[20] AND L[3]
WRITE HEADER RECORD ON UNIT L[3] WHICH WILL CONTAIN RANDOM TRIP MATRIX
DO 120 I = 1, N (N = NUMBER OF CENTROIDS)
DO 110 J = 1, N (N = NUMBER OF CENTROIDS)
SELECT A RANDOM NUMBER BETWEEN 0 AND 1 AND STORE IN 'T(J)'
SUM TOTAL OF ALL RANDOM NUMBERS SELECTED AND STORE IN VARIABLE 'S'
110
WRITE RANDOM NUMBERS SELECTED OUT ON UNIT K[20]
120
END FILE UNIT K[20]; REWIND UNIT K[20]; CALCULATE FACTOR USED TO CALCULATE TRIPS FROM RANDOM NUMBERS WHICH FORCES TOTAL TRIP VOLUME TO BE CORRECT
DO 140 I = 1, N (N = NUMBER OF CENTROIDS)
READ RANDOM NUMBERS FOR CENTROID I OFF UNIT K[20]
DO 130 J = 1, N (N = NUMBER OF CENTROIDS)
CALCULATE NUMBER OF TRIPS FROM 'I' TO 'J' USING CALCULATED FACTOR AND RANDOM NUMBER SELECTED AND STORE IN 'T(J)'
130
WRITE RANDOM TRIPS FOR CENTROID 'I' ON UNIT L[3]
140
END FILE UNIT L[3] REWIND UNITS L[3] AND K[20]
RETURN

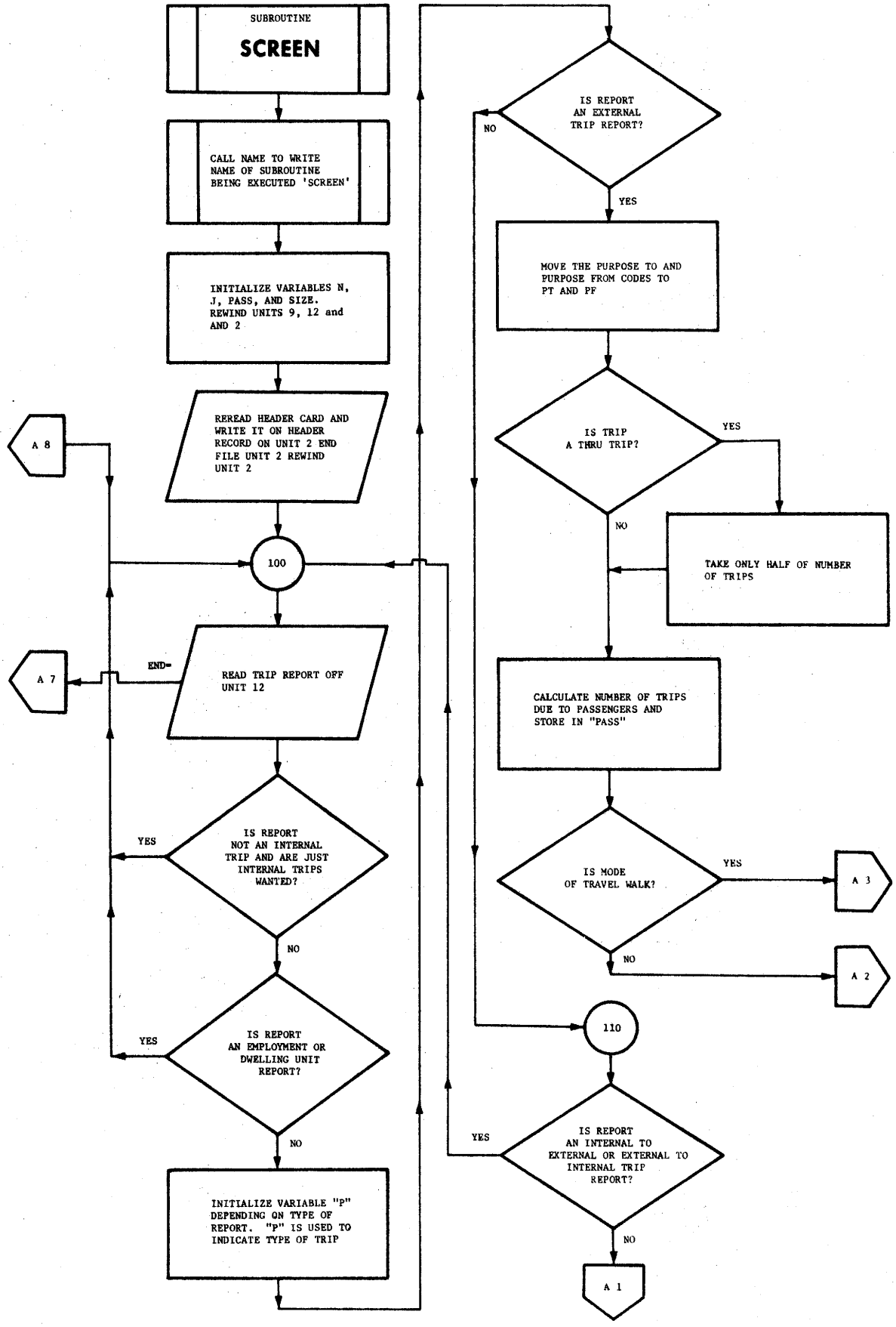

SUBROUTINE
RANK
THIS SUBROUTINE SORTS THE PRODUCTION OR ATTRACTION VOLUMES IN ASCENDING ORDER AND CARRIES ALONG WITH EACH THE RESPECTIVE CENTROID NUMBER ASSOCIATED WITH THAT VOLUME.
RETURN

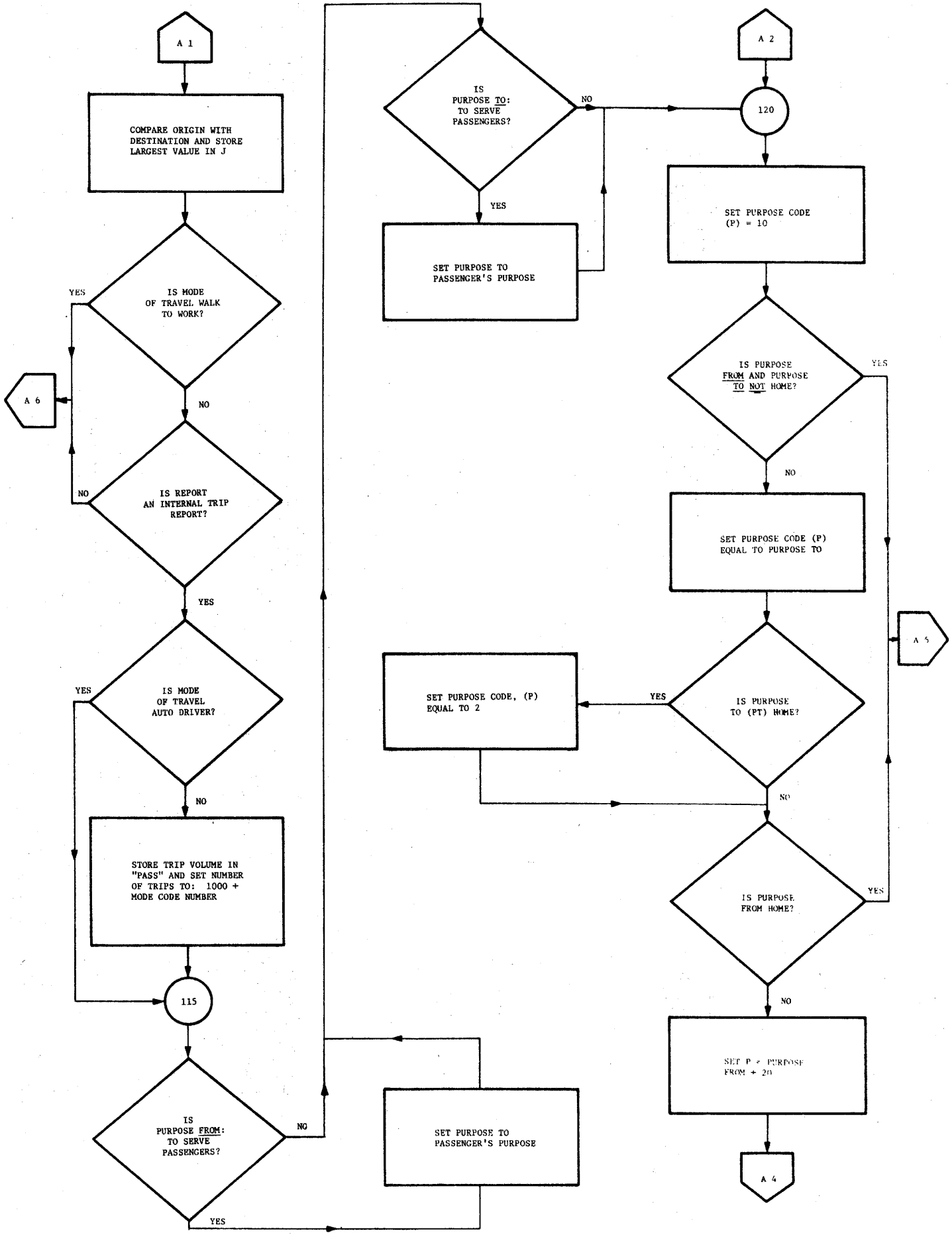

SUBROUTINE REFINE
CALL SUBROUTINE "NAME" TO PRINT OUT NAME OF SUBROUTINE BEING EXECUTED "REFINE"
SET FIRST POINT OF PRODUCTION INTERACTION CURVE TO 0, 0.
IS THE LAST CARD READ AN INTERACTION CARD?
YES
STOP 13
NO
IS THE LAST CARD READ A FORMAT CARD?
NO
YES
REREAD THE LAST CARD TO GET THE FORMAT
WRITE HEADING FOR PRODUCTION INTERACTION CURVE.
READ NEXT CARD FROM UNIT 5.
IS IT AN INTERACTION CARD?
NO
F2
YES
WRITE HEADING FOR PRODUCTION INTERACTION CURVE
WRITE 0, 0 DATA POINT
NPV = 2
WAS FIRST DATA POINT READ 0, 0?
YES
NPV = 1
NO
IS NPV = 2 ?
YES
WRITE FIRST DATA POINT
NO
PUT FIRST DATA POINT IN ARRAYS AT INDEX NPV
F1

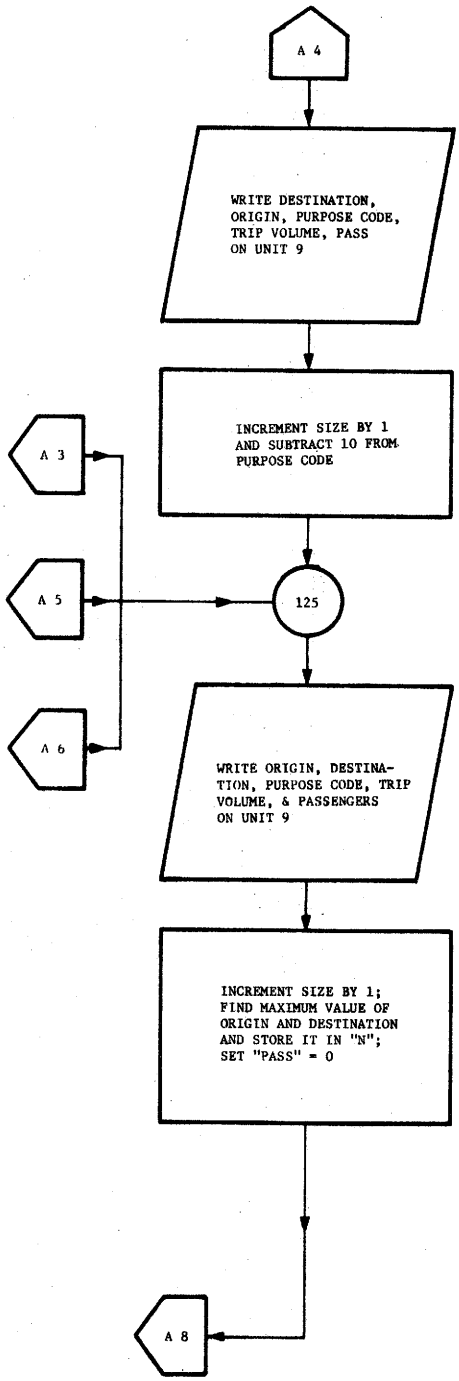

F1
READ NEXT DATA CARD FROM UNIT 5
IS IT AN INTERACTION CARD?
NO
YES
NPV = NPV + 1
REREAD THE INTERACTION CARD
PRINT THE DATA POINT
IS PROD (NPV - 1) ≥ PROD (NPV)
NO
YES
STOP 14
F2
CALL SUBROUTINE "NAME" TO PRINT OUT NAME OF SUB-ROUTINE BEING EXECUTED "REFINE"
ARE THERE ANY PROD. OR ATTRACTION ENTRIES
NO
F3
YES
SET VARIABLE "R" = TO THE RATIO OF TOTAL # INT. TRIPS TO TOTAL # OF TRIPS + # TRIPS AT MAX. INT. SEP; ALSO SET AM = 0.0
CALL SUBROUTINE "FIT" TO PRINT OUT TITLE OF WHAT IS BEING FITTED AND ITITIAL-IZE VARIABLES USED. "PROD. INTERATIONS"
DO 110 I = 1, NM (NM = NUMBER OF INT. CENTROIDS)
WAS TRIP MATRIX BUILT BY SUBROUTINE "BUILD"?
NO
CALL SUBROUTINE "ENTER" TO PERFORM SUMMATIONS ON MODEL FORMULA'S X AND Y VALUES
YES
WAS TRIP MATRIX BUILT BY SUBROUTINE "BUILD"?
YES
CALL SUBROUTINE "ENTER" TO PERFORM SUMMATIONS ON MODEL FORMULA'S X AND Y VALUES
NO
110
CALL SUBROUTINE "FORM" TO PERFORM LINEAR CURVE FIT AND RETURN COEFFICIENTS "AP AND XP"
F3

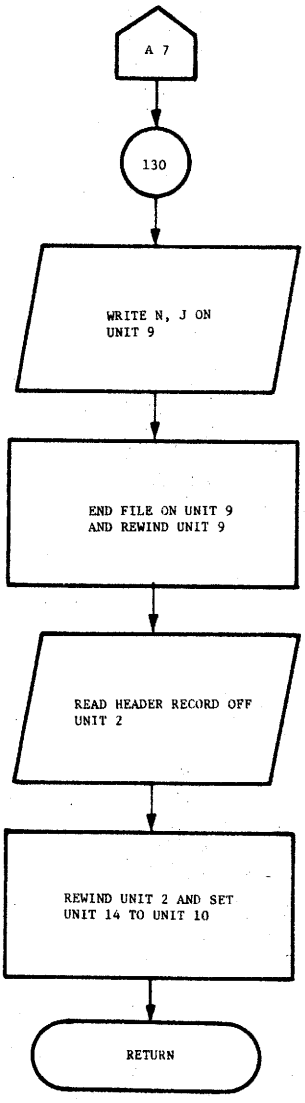

F3
DO 2 I = 1, N
FR(I) = 0.0
YES
IS
S(I) = 0
?
NO
FR(I) = F(I)/S(I)
2
INITIALIZE EXTERNAL
RELATIVE VALUES
RETURN

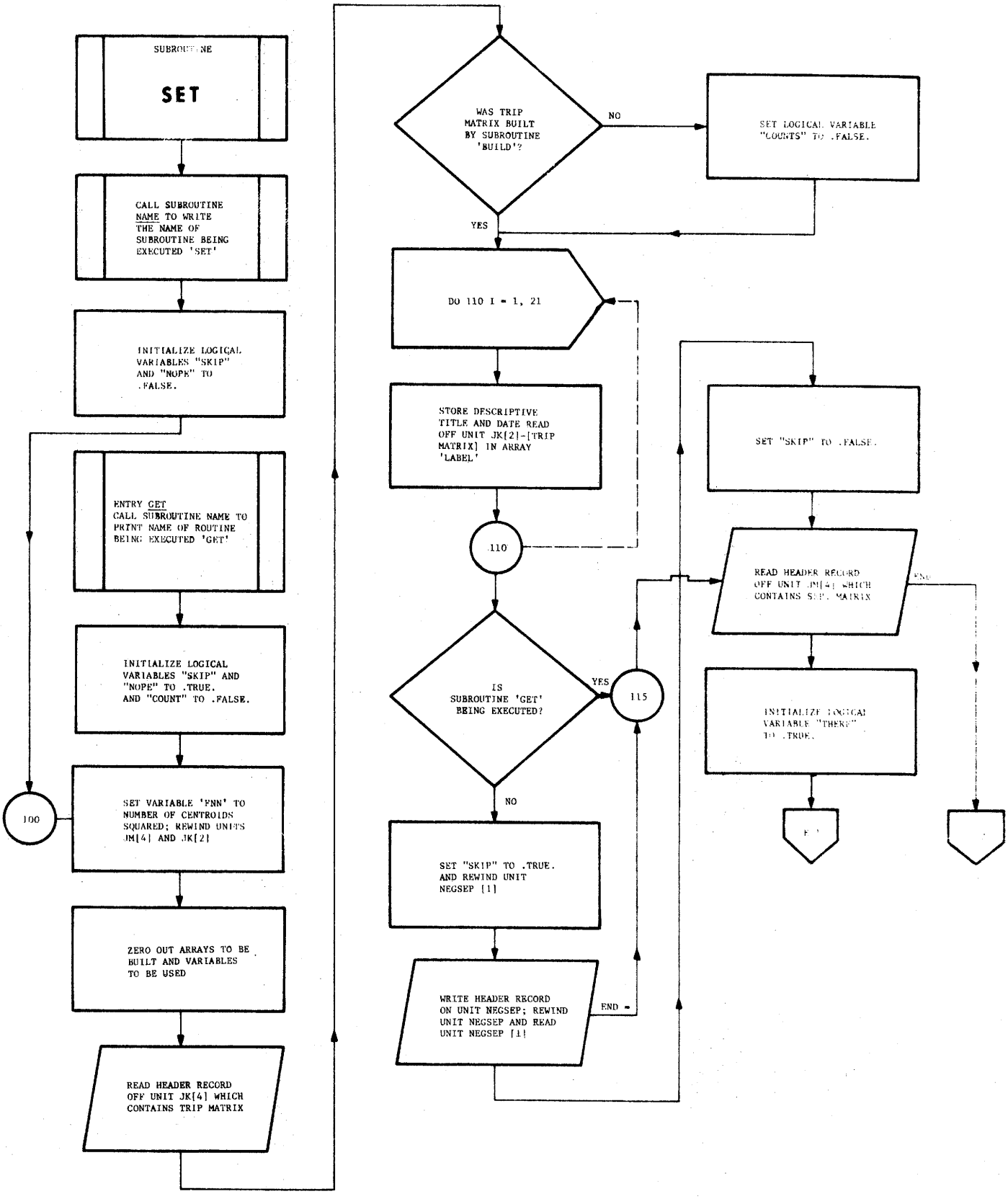

SUBROUTINE
SCAN
INITIALIZE REREAD
CALL DATE
READ CARD
END=
ABORT?
TRUE
STOP 1
FALSE
RETURN
CONTROL CARD?
YES
SCANS THE CONTROL CARD
AND DETERMINES WHICH
ROUTINES ARE TO BE
EXECUTED
NO
HEADER CARD?
YES
NO
STORE CARD IN ARRAY
HEADER

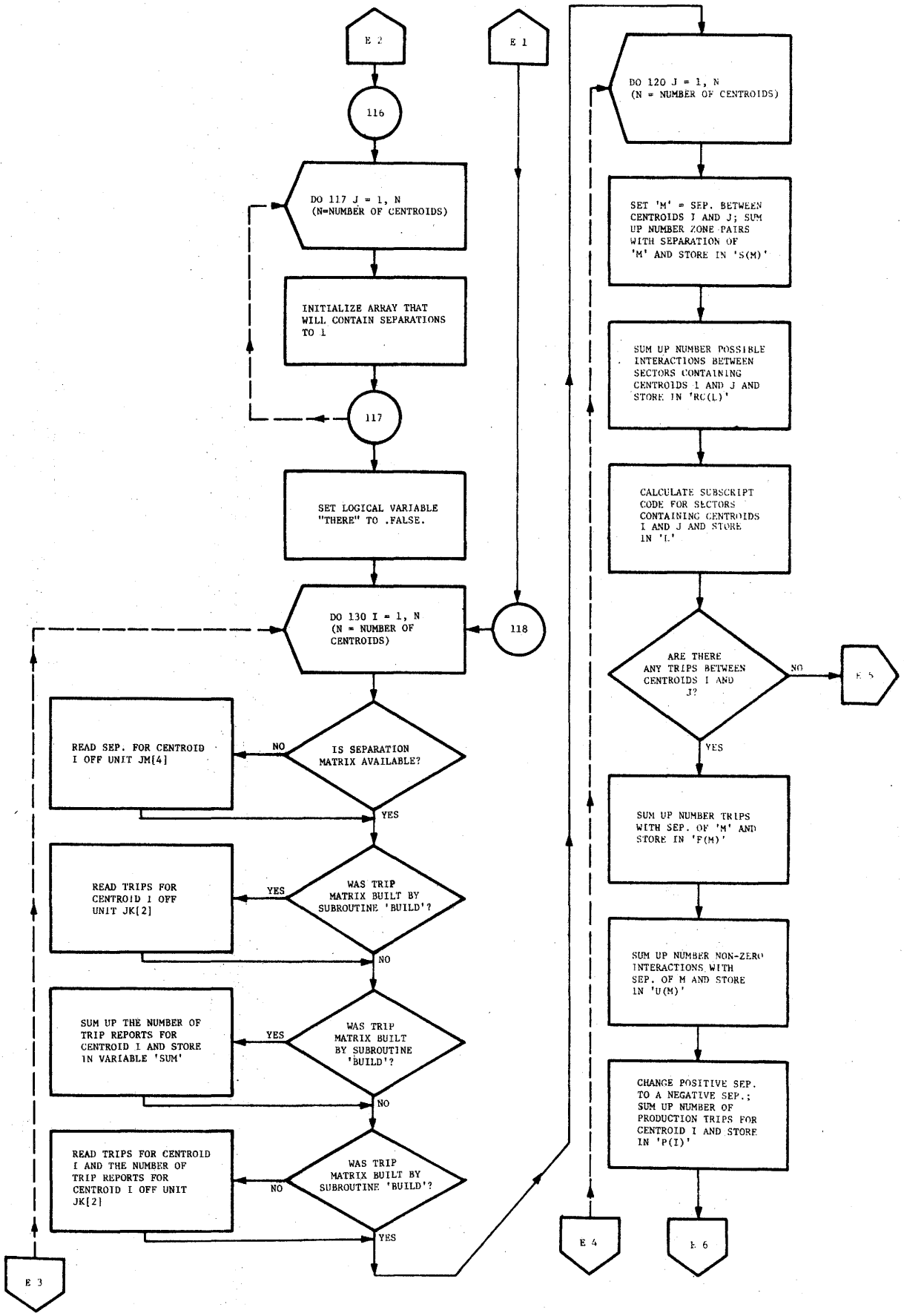

SUBROUTINE
SCREEN
CALL NAME TO WRITE NAME OF SUBROUTINE BEING EXECUTED 'SCREEN'
INITIALIZE VARIABLES N, J, PASS, AND SIZE. REWIND UNITS 9, 12 and AND 2
REREAD HEADER CARD AND WRITE IT ON HEADER RECORD ON UNIT 2 END FILE UNIT 2 REWIND UNIT 2
A 8
100
READ TRIP REPORT OFF UNIT 12
END=
A 7
IS REPORT NOT AN INTERNAL TRIP AND ARE JUST INTERNAL TRIPS WANTED?
YES
NO
IS REPORT AN EMPLOYMENT OR DWELLING UNIT REPORT?
YES
NO
INITIALIZE VARIABLE "P" DEPENDING ON TYPE OF REPORT. "P" IS USED TO INDICATE TYPE OF TRIP
IS REPORT AN EXTERNAL TRIP REPORT?
NO
YES
MOVE THE PURPOSE TO AND PURPOSE FROM CODES TO PT AND PF
IS TRIP A THRU TRIP?
YES
NO
TAKE ONLY HALF OF NUMBER OF TRIPS
CALCULATE NUMBER OF TRIPS DUE TO PASSENGERS AND STORE IN "PASS"
IS MODE OF TRAVEL WALK?
YES
A 3
NO
A 2
110
IS REPORT AN INTERNAL TO EXTERNAL OR EXTERNAL TO INTERNAL TRIP REPORT?
YES
NO
A 1

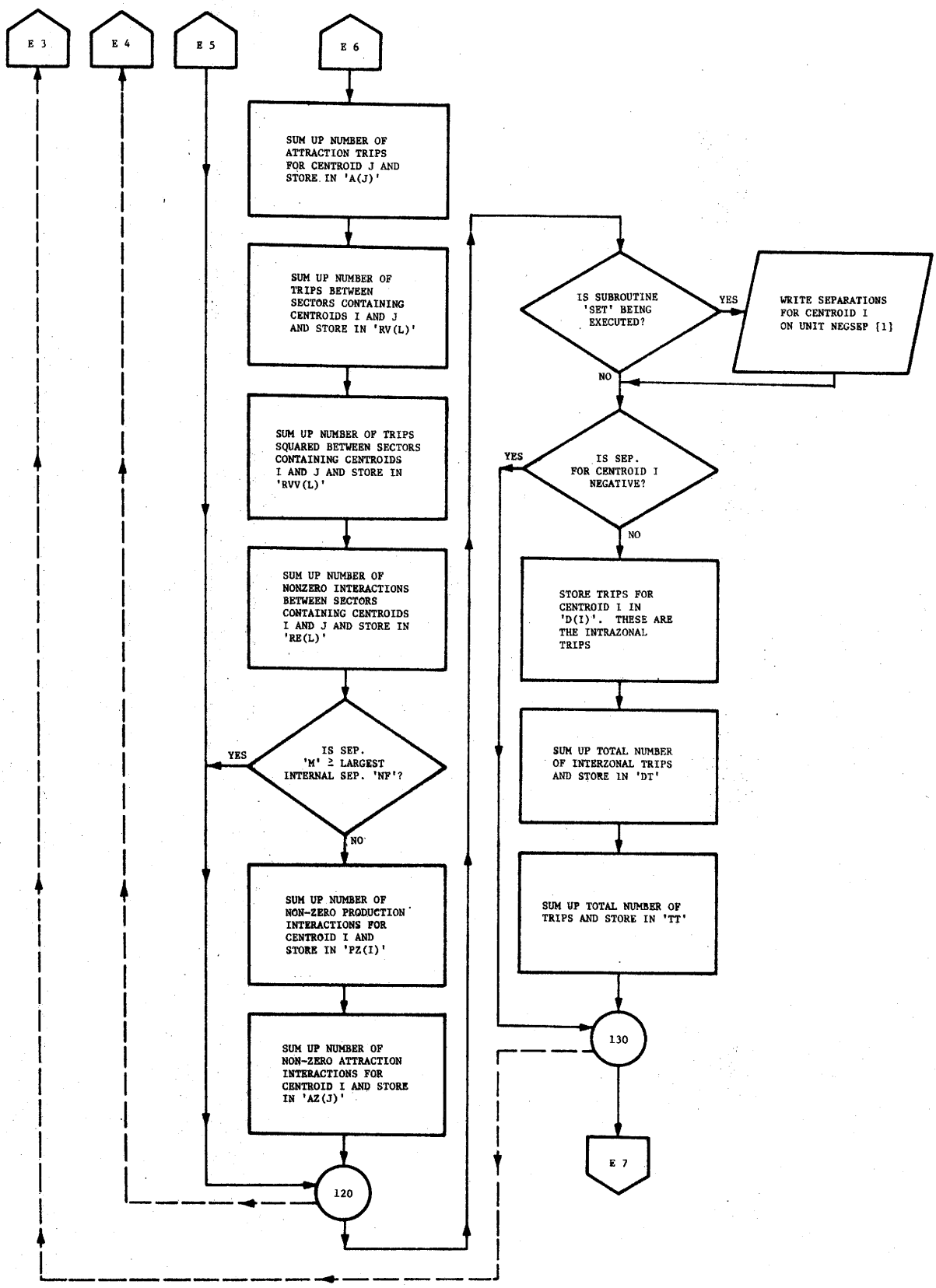

A 1
COMPARE ORIGIN WITH DESTINATION AND STORE LARGEST VALUE IN J
IS MODE OF TRAVEL WALK TO WORK?
YES
NO
A 6
IS REPORT AN INTERNAL TRIP REPORT?
NO
YES
IS MODE OF TRAVEL AUTO DRIVER?
YES
NO
STORE TRIP VOLUME IN "PASS" AND SET NUMBER OF TRIPS TO: 1000 + MODE CODE NUMBER
115
IS PURPOSE FROM: TO SERVE PASSENGERS?
NO
YES
SET PURPOSE TO PASSENGER'S PURPOSE
IS PURPOSE TO: TO SERVE PASSENGERS?
NO
YES
SET PURPOSE TO PASSENGER'S PURPOSE
A 2
120
SET PURPOSE CODE (P) = 10
IS PURPOSE FROM AND PURPOSE TO NOT HOME?
YES
NO
SET PURPOSE CODE (P) EQUAL TO PURPOSE TO
A 5
IS PURPOSE TO (PT) HOME?
YES
SET PURPOSE CODE, (P) EQUAL TO 2
NO
IS PURPOSE FROM HOME?
YES
NO
SET P = PURPOSE FROM + 20
A 4

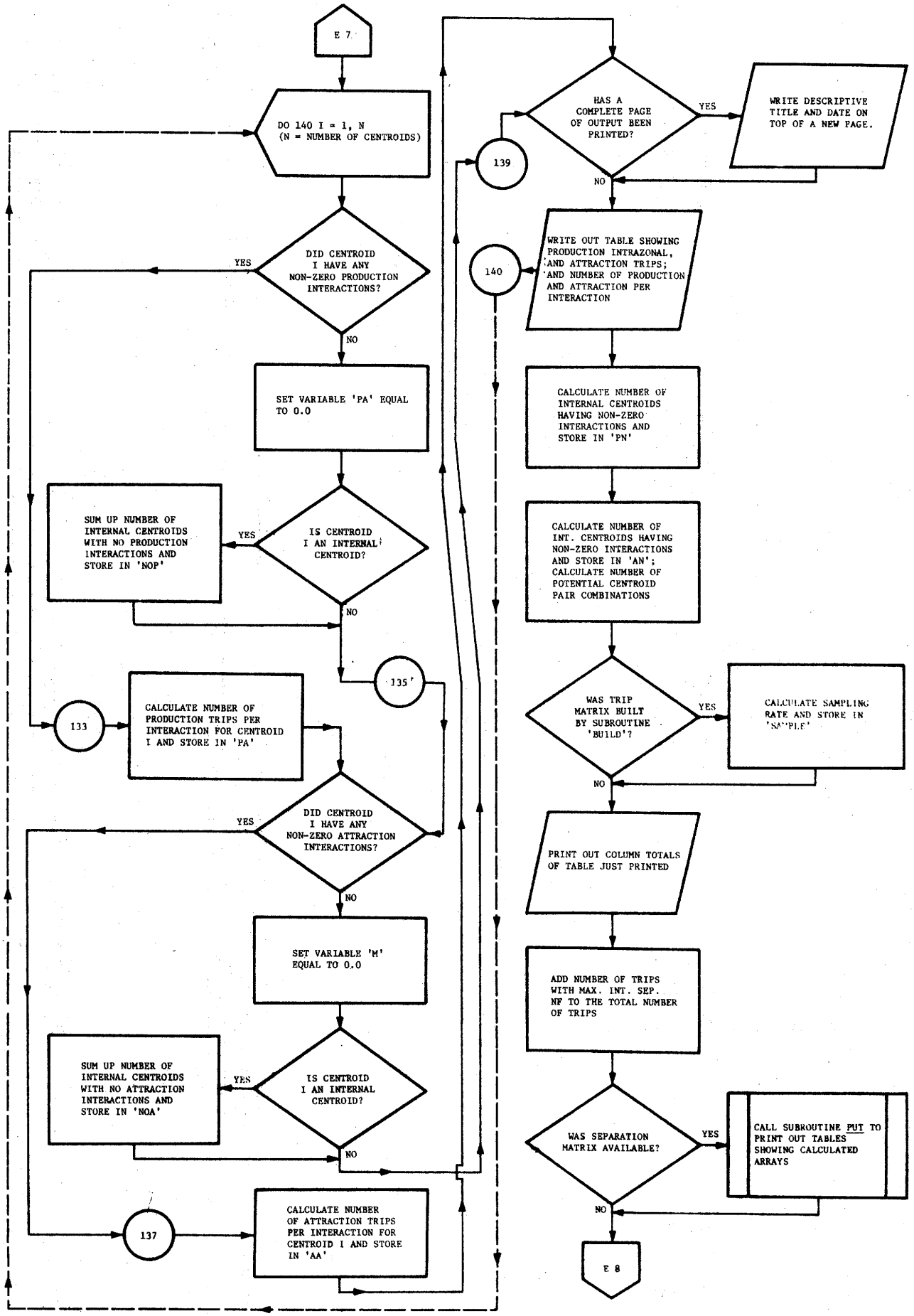

A 4
WRITE DESTINATION, ORIGIN, PURPOSE CODE, TRIP VOLUME, PASS ON UNIT 9
INCREMENT SIZE BY 1 AND SUBTRACT 10 FROM PURPOSE CODE
A 3
A 5
A 6
125
WRITE ORIGIN, DESTINATION, PURPOSE CODE, TRIP VOLUME, & PASSENGERS ON UNIT 9
INCREMENT SIZE BY 1; FIND MAXIMUM VALUE OF ORIGIN AND DESTINATION AND STORE IT IN "N"; SET "PASS" = 0
A 8

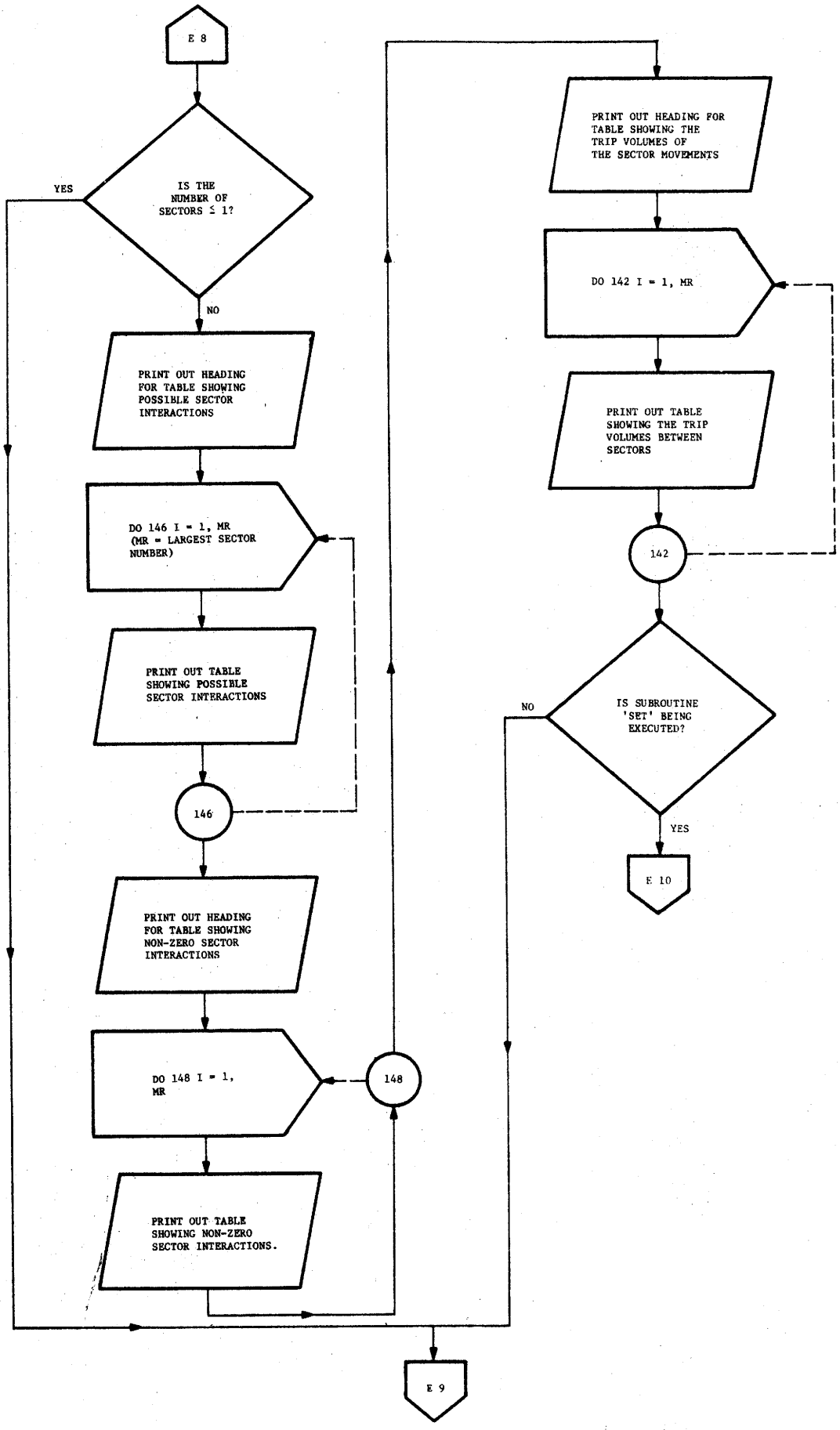

A 7
130
WRITE N, J ON UNIT 9
END FILE ON UNIT 9 AND REWIND UNIT 9
READ HEADER RECORD OFF UNIT 2
REWIND UNIT 2 AND SET UNIT 14 TO UNIT 10
RETURN

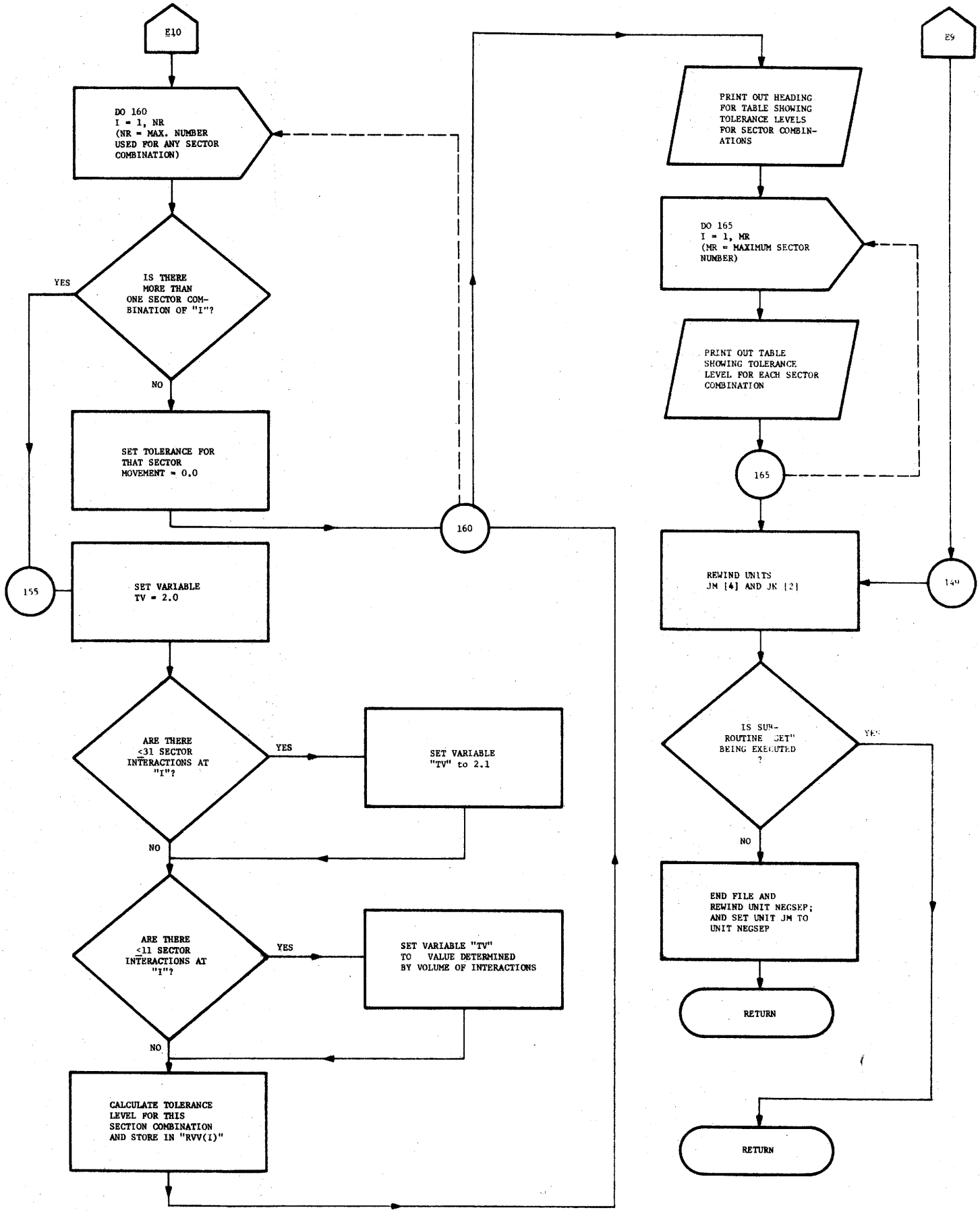

SUBROUTINE
SET
CALL SUBROUTINE NAME TO WRITE THE NAME OF SUBROUTINE BEING EXECUTED 'SET'
INITIALIZE LOGICAL VARIABLES "SKIP" AND "NOPE" TO .FALSE.
ENTRY GET
CALL SUBROUTINE NAME TO PRINT NAME OF ROUTINE BEING EXECUTED 'GET'
INITIALIZE LOGICAL VARIABLES "SKIP" AND "NOPE" TO .TRUE. AND "COUNT" TO .FALSE.
100
SET VARIABLE 'FNN' TO NUMBER OF CENTROIDS SQUARED; REWIND UNITS JM[4] AND JK[2]
ZERO OUT ARRAYS TO BE BUILT AND VARIABLES TO BE USED
READ HEADER RECORD OFF UNIT JK[4] WHICH CONTAINS TRIP MATRIX
WAS TRIP MATRIX BUILT BY SUBROUTINE 'BUILD'?
NO
SET LOGICAL VARIABLE "COUNTS" TO .FALSE.
YES
DO 110 I = 1, 21
STORE DESCRIPTIVE TITLE AND DATE READ OFF UNIT JK[2]-[TRIP MATRIX] IN ARRAY 'LABEL'
110
IS SUBROUTINE 'GET' BEING EXECUTED?
YES
115
NO
SET "SKIP" TO .TRUE. AND REWIND UNIT NEGSEP [1]
WRITE HEADER RECORD ON UNIT NEGSEP; REWIND UNIT NEGSEP AND READ UNIT NEGSEP [1]
END =
SET "SKIP" TO .FALSE.
READ HEADER RECORD OFF UNIT JM[4] WHICH CONTAINS SEP. MATRIX
INITIALIZE LOGICAL VARIABLE "THERE" TO .TRUE.

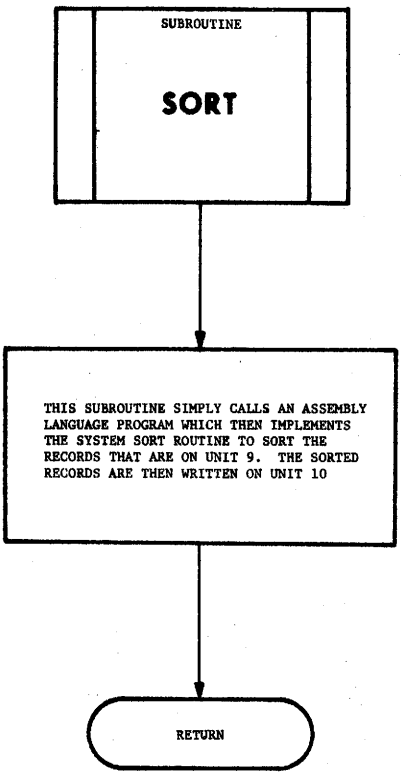

E 2
E 1
116
DO 117 J = 1, N
(N=NUMBER OF CENTROIDS)
INITIALIZE ARRAY THAT WILL CONTAIN SEPARATIONS TO 1
117
SET LOGICAL VARIABLE "THERE" TO .FALSE.
DO 130 I = 1, N
(N = NUMBER OF CENTROIDS)
118
IS SEPARATION MATRIX AVAILABLE?
NO
READ SEP. FOR CENTROID I OFF UNIT JM[4]
YES
WAS TRIP MATRIX BUILT BY SUBROUTINE 'BUILD'?
YES
READ TRIPS FOR CENTROID I OFF UNIT JK[2]
NO
WAS TRIP MATRIX BUILT BY SUBROUTINE 'BUILD'?
YES
SUM UP THE NUMBER OF TRIP REPORTS FOR CENTROID I AND STORE IN VARIABLE 'SUM'
NO
WAS TRIP MATRIX BUILT BY SUBROUTINE 'BUILD'?
NO
READ TRIPS FOR CENTROID I AND THE NUMBER OF TRIP REPORTS FOR CENTROID I OFF UNIT JK[2]
YES
E 3
DO 120 J = 1, N
(N = NUMBER OF CENTROIDS)
SET 'M' = SEP. BETWEEN CENTROIDS I AND J; SUM UP NUMBER ZONE PAIRS WITH SEPARATION OF 'M' AND STORE IN 'S(M)'
SUM UP NUMBER POSSIBLE INTERACTIONS BETWEEN SECTORS CONTAINING CENTROIDS I AND J AND STORE IN 'RC(L)'
CALCULATE SUBSCRIPT CODE FOR SECTORS CONTAINING CENTROIDS I AND J AND STORE IN 'L'
ARE THERE ANY TRIPS BETWEEN CENTROIDS I AND J?
NO
E 5
YES
SUM UP NUMBER TRIPS WITH SEP. OF 'M' AND STORE IN 'F(M)'
SUM UP NUMBER NON-ZERO INTERACTIONS WITH SEP. OF M AND STORE IN 'U(M)'
CHANGE POSITIVE SEP. TO A NEGATIVE SEP.; SUM UP NUMBER OF PRODUCTION TRIPS FOR CENTROID I AND STORE IN 'P(I)'
E 4
E 6

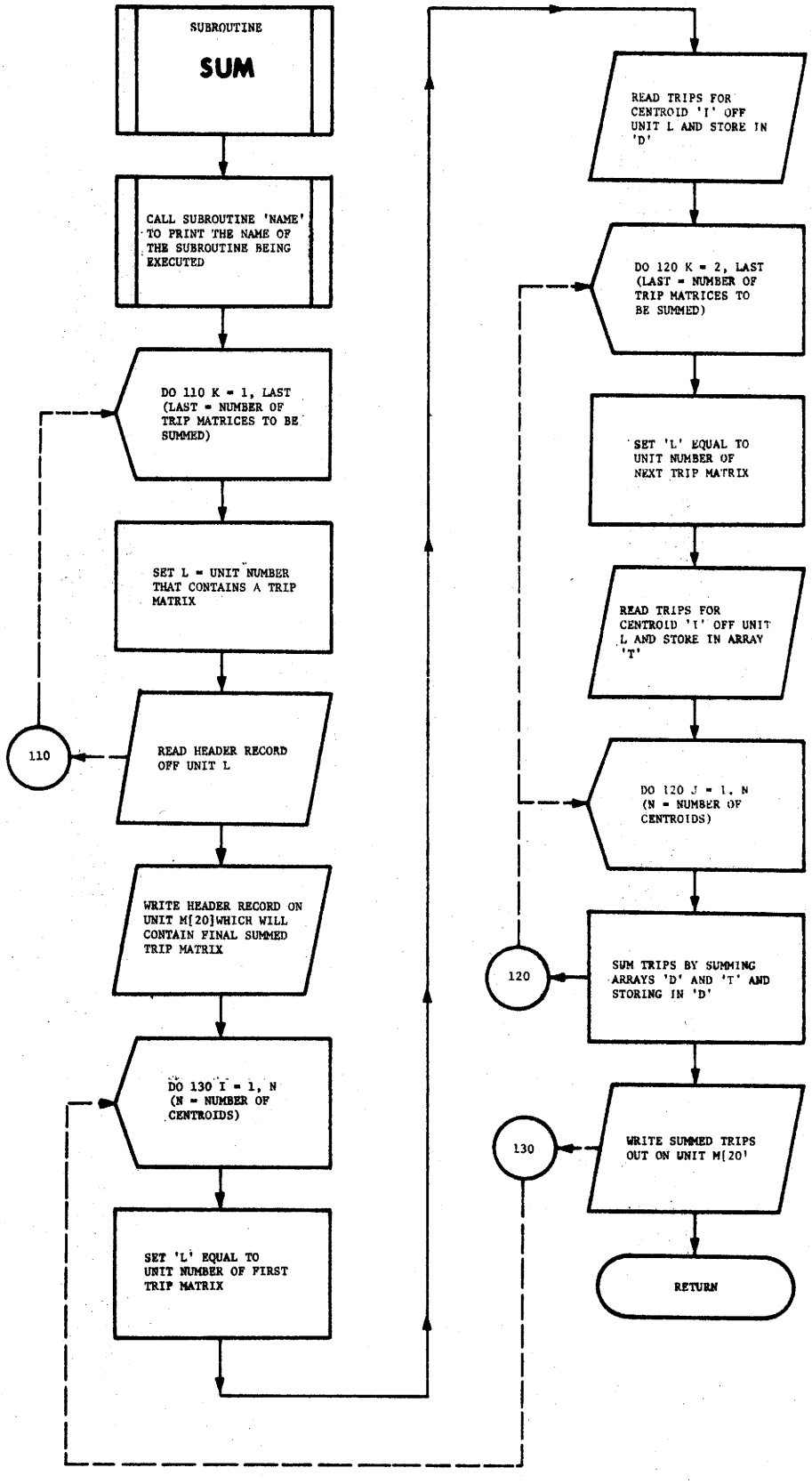

E 3
E 4
E 5
E 6
SUM UP NUMBER OF ATTRACTION TRIPS FOR CENTROID J AND STORE IN 'A(J)'
SUM UP NUMBER OF TRIPS BETWEEN SECTORS CONTAINING CENTROIDS I AND J AND STORE IN 'RV(L)'
SUM UP NUMBER OF TRIPS SQUARED BETWEEN SECTORS CONTAINING CENTROIDS I AND J AND STORE IN 'RVV(L)'
SUM UP NUMBER OF NONZERO INTERACTIONS BETWEEN SECTORS CONTAINING CENTROIDS I AND J AND STORE IN 'RE(L)'
IS SEP. 'M' ≥ LARGEST INTERNAL SEP. 'NF'?
YES
NO
SUM UP NUMBER OF NON-ZERO PRODUCTION INTERACTIONS FOR CENTROID I AND STORE IN 'PZ(I)'
SUM UP NUMBER OF NON-ZERO ATTRACTION INTERACTIONS FOR CENTROID I AND STORE IN 'AZ(J)'
120
IS SUBROUTINE 'SET' BEING EXECUTED?
YES
WRITE SEPARATIONS FOR CENTROID I ON UNIT NEGSEP [1]
NO
IS SEP. FOR CENTROID I NEGATIVE?
YES
NO
STORE TRIPS FOR CENTROID I IN 'D(I)'. THESE ARE THE INTRAZONAL TRIPS
SUM UP TOTAL NUMBER OF INTERZONAL TRIPS AND STORE IN 'DT'
SUM UP TOTAL NUMBER OF TRIPS AND STORE IN 'TT'
130
E 7

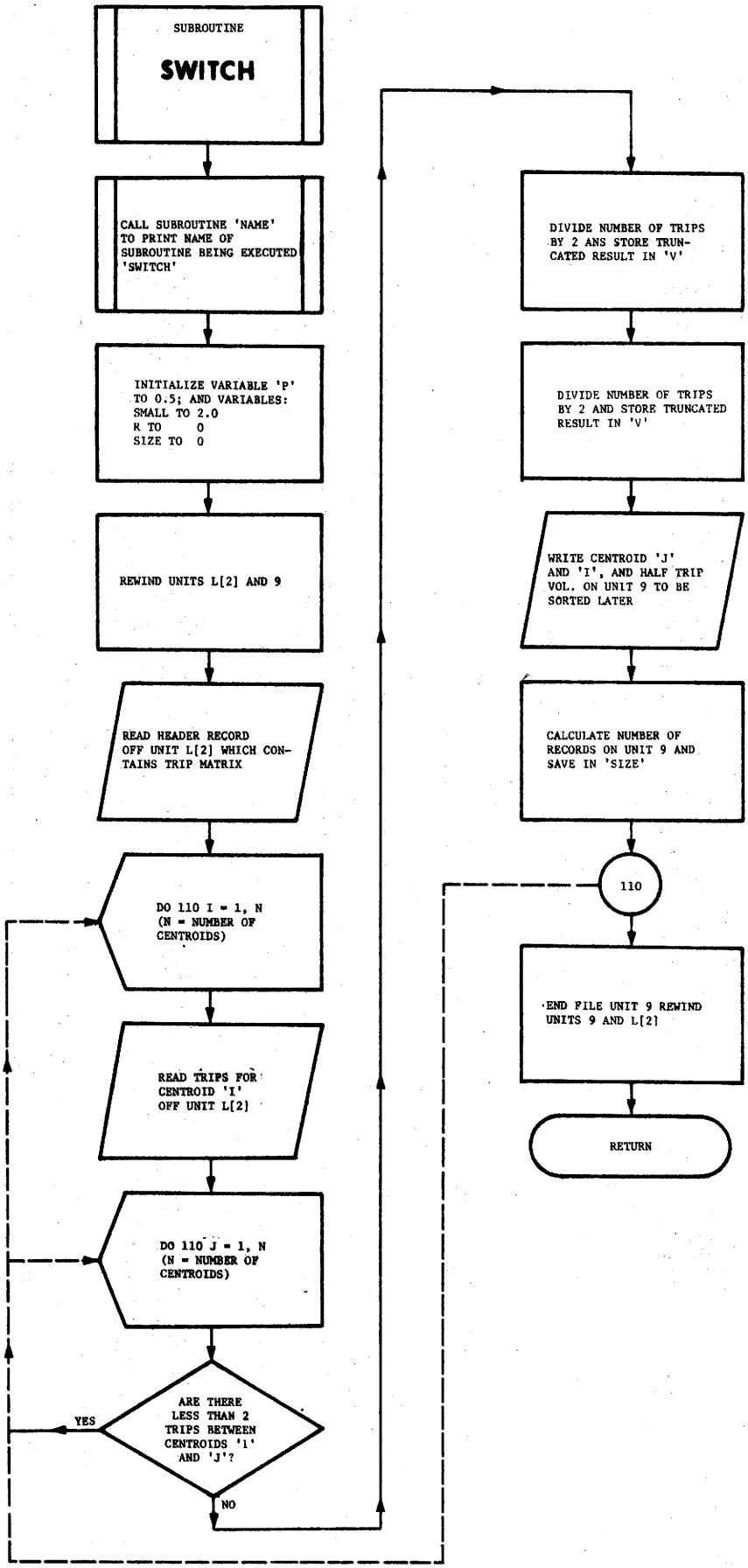

E 7
DO 140 I = 1, N
(N = NUMBER OF CENTROIDS)
DID CENTROID I HAVE ANY NON-ZERO PRODUCTION INTERACTIONS?
YES
NO
SET VARIABLE 'PA' EQUAL TO 0.0
IS CENTROID I AN INTERNAL CENTROID?
YES
SUM UP NUMBER OF INTERNAL CENTROIDS WITH NO PRODUCTION INTERACTIONS AND STORE IN 'NOP'
NO
135
133
CALCULATE NUMBER OF PRODUCTION TRIPS PER INTERACTION FOR CENTROID I AND STORE IN 'PA'
DID CENTROID I HAVE ANY NON-ZERO ATTRACTION INTERACTIONS?
YES
NO
SET VARIABLE 'M' EQUAL TO 0.0
IS CENTROID I AN INTERNAL CENTROID?
YES
SUM UP NUMBER OF INTERNAL CENTROIDS WITH NO ATTRACTION INTERACTIONS AND STORE IN 'NOA'
NO
137
CALCULATE NUMBER OF ATTRACTION TRIPS PER INTERACTION FOR CENTROID I AND STORE IN 'AA'
139
HAS A COMPLETE PAGE OF OUTPUT BEEN PRINTED?
YES
WRITE DESCRIPTIVE TITLE AND DATE ON TOP OF A NEW PAGE.
NO
140
WRITE OUT TABLE SHOWING PRODUCTION INTRAZONAL, AND ATTRACTION TRIPS; AND NUMBER OF PRODUCTION AND ATTRACTION PER INTERACTION
CALCULATE NUMBER OF INTERNAL CENTROIDS HAVING NON-ZERO INTERACTIONS AND STORE IN 'PN'
CALCULATE NUMBER OF INT. CENTROIDS HAVING NON-ZERO INTERACTIONS AND STORE IN 'AN'; CALCULATE NUMBER OF POTENTIAL CENTROID PAIR COMBINATIONS
WAS TRIP MATRIX BUILT BY SUBROUTINE 'BUILD'?
YES
CALCULATE SAMPLING RATE AND STORE IN 'SAMPLE'
NO
PRINT OUT COLUMN TOTALS OF TABLE JUST PRINTED
ADD NUMBER OF TRIPS WITH MAX. INT. SEP. NF TO THE TOTAL NUMBER OF TRIPS
WAS SEPARATION MATRIX AVAILABLE?
YES
CALL SUBROUTINE PUT TO PRINT OUT TABLES SHOWING CALCULATED ARRAYS
NO
E 8

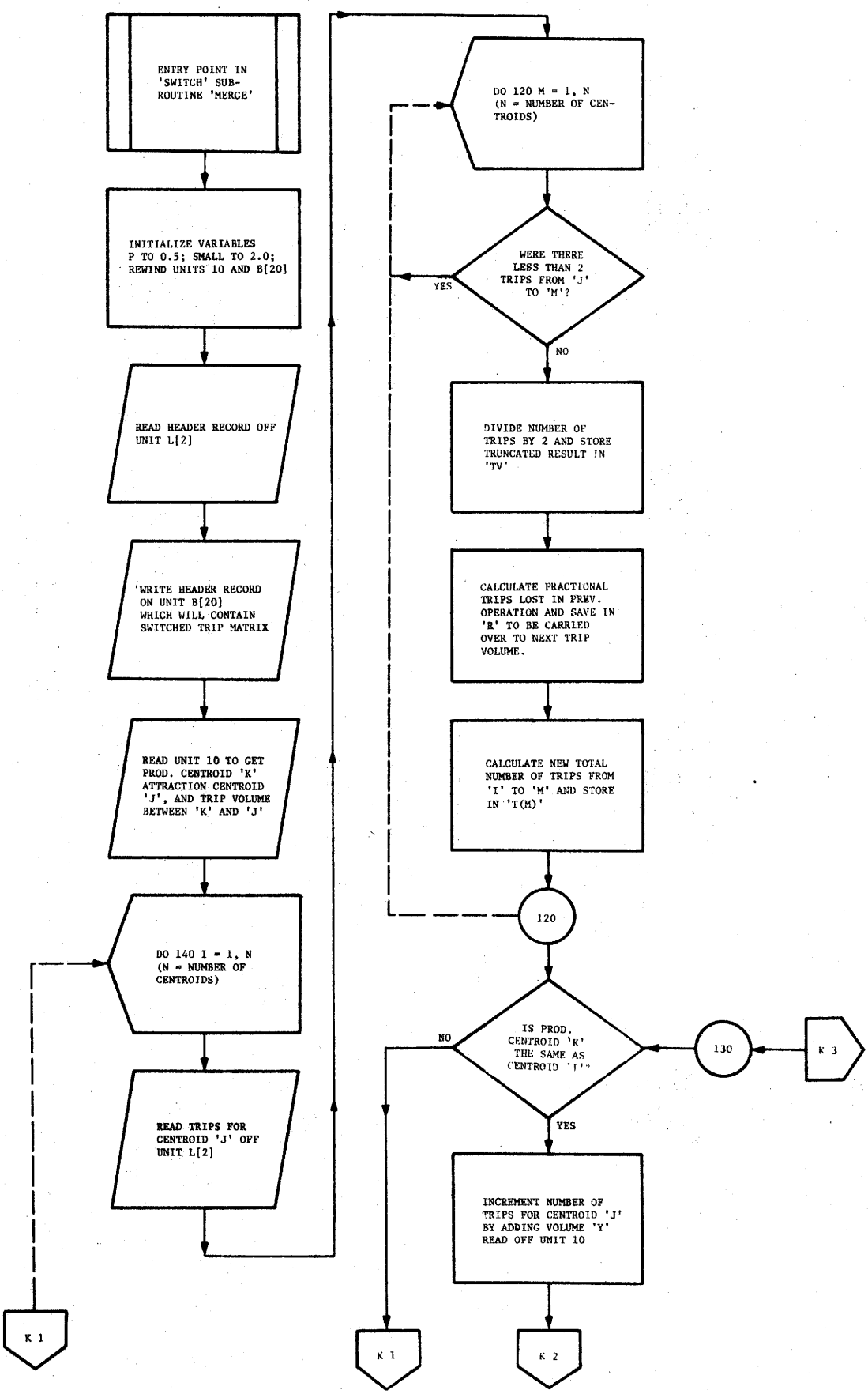

E 8
IS THE NUMBER OF SECTORS ≤ 1?
YES
NO
PRINT OUT HEADING FOR TABLE SHOWING POSSIBLE SECTOR INTERACTIONS
DO 146 I = 1, MR (MR = LARGEST SECTOR NUMBER)
PRINT OUT TABLE SHOWING POSSIBLE SECTOR INTERACTIONS
146
PRINT OUT HEADING FOR TABLE SHOWING NON-ZERO SECTOR INTERACTIONS
DO 148 I = 1, MR
148
PRINT OUT TABLE SHOWING NON-ZERO SECTOR INTERACTIONS.
E 9
PRINT OUT HEADING FOR TABLE SHOWING THE TRIP VOLUMES OF THE SECTOR MOVEMENTS
DO 142 I = 1, MR
PRINT OUT TABLE SHOWING THE TRIP VOLUMES BETWEEN SECTORS
142
IS SUBROUTINE 'SET' BEING EXECUTED?
NO
YES
E 10

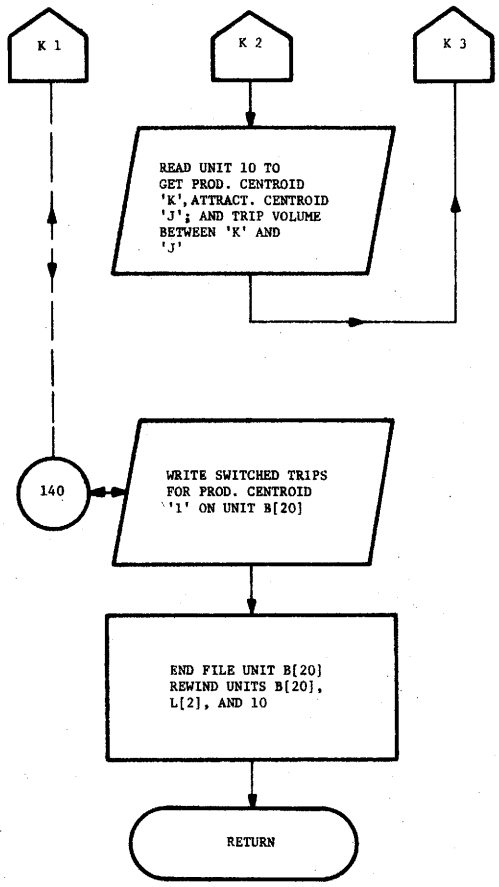

E10
DO 160
I = 1, NR
(NR = MAX. NUMBER
USED FOR ANY SECTOR
COMBINATION)
IS THERE
MORE THAN
ONE SECTOR COM-
BINATION OF "I"?
YES
NO
SET TOLERANCE FOR
THAT SECTOR
MOVEMENT = 0.0
160
155
SET VARIABLE
TV = 2.0
ARE THERE
<31 SECTOR
INTERACTIONS AT
"I"?
YES
SET VARIABLE
"TV" to 2.1
NO
ARE THERE
<11 SECTOR
INTERACTIONS AT
"I"?
YES
SET VARIABLE "TV"
TO VALUE DETERMINED
BY VOLUME OF INTERACTIONS
NO
CALCULATE TOLERANCE
LEVEL FOR THIS
SECTION COMBINATION
AND STORE IN "RVV(I)"
E9
PRINT OUT HEADING
FOR TABLE SHOWING
TOLERANCE LEVELS
FOR SECTOR COMBIN-
ATIONS
DO 165
I = 1, MR
(MR = MAXIMUM SECTOR
NUMBER)
PRINT OUT TABLE
SHOWING TOLERANCE
LEVEL FOR EACH SECTOR
COMBINATION
165
140
REWIND UNITS
JM [4] AND JK [2]
IS SUB-
ROUTINE "SET"
BEING EXECUTED
?
YES
NO
END FILE AND
REWIND UNIT NEGSEP;
AND SET UNIT JM TO
UNIT NEGSEP
RETURN
RETURN

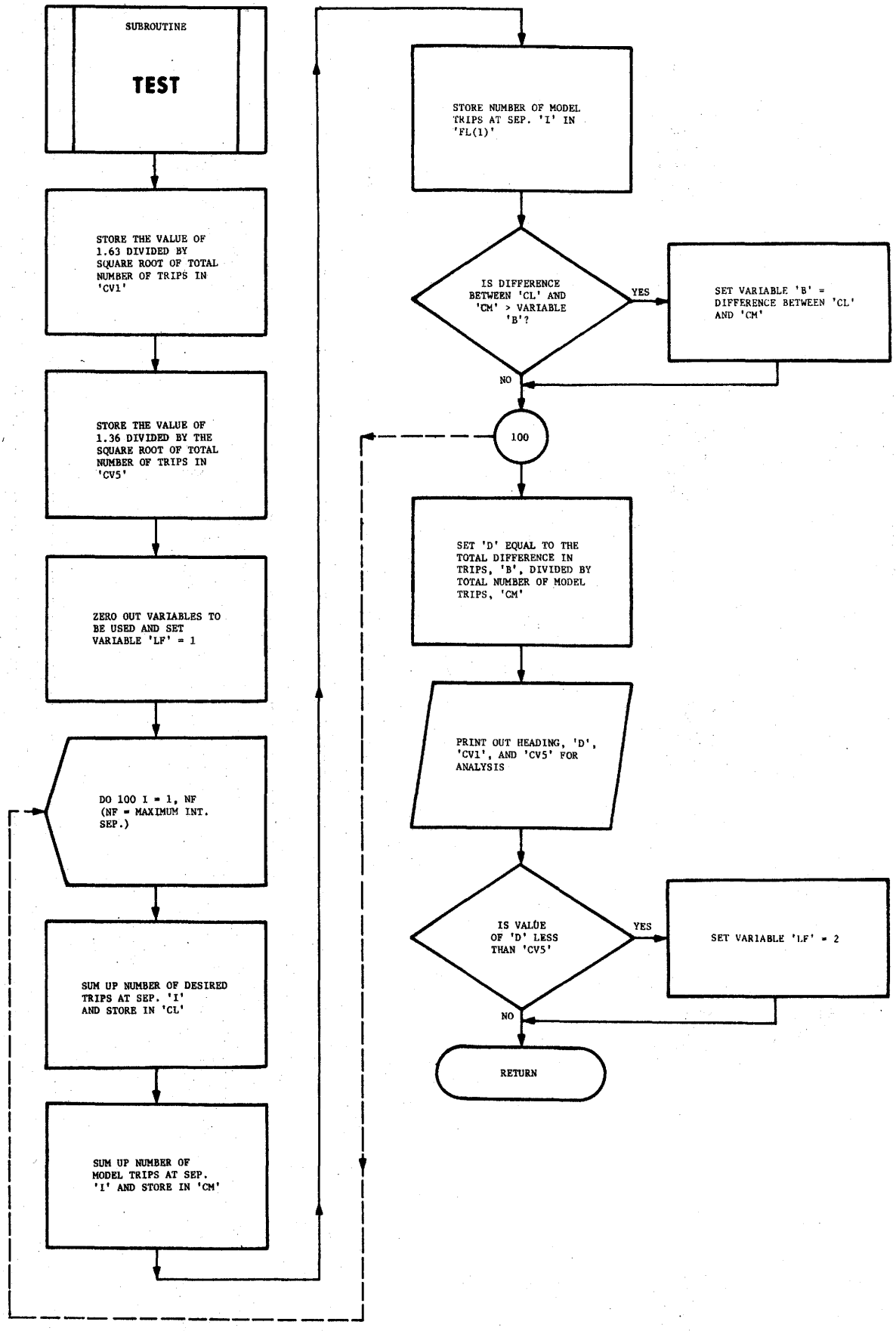

SUBROUTINE
SORT
THIS SUBROUTINE SIMPLY CALLS AN ASSEMBLY LANGUAGE PROGRAM WHICH THEN IMPLEMENTS THE SYSTEM SORT ROUTINE TO SORT THE RECORDS THAT ARE ON UNIT 9. THE SORTED RECORDS ARE THEN WRITTEN ON UNIT 10
RETURN

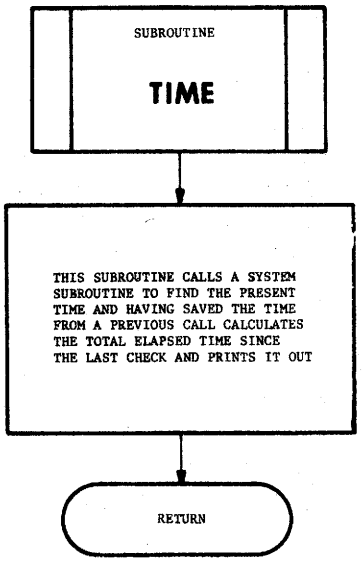

SUBROUTINE
SUM
CALL SUBROUTINE 'NAME' TO PRINT THE NAME OF THE SUBROUTINE BEING EXECUTED
DO 110 K = 1, LAST (LAST = NUMBER OF TRIP MATRICES TO BE SUMMED)
SET L = UNIT NUMBER THAT CONTAINS A TRIP MATRIX
110
READ HEADER RECORD OFF UNIT L
WRITE HEADER RECORD ON UNIT M[20]WHICH WILL CONTAIN FINAL SUMMED TRIP MATRIX
DO 130 I = 1, N (N = NUMBER OF CENTROIDS)
SET 'L' EQUAL TO UNIT NUMBER OF FIRST TRIP MATRIX
READ TRIPS FOR CENTROID 'I' OFF UNIT L AND STORE IN 'D'
DO 120 K = 2, LAST (LAST = NUMBER OF TRIP MATRICES TO BE SUMMED)
SET 'L' EQUAL TO UNIT NUMBER OF NEXT TRIP MATRIX
READ TRIPS FOR CENTROID 'I' OFF UNIT L AND STORE IN ARRAY 'T'
DO 120 J = 1, N (N = NUMBER OF CENTROIDS)
120
SUM TRIPS BY SUMMING ARRAYS 'D' AND 'T' AND STORING IN 'D'
130
WRITE SUMMED TRIPS OUT ON UNIT M[20]
RETURN

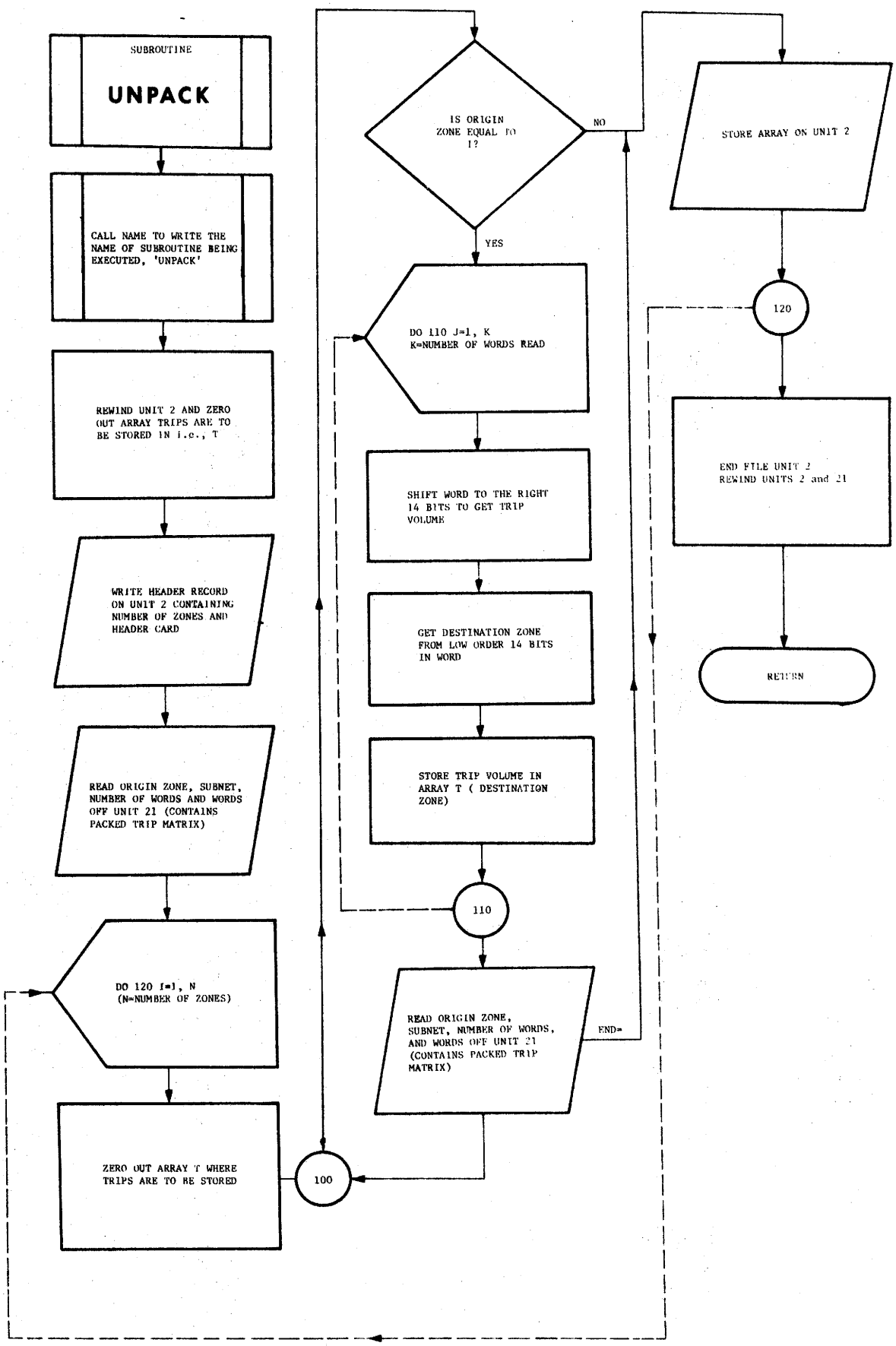

SUBROUTINE
SWITCH
CALL SUBROUTINE 'NAME' TO PRINT NAME OF SUBROUTINE BEING EXECUTED 'SWITCH'
INITIALIZE VARIABLE 'P' TO 0.5; AND VARIABLES: SMALL TO 2.0 R TO 0 SIZE TO 0
REWIND UNITS L[2] AND 9
READ HEADER RECORD OFF UNIT L[2] WHICH CONTAINS TRIP MATRIX
DO 110 I = 1, N (N = NUMBER OF CENTROIDS)
READ TRIPS FOR CENTROID 'I' OFF UNIT L[2]
DO 110 J = 1, N (N = NUMBER OF CENTROIDS)
ARE THERE LESS THAN 2 TRIPS BETWEEN CENTROIDS 'I' AND 'J'?
YES
NO
DIVIDE NUMBER OF TRIPS BY 2 ANS STORE TRUNCATED RESULT IN 'V'
DIVIDE NUMBER OF TRIPS BY 2 AND STORE TRUNCATED RESULT IN 'V'
WRITE CENTROID 'J' AND 'I', AND HALF TRIP VOL. ON UNIT 9 TO BE SORTED LATER
CALCULATE NUMBER OF RECORDS ON UNIT 9 AND SAVE IN 'SIZE'
110
END FILE UNIT 9 REWIND UNITS 9 AND L[2]
RETURN

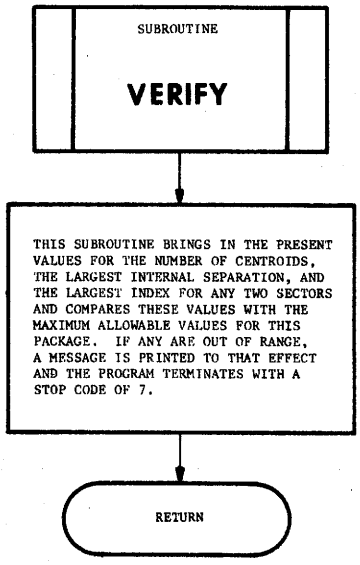

ENTRY POINT IN 'SWITCH' SUB-ROUTINE 'MERGE'
INITIALIZE VARIABLES P TO 0.5; SMALL TO 2.0; REWIND UNITS 10 AND B[20]
READ HEADER RECORD OFF UNIT L[2]
WRITE HEADER RECORD ON UNIT B[20] WHICH WILL CONTAIN SWITCHED TRIP MATRIX
READ UNIT 10 TO GET PROD. CENTROID 'K' ATTRACTION CENTROID 'J', AND TRIP VOLUME BETWEEN 'K' AND 'J'
DO 140 I = 1, N (N = NUMBER OF CENTROIDS)
READ TRIPS FOR CENTROID 'J' OFF UNIT L[2]
K 1
DO 120 M = 1, N (N = NUMBER OF CENTROIDS)
WERE THERE LESS THAN 2 TRIPS FROM 'J' TO 'M'?
YES
NO
DIVIDE NUMBER OF TRIPS BY 2 AND STORE TRUNCATED RESULT IN 'TV'
CALCULATE FRACTIONAL TRIPS LOST IN PREV. OPERATION AND SAVE IN 'R' TO BE CARRIED OVER TO NEXT TRIP VOLUME.
CALCULATE NEW TOTAL NUMBER OF TRIPS FROM 'I' TO 'M' AND STORE IN 'T(M)'
120
IS PROD. CENTROID 'K' THE SAME AS CENTROID 'I'?
NO
130
K 3
YES
INCREMENT NUMBER OF TRIPS FOR CENTROID 'J' BY ADDING VOLUME 'Y' READ OFF UNIT 10
K 1
K 2

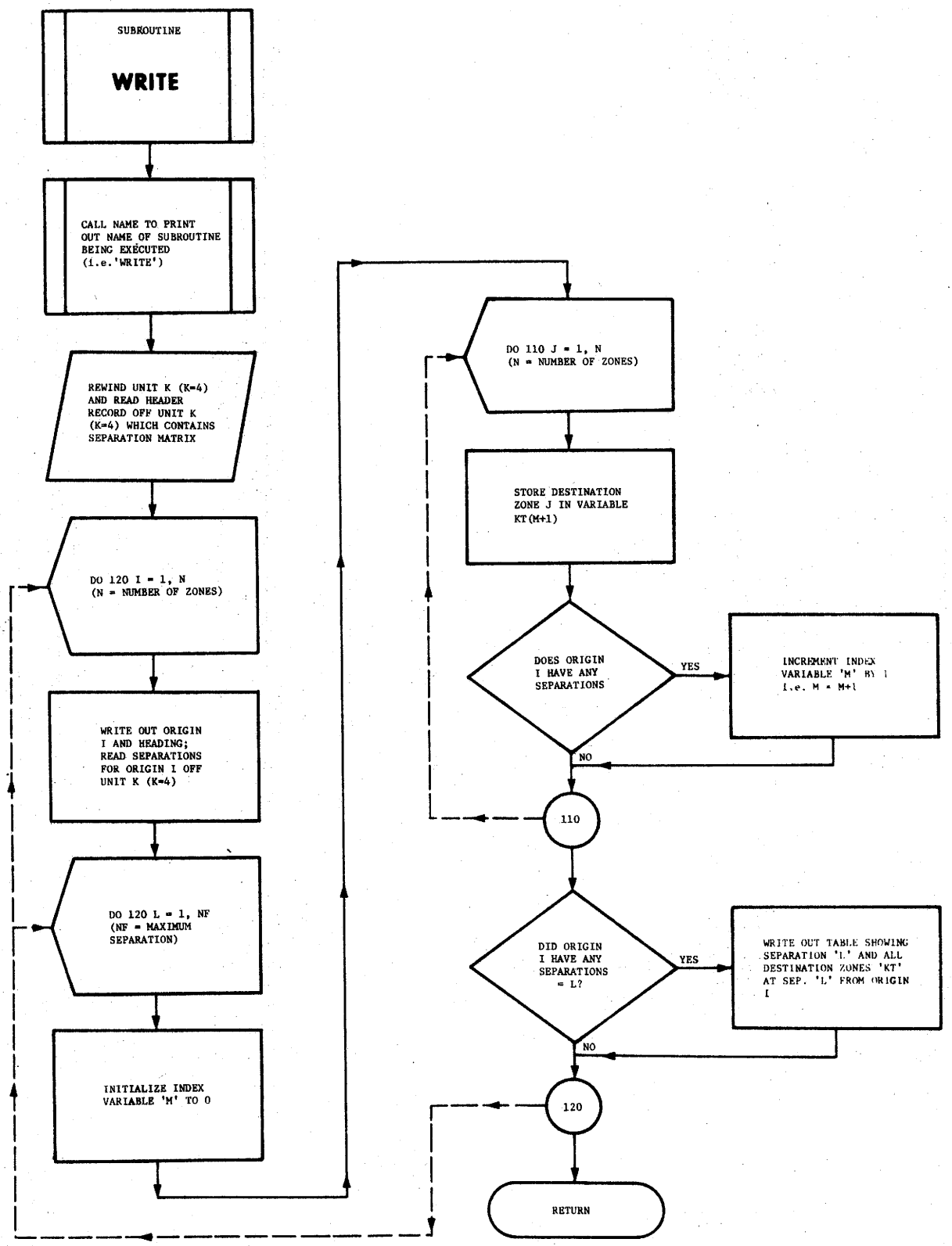

K 1
K 2
K 3
READ UNIT 10 TO GET PROD. CENTROID 'K', ATTRACT. CENTROID 'J'; AND TRIP VOLUME BETWEEN 'K' AND 'J'
140
WRITE SWITCHED TRIPS FOR PROD. CENTROID '1' ON UNIT B[20]
END FILE UNIT B[20] REWIND UNITS B[20], L[2], AND 10
RETURN


SUBROUTINE
TEST
STORE THE VALUE OF 1.63 DIVIDED BY SQUARE ROOT OF TOTAL NUMBER OF TRIPS IN 'CV1'
STORE THE VALUE OF 1.36 DIVIDED BY THE SQUARE ROOT OF TOTAL NUMBER OF TRIPS IN 'CV5'
ZERO OUT VARIABLES TO BE USED AND SET VARIABLE 'LF' = 1
DO 100 I = 1, NF (NF = MAXIMUM INT. SEP.)
SUM UP NUMBER OF DESIRED TRIPS AT SEP. 'I' AND STORE IN 'CL'
SUM UP NUMBER OF MODEL TRIPS AT SEP. 'I' AND STORE IN 'CM'
STORE NUMBER OF MODEL TRIPS AT SEP. 'I' IN 'FL(1)'
IS DIFFERENCE BETWEEN 'CL' AND 'CM' > VARIABLE 'B'?
YES
SET VARIABLE 'B' = DIFFERENCE BETWEEN 'CL' AND 'CM'
NO
100
SET 'D' EQUAL TO THE TOTAL DIFFERENCE IN TRIPS, 'B', DIVIDED BY TOTAL NUMBER OF MODEL TRIPS, 'CM'
PRINT OUT HEADING, 'D', 'CV1', AND 'CV5' FOR ANALYSIS
IS VALUE OF 'D' LESS THAN 'CV5'
YES
SET VARIABLE 'LF' = 2
NO
RETURN


SUBROUTINE
TIME
THIS SUBROUTINE CALLS A SYSTEM SUBROUTINE TO FIND THE PRESENT TIME AND HAVING SAVED THE TIME FROM A PREVIOUS CALL CALCULATES THE TOTAL ELAPSED TIME SINCE THE LAST CHECK AND PRINTS IT OUT
RETURN


SUBROUTINE
UNPACK
CALL NAME TO WRITE THE NAME OF SUBROUTINE BEING EXECUTED, 'UNPACK'
REWIND UNIT 2 AND ZERO OUT ARRAY TRIPS ARE TO BE STORED IN i.e., T
WRITE HEADER RECORD ON UNIT 2 CONTAINING NUMBER OF ZONES AND HEADER CARD
READ ORIGIN ZONE, SUBNET, NUMBER OF WORDS AND WORDS OFF UNIT 21 (CONTAINS PACKED TRIP MATRIX)
DO 120 I=1, N (N=NUMBER OF ZONES)
ZERO OUT ARRAY T WHERE TRIPS ARE TO BE STORED
100
IS ORIGIN ZONE EQUAL TO I?
NO
YES
DO 110 J=1, K K=NUMBER OF WORDS READ
SHIFT WORD TO THE RIGHT 14 BITS TO GET TRIP VOLUME
GET DESTINATION ZONE FROM LOW ORDER 14 BITS IN WORD
STORE TRIP VOLUME IN ARRAY T (DESTINATION ZONE)
110
READ ORIGIN ZONE, SUBNET, NUMBER OF WORDS, AND WORDS OFF UNIT 21 (CONTAINS PACKED TRIP MATRIX)
END=
STORE ARRAY ON UNIT 2
120
END FILE UNIT 2 REWIND UNITS 2 and 21
RETURN


SUBROUTINE
VERIFY
THIS SUBROUTINE BRINGS IN THE PRESENT VALUES FOR THE NUMBER OF CENTROIDS, THE LARGEST INTERNAL SEPARATION, AND THE LARGEST INDEX FOR ANY TWO SECTORS AND COMPARES THESE VALUES WITH THE MAXIMUM ALLOWABLE VALUES FOR THIS PACKAGE. IF ANY ARE OUT OF RANGE, A MESSAGE IS PRINTED TO THAT EFFECT AND THE PROGRAM TERMINATES WITH A STOP CODE OF 7.
RETURN


SUBROUTINE
WRITE
CALL NAME TO PRINT OUT NAME OF SUBROUTINE BEING EXECUTED (i.e.'WRITE')
REWIND UNIT K (K=4) AND READ HEADER RECORD OFF UNIT K (K=4) WHICH CONTAINS SEPARATION MATRIX
DO 120 I = 1, N (N = NUMBER OF ZONES)
WRITE OUT ORIGIN I AND HEADING; READ SEPARATIONS FOR ORIGIN I OFF UNIT K (K=4)
DO 120 L = 1, NF (NF = MAXIMUM SEPARATION)
INITIALIZE INDEX VARIABLE 'M' TO 0
DO 110 J = 1, N (N = NUMBER OF ZONES)
STORE DESTINATION ZONE J IN VARIABLE KT(M+1)
DOES ORIGIN I HAVE ANY SEPARATIONS
YES
INCREMENT INDEX VARIABLE 'M' BY 1 i.e. M = M+1
NO
110
DID ORIGIN I HAVE ANY SEPARATIONS = L?
YES
WRITE OUT TABLE SHOWING SEPARATION 'L' AND ALL DESTINATION ZONES 'KT' AT SEP. 'L' FROM ORIGIN I
NO
120
RETURN

S I G N I F I C A N T V A R I A B L E S

A N D A R R A Y S

SIGNIFICANT VARIABLES

CROSS REFERENCE FOR SIGNIFICANT VARIABLES

SIGNIFICANT ARRAYS

CROSS REFERENCE FOR SIGNIFICANT ARRAYS

PARAMETER REFERENCES

There are 46 parameters in the Texas Trip Distribution Package. Access to these parameters is provided by the MODIFY routine which uses the FORTRAN NAMELIST option. All parameters are contained in a single NAMELIST named VALUES.

Although a large number of parameters are accessible, only a few (if any) will need to be changed by the user under normal operations. The primary reason for the extensive number of accessible parameters is the flexibility provided for unusual applications.

The following table provides a brief description of each of the parameters and its default value. It should be remembered that the default value for any parameter is overridden by the execution of a routine which defines the parameter.

PARAMETER DESCRIPTIONS AND DEFAULT VALUES

Parameter	Default Value	Variable Type	Description
ADD1	3	Integer	The unit number containing the first trip matrix to be summed by the SUM routine
ADD2	17	Integer	The unit number containing the second trip matrix to be summed by the SUM routine
ADD3	18	Integer	The unit number containing the third trip matrix to be summed by the SUM routine
ADD4	19	Integer	The unit number containing the fourth trip matrix to be summed by the SUM routine
ADD5	23		The unit number containing the fifth trip matrix to be summed by the SUM routine
ADDNUM	2	Integer	Number of trip matrices to be summed
ALTTRP	22	Integer	The unit number which contains the altered trip matrix outputed from the ALTER routine
AMOUNT	110000	Integer	Number of bytes in core to be used in sorting
AN	1.0	Real	Number of zones having nonzero attraction volumes
ASSIGN	13	Integer	The unit number which contains the model trip matrix packed in the format for input to the assignment packages
DUMP	TRUE	Logical	When either MODEL or RESTART routine is executed and this variable is true, then necessary arrays and parameters will be saved on the SV data set after the last iteration so that additional iterations may be run, if desired, using the RESTART routine

PARAMETER DESCRIPTIONS AND DEFAULT VALUES
(cont.)

Parameter	Default Value	Variable Type	Description
EXEMPT	FALSE	Logical	If EXEMPT equals TRUE, the interchange constraint is not applied
EXTEND	0	Integer	Factor used to increase the maximum internal separation in the EDIT routine to provide for special separation codes
FUTURE	FALSE	Logical	If FUTURE is true, a future or synthetic distribution is being performed
IMPSEP	15	Integer	The unit number which contains the separation matrix from the IMPOSE routine
INTRIP	21	Integer	The unit number which contains the packed trip matrix outputed from the assignment packages
LIMIT	5	Integer	Number of iterations performed in execution of the MODEL or EXPAND routines
M	0	Integer	Largest internal centroid number
MODTRP	3	Integer	The unit number which contains the trip matrix constructed by MODEL or EXPAND
MR	0	Integer	Largest sector number (includes any default sector)
MS	4	Integer	The unit number which contains the separation matrix being used
MT	2	Integer	The unit number which contains the trip matrix being used
N	0	Integer	Largest external station number
NEGSEP	1	Integer	The unit number which contains the separation matrix from the SET routine
NEWSEP	16	Integer	The unit number which contains a future separation matrix used as input to ALTER

PARAMETER DESCRIPTIONS AND DEFAULT VALUES
(cont.)

Parameter	Default Value	Variable Type	Description
NF	1	Integer	Largest internal separation including the special separation codes (i.e., including EXTEND + 1 special separation codes)
NOWSEP	4	Integer	The unit number which contains the edited separation matrix from the EDIT routine
NOWTRP	2	Integer	Pointer indicating the unit number which contains the trip matrix from BUILD
NR	0	Integer	Number of sector pair combinations
OMIT	FALSE	Logical	When OMIT equals TRUE, the external trips are not included in the data set constructed by the SCREEN routine
ONE	1.0	Real	Factor used in the SUM routine
PLOT	FALSE	Logical	When PLOT equals TRUE, calcomp plots are outputed from the execution of the REFINE routine or the GET routine
PN	1.0	Real	Number of zones having nonzero production volumes
RAWSEP	8	Integer	The unit number which contains the interzonal separations from the assignment package (used as input to EDIT)
RECORD	14	Integer	The unit number which contains the abbreviated trip records (used in conjunction with the SCREEN and BUILD routines)
REPORT	12	Integer	Pointer indicating the unit number which contains the survey trip reports in 104 byte records
RS	26	Integer	The unit number which contains the SV data set which was previously built by either the MODEL or RESTART routine when executed with DUMP = T

PARAMETER DESCRIPTIONS AND DEFAULT VALUES
(cont.)

Parameter	Default Value	Variable Type	Description
SAMPLE	0.125	Real	Nominal sampling rate as a fraction
SIZE	0	Integer	Indicates the number of records sorted
SUMTRP	20	Integer	The unit number which contains the trip matrix constructed by the SUM routine
SWTTRP	24	Integer	The unit number which contains the trip matrix constructed by the SWITCH routine
SV	25	Integer	The unit number on which necessary arrays and parameters are saved when either the MODEL or the RESTART routine is executed with DUMP = T
TV	undefined	Real	Total trip volume for specified purpose(s)
TYPE	1077952576	Integer	Contains a four-byte literal used for identification
UT	0.0	Real	Number of zone pairs having trip interchanges
XP	-0.00068	Real	Exponent for the relative production model

CROSS REFERENCE FOR SIGNIFICANT VARIABLES

The following table is designed to provide a convenient cross reference between twenty-seven of the forty-eight significant variables and the routines in which they are used. The remaining twenty-one variables are those used as pointers to the data sets. The cross reference for the data sets contained in Section V (DATA SET AND DATA CARD FORMATS) can be used to identify the routines in which these pointers are used. The symbols used in the cross reference table are as follows:

Symbol	Meaning
P	PREVIOUSLY DEFINED - The routine assumes the variable has been previously defined.
M	MODIFIED OR CALCULATED - The value of this variable may be either modified or calculated in the indicated routine.
I	INPUT FROM CARDS - The value of this variable is established by card input in the indicated routine.
D	DATA SET I/O - This variable is used as input and/or output in a header record in a data set in the indicated routine.
W	WRITE - The value of this variable is printed and/or punched by the indicated routine.
C	COMMON - The variable is passed to the routine by use of a COMMON statement.

CROSS REFERENCE FOR SIGNIFICANT VARIABLES

VARIABLES	MODIFY	PERUSE	UNPACK	SCREEN	BUILD	PRINT	EDIT	WRITE	EQUATE	SET	LIST	REFINE	ACCEPT	IMPOSE	MODEL	EXPAND	GET	MATCH	SUM	SWITCH	ALTER	PACK	RESTART
ADDNUM	I	W										W											
AMOUNT	I	W		PC								W						PC		PC			
AN	I	W								M		PCW	PCM				M						
DUMP															P								P
EXEMPT	I	W										PW	P		P								P
EXTEND	I	W					P					W											
FUTURE	I	W										WM	P		P								P
LIMIT	I	W										W			P	P							P
M	I	W	D		D		MD			D		WP	D		P		D						P
MR	I	W							M	M		W					M*						
N	I	W	D	M	D	D	MD	D	P	D	D	PW	D	D	P	P	D	D	D	D	D	D	P
NF	I	W					MD	D		D	D	PW	D	D	P		D	D					P
NR	I	W							M	M		PW	P		P		M*	P					P
OMIT	I	W		P								W											
ONE	I	W										W							P				
PLOT	I	W										WPM					P						
PN	I	W								M		PCW	PCM				M						
SAMPLE	I	W								M		PCW					M						
SIZE	I	W		CM								W						CM		CM			
TV	I	W								M		PW	PM		P		M						P
UT	I	W								M		MCW	M				M						
XA	I	W										MCW	PC										
XF	I	W										MCW	PC								PC		
XL	I	W										MCW	PC										
XP	I	W										MCW	PC										
TYPE	I	W	D		D							PW											

* Conditional

SIGNIFICANT ARRAYS

There are 24 arrays used in a package which may be considered significant arrays. The contents of these arrays are not directly accessible to the user of the package. In order to conserve core storage, many of these arrays are used at one time or another as scratch arrays by some of the routines. In these instances, the user is cautioned in the routine's description (under the heading "Execution Requirements") that it may destroy key arrays.

The following table describes the usual contents of each of these 24 arrays. Since the array names are changed in some of the subroutines, the names used in the table are those used in the subroutine DIRECT. Since most of these arrays are variable dimensioned, the dimensions specified in the table are given in terms of the parameters which determine their size.

GENERAL CONTENTS OF

SIGNIFICANT ARRAYS

Array Name	Dimension	Usual Contents
A	N	Attractions for each centroid and external station.
AM	N	Modeled attractions for each centroid and external station.
AR	N	Relative attractions for each centroid and external station.
AZ	N	Number of non-zero attraction interactions for each centroid and external station.
D	N	Scratch array.
F	NF + 3	Number of trips at each separation (including external trips).
FL	NF + 3	Number of modeled trips at each separation (including external trips) from previous iterations.
FM	NF + 3	Number of modeled trips at each separation (including external trips).
KE	N	Sector equivalences.
KS	N	Scratch array.
P	N	Productions for each centroid and external station.
PM	N	Modeled productions for each centroid and external station.
PZ	N	Number of non-zero production interactions for each centroid and external station. In MODEL it is the Accessibility Limit Value.
RC	NR	Number of possible zone pair combinations between sectors.
RE	NR	Number of zone pairs between sectors with non-zero interactions.

GENERAL CONTENTS OF

SIGNIFICANT ARRAYS (Continued)

Array Name	Dimension	Usual Contents
RM	NR	Sum of the squares of trips between sectors.
RR	NR	Bias factors.
RV	NR	Sector interchange volumes.
S	NF	Number of zone pairs at each separation.
SM	NF	Scratch array.
T	N	Scratch array (usually for trip interchanges).
U	NF	Number of non-zero interactions at each separation.
UM		Estimated number of non-zero interactions at each separation.
W	NF	Relative number of trips at each separation.

CROSS REFERENCE FOR SIGNIFICANT ARRAYS

The following table is designed to provide a convenient cross reference between the significant arrays and the routines in which they are used. The symbols used in the table are as follows:

Symbol	Meaning
D	DEFINED - The routine establishes the contents of the array which may be passed to other routines.
P	PREVIOUSLY DEFINED - The routine assumes the contents of the array has been previously defined.
S	SCRATCH - The routine ignores the current contents of the array and puts information into the array which is not intended for use by subsequent routines.
M	MODIFIED - The routine may modify the contents of the array.
I	INPUT - The routine inputs the contents of this array from either cards or external storage.
T	TEMPORARY - The routine uses the array to temporarily store parts of a data set.

CROSS REFERENCE FOR SIGNIFICANT ARRAYS

ARRAYS	MODIFY	PERUSE	UNPACK	SCREEN	BUILD	PRINT	EDIT	WRITE	EQUATE	SET	LIST	REFINE	ACCEPT	IMPOST	MODEL	EXPAND	GET	MATCH	SUM	SWITCH	ALTER	PACK	RESTART
A										D		P	DI		P	S		S					S
AM															S	S	S	S					S
AR												D	D		PM	PM	S						SM
AZ							S			D		P			S		S	S					S
D							S			S							S	T	T		T		
F										D		P	D		P		P						S
FL												S			S								S
FM											S	S	S		S		S						S
KE									D	D					P		D						S
KS	S		S	S	S	T	T	T		T	T	S		T	T		T	T			T	S	T
P										D		P	DI		P	P							S
PM										D		P					S						
PZ							S			D		P			S		S	S					S
RC										S							S						
RE										S					S		S						S
RM										D					P		S						S
RR												D	D		PM*								SM*
RV										D					P*		S						S*
S							D			D		P	P		S		S	S					S
SM											S	D	D		P			S					S
T			T		T			S	S	T	T	S			T	T	T	T	T	T	T	T	T
U										D		P						S					
UM											S	D	D		PM		S	S					SM
W												DM	D		PM			S					SM

* Conditional

D A T A S E T A N D D A T A C A R D F O R M A T S

DATA SET FORMATS

TRIP MATRIX DATA SETS

Assignment Data Sets (ASSIGN and INTRP)

NOWTRP Data Set

Other Trip Matrix Data Sets (ADD1, ADD2, ADD3, ADD4, ADD5, ALTTRP, MODTRP, SUMTRP, SWTTRP)

SEPARATION MATRIX DATA SETS

RAWSEP Data Set

Other Separation Matrices (IMPSEP, NEGSEP, NEWSEP, NOWSEP)

OTHER DATA SETS

REPORT Data Set

RECORD Data Set

SAVE Data Set

DATA SET SPECIFICATIONS

DATA CARD FORMATS

ADMIT
BIAS
CATEGORY
CONTROL
EQUALS
FORMAT
GENERATION
HEADING
INTERACTION
LENGTH
SEPARATION
& VALUES

CROSS REFERENCE TABLE

DATA SET FORMATS

Twenty-four data sets are associated with the Texas Trip Distribution Package. It is doubtful that all of these data sets would ever be used in any given application of the package. The data sets needed are determined by the routines to be executed and the options (if any) selected by the user under the various routines. The data set requirements for each routine are listed under the heading "Data Set References" in the descriptions of each individual routine.

The data sets associated can be grouped into five classes as follows:

- Matrix--those data sets which contain either a trip matrix or a separation matrix.
- Sort--those data sets used by the sort routine
- Assignment--those data sets created by or to be used as input to either the Texas Large Network Package or the Texas Small Network Package.
- Trip Report--the data set containing the original trip reports used as input to the package (record size = 104 bytes)
- Save data set used with RESTART.

The data sets are listed in the following tables along with their default unit numbers, classifications and brief descriptions of their contents.

DATA SET DESCRIPTIONS, CLASSIFICATIONS,

AND DEFAULT UNIT NUMBERS

Data Set Name	Default Unit Number	Class	Contents
ADD1	3	matrix	First trip matrix to be summed by the SUM routine
ADD2	17	matrix	Second trip matrix to be summed by the SUM routine
ADD3	18	matrix	Third trip matrix to be summed by the SUM routine
ADD4	19	matrix	Fourth trip matrix to be summed by the SUM routine
ADD5	23	matrix	Fifth trip matrix to be summed by the SUM routine.
ALTTRP	22	matrix	The altered trip matrix outputed from the ALTER routine.
ASSIGN	13	assignment	The model trip matrix packed in the format for input to the assignment packages.
IMPSEP	15	matrix	The separation matrix from the IMPOSE routine
INTRIP	21	assignment	The packed trip matrix outputed from the assignment packages.
MODTRP	3	matrix	The trip matrix constructed by Model or Expand
NEGSEP	1	matrix	The separation matrix from the SET routine.
NEWSEP	16	matrix	The future separation matrix used as input to ALTER
NOWSEP	4	matrix	The edited separation matrix from the EDIT routine.

DATA SET DESCRIPTIONS, CLASSIFICATIONS,

AND DEFAULT UNIT NUMBERS
(continued)

Data Set Name	Default Unit Number	Class	Contents
NOWTRP	2	matrix	The trip matrix outputed from BUILD
RAWSEP	8	assignment	The interzonal separations from the assignment package (used as input to EDIT)
RECORD	14	sort	The abbreviated trip records (used in conjunction with the SCREEN and BUILD routines)
REPORT	12	trip report	The survey trip reports in 104 byte records.
RS	26	save	Parameters and arrays used by RESTART
SORTIN	9	sort	Records to be sorted by system sort.
SUMTRP	20	matrix	The trip matrix constructed by the SUM routine.
SV	25	save	Parameters and arrays written by MODEL or RESTART.
SWTTRP	24	matrix	The trip matrix constructed by the SWITCH routine.
PLOTTAPE	--	--	The calcomp plots for input to the calcomp plotter.
SORTOUT	10	sort	The sorted records outputed from the system sort.

TRIP MATRIX DATA SETS

Twelve of the data sets associated with the package are trip matrix data sets. Two of these are assignment data sets (INTRP and ASSIGN) which are in the format used by the Texas Large and Small Network Packages. The remaining ten data sets use a simpler and more convenient format. Except for the information stored in the header record of the NOWTRP data set, the format for each of these ten data sets are identical. The following are the formats for the twelve trip matrix data sets:

Assignment Data Sets
(ASSIGN and INTRP)

Header Record

Displacement In Bytes	Length In Bytes	Contents
0	4	Number of Subnetworks
4	4	First centroid number
8	4	Last centroid (or external station) number

Trip Record

Displacement Bytes	Length Bytes	Contents
0	4	Origin zone of all interchanges in this record
4	4	Subnet of the origin zone (= 1)
8	4	N-Number of interchanges in this record (from 1 to 100)
12	4	Interchange item
8+4N	4	Interchange item

The interchange item is an 18 bit interchange volume followed by a 14-bit destination zone number.

The trip records are in sort on the origin zone and the interchange items for each origin are in sort on the destination zone.

NOWTRP Data Set

The following is the format used in the NOWTRP data set. There are a total of N + 1 records, each N + 1 words in length. The first record is a header record. Following it there are N trip records. Each trip record K contains the number of trips from centroid K to each centroid and external station in the network.

FORMAT

HEADER RECORD

Displacement In Words	Data Type	Length In Words	Contents
0	Integer	1	Last external station number (= N)
1	Integer	1	Zero
2	Integer	1	Last centroid number (= M)
3	Integer	1	Four byte literal identifying the category of trips contained on this data set
4	Literal	7	Blank
11	Literal	10	Literal heading describing trips contained on this data set.
21	Literal	3	Data that this data set was built
24	Literal	1	The literal word 'CATE'
25	-	N-25	Disregarded

TRIP RECORD K (K = 1, 2, ..., N)

Displacement In Words	Data Type	Length In Words	Contents
0	Real	1	Trip Volume from centroid K to centroid 1
1	Real	1	Trip Volume from centroid K to centroid 2
⋮	⋮	⋮	⋮
N-1	Real	1	Trip volume from centroid K to external station N
N	Real	1	Number of trip reports for centroid K

Other Trip Matrix Data Sets (ADD1, ADD2, ADD3, ADD4, ADD5, ALTTRP MODTRP, SUMTRP, and SWTTRP)

The following is the format used in the data sets ADD1, ADD2, ADD3, ADD4, ADD5, MODTRP, ALTTRP, SUMTRP, and SWTTRP. There are a total of N + 1 records, each N works in length. The first record is a header record. Following it, there are N trip records (one trip record for each centroid and external station). Each trip record contains the number of trips from that centroid or external station to every centroid and external station in the network.

FORMAT

HEADER RECORD

Displacement In Words	Data Type	Length In Words	Contents
0	Integer	1	Last external station number (= N)
1	Integer	1	Maximum internal separation (= NF)
2	Integer	1	Last centroid number (= M)
3	Literal	18	Heading obtained from Heading card
21	Literal	3	Date that this data set was built
24	Literal	N-24	Array containing number of zone pairs at each separation

TRIP RECORD K (K = 1, 2, ..., N)

Displacement In Words	Data Type	Length In Words	Contents
0	Real	1	Trip Volume from centroid 1
1	Real	1	Trip volume from centroid K to centroid 2
⋮	⋮	⋮	⋮
N-1	Real	1	Trip volume from centroid K to external station N

SEPARATION MATRIX DATA SETS

Five of the data sets associated with the package are separation matrix data sets. One of these is an assignment data set (RAWSEP) which is in the format used by the Texas Large and Small Network Packages. The format for the remaining four data sets differs only slightly (primarily in the header record) from the RAWSEP data set format. The major difference, of course, is that the separations in the RAWSEP data set are in units of 0.01 minutes while the separations in the other data sets are in units of 1.0 minutes. The following are the formats for thses five separation matrix data sets:

RAWSEP Data Set

The following is the format used in the data set RAWSEP. There are a total of N + 1 records, each N words in length. The first record is a header record. Following it, there are N separation records. Each separation record K contains the separations between centroid K and each centroid and external station in the network.

FORMAT

HEADER RECORD

Displacement In Words	Data Type	Length In Words	Contents
0	Integer	1	Last external station number (= N)
1	Integer	1	Zero
⋮	⋮	⋮	⋮
N-1	Integer	1	Zero

SEPARATION RECORD K (K = 1, N)

Displacement In Words	Data Type	Length In Words	Contents
0	Integer	1	Separation* between centroid K and centroid 1
1	Integer	1	Separation between centroid K and centroid 2
⋮	⋮	⋮	⋮
N-1	Integer	1	Separation between centroid K and external station N

*Separation values are in units of 0.01 minutes

Other Separation Matrix Data Sets (IMPSEP, NEGSEP, NEWSEP, and NOWSEP)

The following is the format used in the data sets NOWSEP, NEWSEP, NEGSEP, IMPSEP. There are a total of N + 1 records, each N works in length. The first record is a header record. Following it there are N separation records. Each separation record K contains the separations between centroid K and each centroid and external station in the network.

FORMAT

HEADER RECORD

Displacement In Words	Data Type	Length In Words	Contents
0	Integer	1	Last external station number (= N)
1	Integer	1	Maximum internal separation (= NF)
2	Integer	1	Last centroid number (= M)
3	Literal	18	Heading obtained from Heading card
21	Literal	3	Date that this data set was built
24	Real	N-24	Array containing number of zone pairs at each separation.

SEPARATION RECORD K (K = 1, 2, ..., N)

Displacement In Words	Data Type	Length In Words	Contents
0	Integer	1	Separation between centroid K and centroid 1
1	Integer	1	Separation between centroid K and centroid 2
⋮	⋮	⋮	⋮
N-1	Integer	1	Separation between centroid K and external station N.

Save Data Set Format
(SV and RS)

The following is the format of the data set written by MODEL and RESTART for input to RESTART to restart the MODEL program. The data set is written in two records.

FORMAT

PARAMETER RECORD

Displacement In Words	Data Type	Length In Words	Contents (or Parameter name)
0	Real	1	AN
1	Logical	1	EXEMPT
2	Integer	1	EXTEND
3	Logical	1	FUTURE
4	Integer	1	Iteration number represented by this data set
5	Integer	1	M
6	Integer	1	MR
7	Integer	1	N
8	Integer	1	NF
9	Integer	1	NR
10	Logical	1	OMIT
11	Real	1	ONE
12	Real	1	PN
13	Real	1	SAMPLE
14	Real	1	TV
15	Real	1	UT
16	Integer	1	The number of data points in the production interaction curve
17	Real	53	The production volumes of the data points in the production interaction curve.

Displacement In Words	Data Type	Length In Words	Contents (or Parameter name)
70	Real	53	The percent interactions divided by 100 for the data points in the production interaction curve.
123	Real	1	XP

ARRAY RECORD

Displacement In Words	Data Type	Length In Words	Contents (Significant Array Names)
0	Real	N	P
N	Real	N	PM
2N	Real	N	The accessibility eliminator limits.
3N	Real	N	A
4N	Real	N	AM
5N	Real	N	AZ
6N	Real	N	AR
7N	Real	N	D
8N	Real	N	T
9N	Real	N	KS
10N	Real	N	KE
11N	Real	NF+3	W
11N+NF+3	Real	NF+3	F
11N+2NF+6	Real	NF+3	FM
11N+3NF+9	Real	NF+3	FL
11N+4NF+12	Real	NF+3	S
11N+5NF+15	Real	NF+3	SM
11N+6NF+18	Real	NF+3	U
11N+7NF+21	Real	NF+3	UM
11N+8NF+24	Real	NR	RV
11N+8NF+24+NR	Real	NR	RM
11N+8NF+24+2NR	Real	NR	RC
11N+8NF+24+3NR	Real	NR	RE
11N+8NF+24+4NR	Real	NR	RR

OTHER DATA SETS

Two other data sets will be of interest to the data processing programmer. They are the REPORT data set and the RECORD data set. Their formats are as follows:

REPORT Data Set

The format of the REPORT data set containing the trip reports may be of concern to the user since it is produced externally. The format for this data set is, of course, compatible with the format currently being used by the Texas Highway Department in their urban transportation studies.

The REPORT data set consists of 104 byte records and up to six types of records. The six types of 104 byte are:

- Type 1 records contain information regarding the dwelling unit in which the home interview was conducted and are identified by an integer 1 in the first byte of the record.
- Type 2 records contain information regarding each trip reported in the home interviews and are identified by an integer 2 in the first byte of the record.
- Type 3 records contain the external trip report information obtained at external stations and are identified by an integer 3 in the first byte of the record.
- Type 4 records contain truck trip information and are identified by an integer 4 in the first byte of the record.
- Type 5 records contain taxi trip information and are identified by an integer 5 in the first byte of the record.
- Type 6 records contain employment information and are identified by an integer 6 in the first byte of the record.

Type 1 and type 6 records are ignored by the trip distribution package. Only selected information is used from the remaining records. The formats for these records (i.e. types 2-5) are contained on the following pages. These formats only specify the contents of fields which are used by the trip distribution package; all other fields are disregarded.

Type 2 Records

Byte	Content
1	Report type (Integer 2)
2-26	Disregarded
27-28	Origin location code (blank if inside study area)
29-40	Disregarded
41-42	Destination location code (blank if inside study area)
43-54	Disregarded
55	Travel mode
56-59	Disregarded
60	Origin primary trip purpose
61	Destination primary trip purpose
62	Origin secondary trip purpose
63	Destination secondary trip purpose
64-77	Disregarded
78-80	Trip volume (decimal point assumed between bytes 79 and 80)
81-94	Disregarded
95-99	Origin centroid
100-104	Destination centroid

Type 3 Records

Byte	Content
1	Report type (Integer 3)
2-20	Disregarded
21	Vehicle occupancy
22-26	Disregarded
27-28	Origin location code (blank if inside study area)
29-40	Disregarded
41-42	Destination location code (blank if inside study area)
43-54	Disregarded
55	Travel mode
56-64	Disregarded
65	External origin purpose
66	External destination purpose
67-77	Disregarded
78-80	Trip volume (decimal point assumed between bytes 79 and 80)
81-94	Disregarded
95-99	Origin centroid
100-104	Destination centroid

Type 4 Records

Byte	Content
1	Report type (Integer 4)
2-26	Disregarded
27-28	Origin location code (blank if inside study area)
29-40	Disregarded
41-42	Destination location code (blank if inside study area)
43-54	Disregarded
55	Travel mode
56-77	Disregarded
78-80	Trip volume (decimal point assumed between bytes 79 and 80)
81-94	Disregarded
95-99	Origin centroid
100-104	Destination centroid

Type 5 Records

Byte	Content
1	Report type (Integer 5)
2-26	Disregarded
27-28	Origin location code (blank if inside study area)
29-40	Disregarded
41-42	Destination location code (blank if inside study area)
43-54	Disregarded
55	Travel mode
56-77	Disregarded
78-80	Trip volume (decimal point assumed between bytes 79 and 80)
81-94	Disregarded
95-99	Origin centroid
100-104	Destination centroid

RECORD Data Set

The following is the format used in the data set RECORD. The number of records in this data set may vary, depending upon the number of trip reports in the data set REPORT. For example, if there were L trip reports in REPORT, the maximum number of records in RECORD would be (2 X L) + 1, each five words in length. The first record is a header record.

FORMAT

HEADER RECORD

Displacement In Words	Data Type	Length In Words	Contents
0	Integer	1	Zero
1	Integer	1	Last external station number (= N)
2	Integer	1	Last centroid number (= M)
3	Integer	1	Zero
4	Integer	1	Zero

TRIP RECORD

Displacement In Words	Data Type	Length In Words	Contents
0	Integer	1	Centroid number from which trips originate.
1	Integer	1	Centroid number for which trips are destined.
2	Integer	1	Code number for trip category
3	Real	1	Code number which contains trip mode and trip volume.
4	Real	1	Number of passengers carried

Data Set Specifications

In order to operate the Texas Trip Distribution Package, it will be necessary to provide appropriate specifications for each data set involved with each routine being executed. Sample specifications are provided in the following table. All parameters refer to Job Control Language (JCL) Data Definitions (DD). The appropriate IBM manual should be consulted for further description of the requirements.

Data Set Class	REDFM	LRECL	BLKSIZE
Matrix	VBS	6000	6004
Sort	VBS	24	6004
Assignment	VBS	416	792
Trip Report	FB	104	1040

DATA CARD FORMATS

There are 12 types of data cards associated with the Texas Trip Distribution Package. Each type contains a literal identification field. Nine of the card types are either used as input for specific routines or are outputted from certain routines for later use. The other two card types are used either to specify the routines to be executed (i.e. the CONTROL card) or to specify the heading to be used on printed output (i.e. the HEADING card).

Each of the following data card descriptions have been divided into five sections. The sections describe the card's purpose, the routines directly associated with the card type, the entry sequence required, the card layout, and a description of the data contained in the card.

ADMIT

Purpose

The ADMIT cards enter individual movements which are to be imposed in the trip distribution.

Associated Routines

Input

IMPOSE

Entry Sequence

All ADMIT cards must be in numerically increasing sequence of production (or origin) zone numbers. Attraction (or destination) zone numbers, with respect to each production zone, may be in any order.

Card Layout (fixed): FORMAT (A4, 1X, 1515)

Columns	Type	Content
1 - 5	Literal	'ADMIT' or 'LOCAL'
6 - 10	Integer	Production (or origin) zone number
11 - 15	Integer	Attraction (or destination) zone number
.	.	. (consecutive fields of five columns)
.	.	.
.	.	.
76 - 80	Integer	Attraction (or destination) zone number

Data Description

Each data card must contain the production zone number in columns

6-10. The remainder of the data card is interpreted as fourteen five-column fields which are provided for attraction zone numbers. Blank fields are disregarded, attraction zone numbers exceeding the last valid attraction zone are indicated with an error message and disregarded, and all fourteen fields are examined for valid entries. All data must be right-justified in the fields.

As many cards as are necessary may be supplied for each production zone. Consecutive attraction zone numbers appearing in strings may be entered through a shorthand notation by coding only the first and last zone number in the string, with the last number preceeded by a minus sign. The shorthand coding may not span to the next card and the first attraction zone number appearing on any card must not be preceeded by a minus sign.

All entries are error checked. Duplicate entries are ignored.

BIAS

Purpose

The BIAS cards enter correction factors which compensate for biases in travel movements between sector pairs.

Associated Routines

Input	Output
ACCEPT	MODEL

Entry Sequence

The BIAS cards immediately follow the FORMAT card which describes their format. BIAS cards may be in any order.

Card Layout (variable); sample FORMAT (A4, 6X, 2I5, F10.3)

Field	Type	Content
1	Literal	'BIAS'
2	Integer	Sector number or movement code
3	Integer	Sector number or blank
4	Real	Bias correction factor

Data Description

BIAS cards are interpreted by a variable format as supplied by a FORMAT card. The word BIAS should appear as the first item on every BIAS card. The second and third items on each BIAS card should index the appropriate movement, and the bias correction factor should be punched as the fourth item. BIAS cards are not required if the correction factor is 1.0. Movements not entered on BIAS cards are assumed not to require bias correction.

Movements may be identified on BIAS cards by either one of two schemes. The two related sector numbers may be entered. The other method is to enter the actual movement index. The two methods are distinguished on the data cards by whether or not the field corresponding to the second sector number is blank. If the field is blank, the other field is interpreted to contain the actual movement index. If a nonzero value is encountered, the two sector numbers are assumed to be provided. If two or more entries for any movement are encountered, the last entry will be retained and no message is written to notify the user of this potential error in data preparation.

CATEGORY

Purpose

The CATEGORY card enters criteria for selecting trips of a desired category from survey data trip records.

Associated Routines

Input

BUILD

Entry Sequence

Not applicable

Card Layout (fixed); FORMAT (A4, 4X, A4, 1412, 10A4)

Columns	Type	Content
1 - 8	Literal	'CATEGORY'
9 - 12	Literal	Literal identification (e.g., HBW, HBNW, NHB, TRTX)
13 - 14	Integer	Category code
. . .	. . .	. . . (consecutive fields of two columns)
39 - 40	Integer	Category code
41 - 80	Literal	Literal description used in table headings

Data Description

The word CATEGORY should appear in the first eight columns of the CATEGORY card. Columns nine through twelve should be left blank or

coded with a four-byte literal which will serve as an abbreviation for the category being specified. This four-byte literal will be inserted as the TYPE parameter. The appropriate FORMAT for the GENERATION cards for this cateogry is identified by the parameter TYPE.

Category codes are punched in columns thirteen through forty in two-digit fields. These category codes determine the type of trip matrix which is to be prepared. The category codes are defined in an accompanying table. All codes have two digits; there are no codes less than ten or greater than forty. At least one code between the values of ten and thirty, inclusively, must be specified. This will select some form of vehicle trips. Indicator codes are provided for selecting other combinations.

If the CMIT parameter was set .TRUE. during the execution of SCREEN, only internal trips will be available and indicators 37, 38 and 39 will have no significance. However, if external trips are included, indicators 37, 38 and 39 should be coded as desired. If none of these three codes are entered, indicators 37 and 38 will be entered by default.

Columns 41 through 80 of the CATEGORY card are used as a literal description of the trip matrix being prepared. This will be entered in the parameter record of the trip matrix, and will subsequently appear in printed page headings.

If indicators 31-36 and the associated purposes (codes 10-30) are coded, only passenger trips will be selected. In order to obtain person trips, indicator 40 must be coded, also.

CATEGORY CODES

Code (Indicator)	Code Origin-Destination	Code Production Attraction	Description	Type Available	Mode
	10		All nonhome-based trips	Internal & External	Auto
	11	21	Home-based work trips	Internal & External	Auto
	12	22	Home-based personal business trips	Internal & External	Auto
	13	23	Home-based medical-dental trips	Internal & External	Auto
	14	24	Home-based school trips	Internal & External	Auto
	15	25	Home-based social-recreational trips	Internal & External	Auto
	16	26	Home-based change travel mode trips	Internal & External	Auto
	17	27	Home-based eat meal trips	Internal & External	Auto
	18	28	Home-based shop trips	Internal & External	Auto
	19	29	Home-based serve passenger trips	Internal & External	Auto
	20		Truck trips	Internal & External	Truck
	30		Taxi trips	Internal only	Taxi
31			Select walk trips (to work only)	Internal only	Person
32			Select auto passenger trips	Internal & External	
33			Select bus passenger trips	Internal only	Passenger
34			Select taxi passenger trips	Internal only	Passenger
35			Select truck passenger trips	Internal & External	Passenger
36			Select school bus passenger trips	Internal only	Passenger
37			Select Internal-Internal trips		
38			Select Internal-External trips		
39			Select External-External trips		
40			Select person trips		

Purpose

The CONTROL cards specify the routines to be executed and their sequence.

Associated Routines

NONE

Entry Sequence

The CONTROL cards must appear first in the input data stream. Only HEADING cards may be intermingled.

Card Layout (fixed): FORMAT (A4, 4X, 18A4)

Columns	Type	Contents
1 - 7	Literal	'CONTROL'
8		blank
9-80	Literal	Routine names separated by commas

Data Description

The word CONTROL must appear in the first seven columns of each and every control card. The eighth column should be blank. The remaining columns, nine through eighty, contain the literal names of the routines which are to be executed. The names should be separated by commas. No CONTROL entry name should contain any embedded blanks. However, blanks are fully acceptable between control entry names. The last entry on a control card may be followed by a period to terminate the card scan.

Multiple control cards may be used when the number of routines extend beyond one card. There is no limit to the number of CONTROL cards which may be used, except that only the first 40 entries will be executed. It is not permissible to split a control entry name and put a portion of the name on a following control card. Any invalid control entry will cause immediate program termination.

Purpose

The EQUALS cards enter equivalences between centroids and sectors.

Associated Routines

Input

EQUATE

Entry Sequence

The EQUALS cards may be in any sequence.

Card Layout (fixed): FORMAT (I3, 1X, A4, 2X, 14I5)

Columns	Type	Content
1 - 3	Integer	Sector number
4	-	blank
5 - 10	Literal	'EQUALS'
11 - 15	Integer	Centroid number
.	.	.
.	.	. (consecutive fields of five columns)
.	.	.
76 - 80	Integer	Centroid number

Data Description

A sector number should be punched right justified in columns one to three of every EQUALS card. Its value should range between one and the largest sector number. Column four should be blank. Columns five

through ten should contain the word EQUALS. Fourteen five-digit fields constitute the remainder of each EQUALS card. These should contain valid centroid numbers right justified in each field.

EQUALS cards may be in any order. Within any EQUALS card, centroid numbers may appear in any order. Any of the 14 centroid number fields may be left blank. It is advantageous not to skip any sector numbers when preparing EQUALS cards.

FORMAT

Purpose

The FORMAT card enters the format by which GENERATION, LENGTH, or BIAS cards are to be read.

Associated Routines

Input	Output
ACCEPT	REFINE
	MODEL

Entry Sequence

The FORMAT card precedes the first GENERATION, LENGTH, or BIAS cards to which it pertains.

Card Layout (fixed): FORMAT (A4, 2X, 17A4, A2, A4)

Columns	Type	Content
1 - 6	Literal	'FORMAT'
7 - 76	-	Variable format enclosed in parentheses
77 - 80	Literal	Literal identification

Data Description

The word FORMAT should appear in the first six columns of any FORMAT card. The remainder of the card may be used for format coding. The format should be enclosed in parentheses. Columns 77 through 80 may be used for a four-byte literal identification which will be compared with the parameter type.

It might be pointed out that the T-format code may prove very useful in coding variable formats when the data is not in the same sequence as required by the read statement. If questions arise regarding proper format coding, the appropriate FORTRAN reference manual should be consulted.

GENERATION

Purpose

The GENERATION cards enter production and attraction (or origin and destination) volumes for each centroid.

Associated Routines

Input

ACCEPT

Entry Sequence

The GENERATION cards immediately follow a FORMAT card specifying their format. The GENERATION cards may be in any order.

Card Layout (variable): sample (FORMAT (A4, 6X, 15, T26, 2F5.0)

Field	Type	Content
1	Literal	'GENERATION' or 'FORECAST'
2	Integer	Centroid number
3	Real	Production (or origin) volume
4	Real	Attraction (or destination) volume

Data Description

GENERATION cards are read in a variable format depending upon what is supplied by the preceding FORMAT card. Four items are read from each card. The first item is the word generation or forecast. The second item is the centroid number which should appear as an integer. The third item is the production or origin volume and the fourth item is the attraction or destination volume. These two volumes are read as real variables.

HEADING

Purpose

The HEADING card enters a literal message which is used for identification of printed output.

Associated Routines

NONE

Entry Sequence

Any encounter of a HEADING card in the data card input stream, between routine executions, results in a heading change.

Card Layout (fixed); FORMAT (A4, 4X, 18A4)

Columns	Type	Contents
1 - 7	Literal	'HEADING'
8	-	blank
9 - 80	Literal	Literal heading

Data Description

The word HEADING must appear in the first seven columns of every HEADING card. The eighth column should be blank. The remainder of the card, columns nine through eighty, may contain any literal information desired to be used as a heading.

INTERACTION

Purpose

The INTERACTION cards enter the points which describe a production interaction curve.

Associated Routines

Input

ACCEPT

REFINE

Entry Sequence

The INTERACTION cards immediately follow a FORMAT card specifying their format. These cards should follow the GENERATION cards, LENGTH cards, and BIAS cards. The INTERACTION cards should be in ascending order on production volumes.

Card Layout (variable): sample (FORMAT (A4, 7X, 2F8.0))

Field	Type	Content
1	Lateral	'INTE' (the 'RACTION' is ignored)
2	Real	production volume
3	Real	the maximum percent of internal zones with which a production zone of this size would be expected to interact. (For example, 50% would be entered as 50.0)

Data Description

The INTERACTION cards are read in a variable format as presented by the preceding FORMAT card. The first field of the card should be a four-column field containing the letters "INTE" (i.e. the first four letters

of the word INTERACTION). The second field should contain a production volume. The third field should contain the maximum percent of the internal zones with which a zone having the production volume specified in the second field would be expected to interact.

In essence, the INTERACTION cards are used to describe the production-interaction curve which will be used by the MODEL routine to constrain the number of interactions. Each card represents a point on the production-interaction curve. Straight line interpolation will be used to determine any needed points between two points specified by INTERACTION cards. If points are needed beyond the last point specified by INTERACTION cards, straight line extrapolation will be used to determine the points based on the last two points specified by INTERACTION cards. If the production-interaction curve specifies a percent of the internal zones which is larger than the percent of internal zones with a non-zero attraction volume, then the percent specified by the curve will be ignored and the percent of internal zones with a non-zero attraction volume will be used instead.

Purpose

The LENGTH cards enter relative trip volumes corresponding to each value of trip length.

Associated Routines

Input	Output
ACCEPT	REFINE

Entry Sequence

The LENGTH cards immediately follow the FORMAT card which describes their FORMAT, if a FORMAT card is included. The LENGTH cards may be in any order.

Card Layout (variable): sample FORMAT(A4, 6X, I5, 5X, F10.3, 10X, A4)

Field	Type	Content
1	Literal	'LENGTH'
2	Integer	Separation (length) value
3	Real	Trip frequency volume
4	Literal	blank, 'INT-EXT', 'EXT-INT', or 'THRU'

Data Description

The LENGTH cards are read in a variable format as presented by the preceding FORMAT card. It should be noted that if no FORMAT card is presented after the GENERATION cards and before the LENGTH cards, the format used for the GENERATION cards will be assumed. This will

cause errors since the data is incompatible. The word length should appear as the first item on each LENGTH card. The next item should be a separation value. The following item should be the frequency volume. The last item should be a literal field which is normally left blank. However, if the separation values used for trips with external terminals are not known, the expressions INT-EXT, EXT-INT, or THRU may be supplied in the fourth field to identify these movements and the first field left blank. If literal codes are used then any numerical separation value contained in the first field is ignored.

SEPARATION

Purpose

The SEPARATION cards enter replacement values for separations between centroid pairs.

Associated Routines

Input

EDIT

Entry Sequence

All SEPARATION cards must be in numerically increasing sequence with regard to production (or origin) zone numbers. Attraction (or destination) zone numbers, corresponding to each production zone, may be in any order.

Card Layout (fixed): FORMAT (A4, 6X, 3I5)

Columns	Type	Contents
1 - 10	Literal	'SEPARATION'
11 - 15	Integer	Production (or origin) centroid number
16 - 20	Integer	Attraction (or destination centroid number
21 - 25	Integer	Separation value

Data Description

The word SEPARATION should appear in columns one through ten on each SEPARATION card. Columns eleven through fifteen, and sixteen through twenty represent the fields for the production and attraction centroid numbers, respectively. These numbers must be right justified in these

fields, and must range between one and the numerical value of the last centroid. Columns 21 through 25 define the field for the replacement separation value. This value must be an integer, right justified in the field, and range between one and the largest separation value as augmented by the parameter EXTEND.

&VALUES

Purpose

The &VALUES card enters changes in parameter values.

Associated Routines

Input	Output
MODIFY	REFINE

Entry Sequence

Random

Card Layout (fixed): NAMELIST Control

Columns	Type	Content
1	-	blank
2 - 8	Literal	'&VALUES'
9		blank
10 - 80	-	Parameter names and corresponding values
9 - 80 (unconfined)	Literal	'&END'

Data Description

The first column in the &VALUES card must be blank. The second column must contain an & and the third through the eighth column the word VALUES. The tenth through seventy-fifth columns may be used to input parameter names and corresponding values. An equals sign should separate a parameter name from its associated value. Commas should be

used to distinguish entries, and blanks may appear anywhere but within parameter names. Following the last entry on the card should be an & followed immediately by the work END. The &END must appear somewhere on the card. Either a comma or a blank may separate the &END from the last entry. Entires cannot be continued on a following card. If a question should arise in coding this card, it is recommended that the appropriate FORTRAN manual be consulted with regard to NAMELIST usage. If the &VALUES cars is coded improperly, the Texas Trip Distribution Package will terminate with a STOP code of 12.

CROSS REFERENCE TABLE

The following table is designed to provide the user a convenient summary of the data sets and data cards associated with each routine.

Cross Reference	DATA SETS																									Pointers		Data Cards		
	INTRIP	REPORT	RECORD	NOWTRP	RAWSEP	NOWSEP	NEGSEP	IMPSEP	MODTRP	ADD1	ADD2	ADD3	ADD4	ADD5	SUMTRP	SWTTRP	NEWSEP	ALTTRP	ASSIGN	PLOTTAPE	SORTIN	SORTOUT	SV	RS		MT	MS			
Default Unit	[21]	[12]	[14]	[2]	[8]	[4]	[1]	[15]	[3]	[3]	[17]	[18]	[19]	[23]	[20]	[24]	[16]	[22]	[13]		9	10	25	26		2	4	5	7	
MODIFY																												I		&VALUES
PERUSE																														
UNPACK	I			0																										
SCREEN		I		S																	S	0								
BUILD			0/I	0																		I				NOWTRP		I		CATEGORY
PRINT																										I				
EDIT					I	0																					NOWSEP	I		SEPARATION
WRITE																											I			
EQUATE																												I		EQUALS
SET							0																			I	I/NEGSEP	I		EQUALS
LIST																										I	I			
REFINE																				0									0	&VALUES FORMAT LENGTH
ACCEPT																												I		FORMAT GENERATION LENGTH BIAS
IMPOSE								0																			I/IMPSEP	I		ADMIT
MODEL									0														0			MODTRP	I/NOWSEP		0	FORMAT BIAS
EXPAND				I					0																	MODTRP				
RANDOM									0						S															
GET																				0						I	I	I		EQUALS
MATCH				I		I			I							0					S	S								
SUM										I	I	I	I	I	0											SUMTRP				
SWITCH																0					S	S				I/SWTTRP				
ALTER						I			I								I	0								ALTTRP	NEWSEP			
PACK																			0							I				
RESTART									0														0	I		MODTRP	I/NOWSEP	I		

I = Input, O = Output, S = Scratch

O T H E R I N F O R M A T I O N

PACKAGE CAPACITY

PROCESSING TIMES

PROGRAM MODULES

SIGNIFICANT FORMULATIONS

- RELATIVE PRODUCTION MODEL
- INTERCHANGE LIMIT CONSTRAINT
- RELATIVE ATTRACTION MODEL
- RELATIVE TRIP LENGTH FREQUENCY MODEL
- CONSTRAINED INTERACTANCE MODEL

PACKAGE CAPACITY

The INTEGER statement in the FORTRAN program, MAIN controls the capacity of the Texas Trip Distribution Package. Variable A in the INTEGER statement, defines the storage for the 11 vectors related to the centroids; the first number in the dimension for variable A controls the maximum number of centroids. Variable B defines the storage for the 8 vectors related to travel separation; the first number in the dimension for variable B controls the largest separation that can be handled (which includes provisions for 4 external movements). Variable C defines the storage for the 5 vectors related to sector interchanges. The first number in the dimension for variable C controls the largest number of sector movements that can be processed. The largest number of sector movements controls, in turn, the maximum number of sectors (and the largest sector designation number) which are permitted. The number of sector combinations may be computed as follows:

sector combinations = (last sector)*(last sector +1)/2

The capacity of the package may, therefore, be changed by simply redimensioning the variables A, B and/or C in the INTEGER statement in the program MAIN.

An exception to this is the MATCH routine. When the MATCH routine is executed, the variable C must be dimensioned for a minimum of 320 sector combinations (corresponding to about 25 sectors) regardless of how many are actually used. If this condition is not met, the Trip Distribution Package will terminate with a STOP code of 9.

At the beginning of execution, the following message is always printed:

EFFECTIVE CAPACITIES

700 CENTROIDS

100 SEPARATIONS (INCLUDING EXTERNAL CODES)

25 SECTORS (325 COMBINATIONS)

The numbers, of course, may vary depending upon the array dimensions being used.

It should be noted that during processing continuous checks are made to insure that effective capacities are not exceeded. If a routine detects that some capacity has been exceeded, the Trip Distribution Package will terminate with a STOP code of 7, and an appropriate message will be printed to explain the situation.

The core storage requirements under various sets of package capacities are summarized in the following table. This table should serve as a guideline for determining the region size needed for any desired set of capacities.

CORE STORAGE REQUIREMENTS UNDER VARIOUS SETS OF PACKAGE CAPACITIES

Capacities					
Centroids	400	700	1600	3300	4800
Separations	50	100	150	200	250
Sector Combinations	325	325	325	325	325
Core Storage Needed for Arrays					
11 Centroid Vectors	4400	7700	17600	36300	51800
8 Separation Vectors	400	800	1200	1600	2000
5 Sector Combination Vectors	1625	1625	1625	1625	1625
Total Words	6425	10125	20425	39525	55425
Total Bytes	25,700	40,500	81,700	158,100	221,700
Array Storage Required	25k	40k	80k	155k	216k
Buffer Requirements Per Data Set*					
Matrix	3.2k**	14.5k	14.5k	14.5k	14.5k
Sort	14.5k	14.5k	14.5k	14.5k	14.5k
Assignment	1.5k	1.5k	1.5k	1.5k	1.5k
Trip Report	2k	2k	2k	2k	2k
Other Core Storage Requirements					
Program Storage (with overlay feature)	51k	51k	51k	51k	51k
System Storage	8k	8k	8k	8k	8k
Sort Allocation (Amount = 110,000)	110k	110k	110k	110k	110k

*Assuming the specifications given under Data Control Block Suggestions
**Assumes LRECL = 1608 and BLKSIFE - 1612

PROCESSING TIMES

Processing times are difficult to estimate. Under MVT, the times printed from the core clock will have significance only if the Texas Trip Distribution Package is the only job being processed by the IBM 360 computer.

Execution times can be controlled to a degree by the user. Several of the routines are heavily input/output bound. IF a large blocking factor is used with the associated date sets, this will improve execution time at a cost to the amount of core storage used. Increasing the AMOUNT of storage allocated to the sort for work space will decrease sort time.

The following table is provided to aid the user in estimating the sort time. It should be recognized that the times are appropriate, and that recent experience has demonstrated considerable variation from the estimates provided.

More zones require more processing time. No relationship has been found to estimate the amount of processing time as a function of the number of zones.

SYSTEM/360 MODEL 50 SORT ESTIMATES

Number of Records (In thousands)	Sort Time in Minutes		
	44K	110K	110K
2	1.1	1.1	1.0
5	1.2	1.2	1.1
10	1.3	1.3	1.2
20	2.0	1.6	1.4
25	2.2	1.7	1.5
50	3.3	2.5	2.0
75	4.5	3.2	2.5
100	5.6	5.7	4.6
125	9.1	6.9	5.5
150	11	8.0	6.4
200	14	11	8.1
300	20	14	11

These estimates are taken from C28-6543-3 IBM Sort/MERGE MANUAL from Feb. 1967.

PROGRAM MODULES

All of the routines prepared especially for use in the Texas Trip Distribution Package, except INVOKE, have been written in the FORTRAN IV programming language. INVOKE has been written in assembly language and functions as an entry into the system sorting package. REREAD is a library routine written in assembly language. It is highly installation dependent and therefore may require alternation to be compatible with many computer systems, if it is not available locally.

Each of the FORTRAN subroutines which has been modified since June 18, 1973 contains a comment card stating whether FORTRAN (G) or FORTRAN (H) should be used to compile it. If it specifies FORTRAN (H), it also gives the highest optimization number for which it has been successfully tested.

SIGNIFICANT FORMULATIONS

The following are some of the significant formulas used by the Texas Trip Distribution Package:

RELATIVE PRODUCTION MODEL

The formula for relative production model is:

$$PZ_i = AN\ (1.0 - e^{(XP)(P_i)})$$

where:

i = zone number

PZ_i = expected number of interactions for production zone i

AN = number of zones with non-zero attraction volumes

XP = exponent for the relative production model

P_i = production volume for zone i.

CONSTRAINED INTERACTANCE MODEL

Basic Formula and Constraints

Mathematically, the constrained interactance model may be stated as follows:

$$T_{ij} = P_i A_j F_l BE$$

Subject to the following constraints:

A. Direct Constraints:

1. For all values of i, P_i must remain fixed (i.e., $\sum_{\text{all } k} T_{ik} = P_i$)

2. $\sum_{\text{all } i} P_i = \sum_{\text{all } i} A_i = \sum_{\text{all } l} F_l$

B. Indirect Constraints:

1. $\sum_{\text{all } k} T_{kj} = A_j$

2. The summation of all trips at a given separation, l, should equal F_l

3. $E = 0 \quad \left(\text{if } \frac{A_j}{S_j} < G_i\right)$

   $E = 1 \quad \left(\text{if } \frac{A_j}{S_j} \geq G_i\right)$

The value of G_i is picked for each zone to meet the production interaction curve. This interaction constraint sets an upper limit for the number of interactions for each zone.

Symbols

T_{ij} = trips produced by zone i and attracted to zone j.

P_i = trips produced by zone i.

A_j = trips attracted by zone j.

F_l = expected number of trips between all zone pairs with a separation l (i.e., trip length frequency).

B = bias correction factor for sector interchange bias.

E = elimination function used to limit the number zone pairs which exchange trips.

S_j = separation between zones i and j

G_i = accessibility limit for zone i

RECENT CHANGES AND MODIFICATIONS