DEVELOPMENT OF A FORTRAN 77 INTERPRETER

By

SUSAN JANE ZIMMER

Bachelor of Science

Oklahoma State University

Stillwater, Oklahoma

1983

Submitted to the Faculty of the Graduate College
of the Oklahoma State University
in partial fulfillment of the requirements
for the Degree of
MASTER OF SCIENCE
July, 1986

DEVELOPMENT OF A FORTRAN 77

INTERPRETER

Thesis Approved:

Thomas A. Cook
Thesis Adviser

Joe H. Mize

Norman N Durham
Dean of the Graduate College

ACKNOWLEDGMENTS

I wish to express my gratitude to all of the people who have helped me with my work at Oklahoma State University. Thanks is due to the Mechanical Engineering Department for supporting this project. I would especially like to thank Thomas A. Cook for his enthusiasm for this project, his willingness to provide well thought-out guidance and his invaluable help in many other areas.

A special thanks is due to Dr. J. A. Wiebelt and Dr. J. H. Mize for their participation on my committee. I would also like to thank the Harris Corporation for providing the College of Engineering with such an outstanding system and facility.

Finally I would like to thank my husband Mark for his encouragement and constant support of my education. Without his moral and domestic help this project would not have been possible. I would also like to thank my 21-month-son, Jay, for his patience in waiting till Mommy has time to play and to my unborn baby for waiting until this project is finished to be born.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

The function of this FORTRAN interpreter is to create a programming environment similar to that of BASIC, when used on a personal computer, while using a main frame system. Whether a program is interpreted rather than compiled creates the difference between the BASIC environment and the main frame environment. To execute a program that is to be compiled, it must be a complete program free of all syntax and structure errors before execution can begin. Execution of a program to be interpreted begins on the first line and continues until an error is detected. At the time an error is detected the programmer knows which line the error is on and has available all of the variable values that have been assigned to that point in the program.

The interpreter may be used as a teaching tool for student programmers. It is frequently a difficult task for a student programmer to write a completely error free program. A student can more readily locate an error by knowing where the interpreter found the error. With an interpreter, a student can also test parts of a program since a complete program is not necessary for execution. The student programmer requires little or no knowledge of the operating system to use an interpreter. This allows the student the opportunity to concentrate on the language being studied without the distraction of learning the operating system.

Use of an interpreter is not only advantageous for the student programmer, but an experienced programmer may find the interpreter a useful debugging tool. The interpreter helps with syntax errors and also allows the programmer to have available all variable values at any point in the program. This allows the programmer to detect any logic flaws in the program.

Interpreters are available for some systems although there appears to be none for FORTRAN. Interpreters for other languages are written in machine language and therefore are machine dependent. One goal was to make this FORTRAN interpreter as portable as possible. To achieve this, the interpreting program needed to be written in a high level language. FORTRAN 77 was chosen because of its widespread use and the author's familiarity with it. It was also considered that a system on which a FORTRAN 77 interpreter could be used could also accommodate a FORTRAN 77 program. The interpreter was written using standard FORTRAN 77. The user's program must be a subset of standard FORTRAN 77 which is defined herein.

A thorough literature search was performed to find an interpreter written in a high level language. No such interpreter has been described in the literature nor were any available in the Software Index. This leads the author to believe that the program described herein is one of a kind and that the generation of this interpreter was long over due.

The description of the interpreter is broken into two separate segments. The first is the description of how the interpreter was written and the logic of the program. This segment is contained in the body of this thesis. The second is an Interpreter User's Guide which is

directed to students with minimal programming experience and which also contains the limits of the interpreter. This segment is contained in Appendix A.

CHAPTER II

PROGRAM OVERVIEW

Limitations of Interpreter

The FORTRAN 77 interpreter is designed to interpret a subset of standard FORTRAN 77 commands. Any program that can be executed by this interpreter can also be compiled by a FORTRAN 77 compiler. The subset was selected on the basis of common usage of the commands. This subset should accommodate the average programmer's needs but will fall short of the needs of the advanced programmer.

All data types were implemented with the exception of the complex data type. Complex numbers are rarely used in most programming and were therefore not included in the subset. Only one and two dimensional arrays are included. In an attempt to keep the interpreter program memory requirements to a reasonable size, three dimensional and larger arrays are not allowed in the subset.

A list of FORTRAN statements and structures allowed by the interpreter are shown in Table I. This table also gives any limitations that apply to that element. In addition to these statements, comment statements are allowed at any point in the program.

Program States

When using this FORTRAN interpreter, named INTERP, the user is in one of several modes or states. Each state has different options as to

TABLE I

FORTRAN 77 SUBSET ALLOWED BY INTERP

DESCRIPTION OF STATEMENT	SYNTAX FORM	LIMITS/MAXIMUMS
Program Title	PROGRAM name	
Define type of data	INTEGER v1[,v2]*	max of 20 names
	REAL v1[,v2]	max of 20 names
	LOGICAL v1[,v2]	max of 20 names
	DOUBLE PRECISION v1[,v2]	max of 20 names
	CHARACTER v1[,v2]	max of 20 names each storage location is 60 characters long
Dimension of maximum dimensional arrays only. maximum size allow: 10-INTEGER - (100,100) 10-REAL - (100,100) 10-DOUBLE PRECISION (100,100) 5-CHARACTER (100) 5-LOGICAL (100)	DIMENSION A1(I),A2(I1,I2)	1 and 2 size of
Logical IF	IF(re)statement	1- logical operator 2- relational operators

Block IF	IF(re)THEN
	statement block
	ELSE
	statement block
	ENDIF
		1-logical operator
		2-relational operators

ELSE IF structure	IF(re)THEN
	statement block
	ELSE IF(re)THEN
	statement block
	ELSE IF(re)THEN
	statement block
	.
	.
	.
	ELSE
	statement block
	ENDIF
		1-logical operator
		2-relational operators
		
		no limit on number of ELSE IF structures

* see symbol definitions at end of table

TABLE I (Continued)

Nested IF	IF(re)THEN
	  IF(re)THEN
	  statement block
	  ELSE
	  statement block
	  ENDIF
	ELSE
	  statement
	  block
	ENDIF
		maximum of 5 levels of nesting

Establishing loop Parameters	DO sn[,]v1=v2,v3[,v4]	
Continue	CONTINUE	
Unconditional GOTO	GOTO sn or GO TO sn	
Computed GOTO	GOTO(sn1,sn2[,sn3])[,]v	
Pause with restart	PAUSE	
Read,list direct	READ(u,*)[list]	
Write,list direct	WRITE(u,*)[list]	
Stop with no restart	STOP or END	

SYMBOLS:

st	statement
sn	statement number
v	variable name
X	number
vX	variable name or a number
u	unit number of device
re	relational or logical expression
[ ]	optional

relational expressions include:

.LT.	less then
.LE.	less then or equal
.EQ.	equal
.NE.	not equal
.GT.	greater then
.GE.	greater then or equal

TABLE I (Continued)

logical operators include:
.AND.
.OR.

which other state may be addressed. A flowchart of the different states and their interactions is shown in Figure 1. The different states are as follows:

INTRO

FILE SELECT

RUN

ERROR

PAUSE

END

PATHWAY

INTRO State. The INTRO State is the first state the user is in when he enters INTERP. In this state, INTERP welcomes the user to the program. INTERP immediately pushes the user into the FILE SELECT State.

FILE SELECT State. The FILE SELECT state asks the user for the name of the program to be interpreted. The user's program must have been previously stored as a file. Once the user's program name has been entered, INTERP pushes the user into the next state which is the Run State.

RUN State. In the RUN State INTERP reads one line of code, parses it, checks for errors and takes appropriate action. If there are no errors found in the line, the appropriate action is taken and then the next line of code is read. This continues until an error is found or until a PAUSE or END statement is encountered. Depending upon which of these conditions is encountered, INTERP will push the user into the ERROR State, PAUSE State or END State, respectively.

ERROR State. The ERROR State is entered when INTERP has recognized an error. This error could be one of a variety of different types

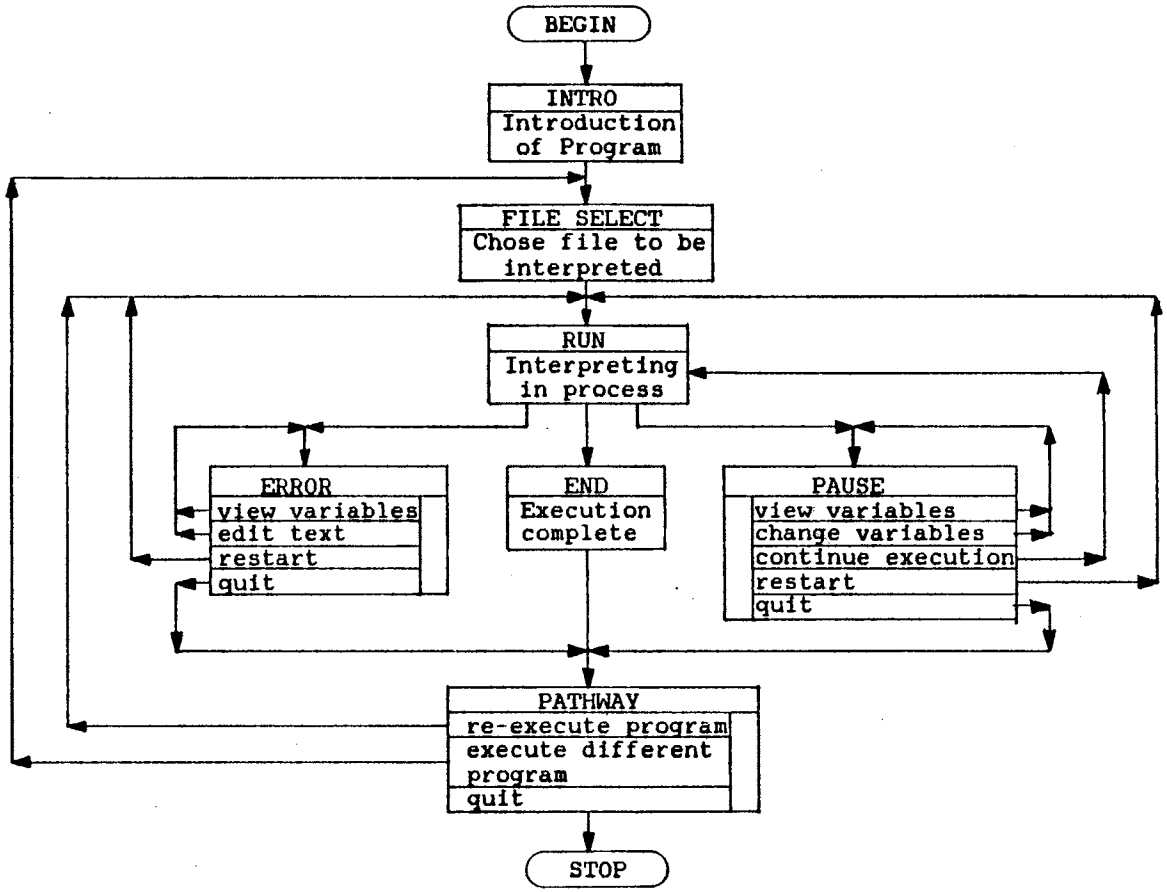


Figure 1. Flowchart of Program STATES

including syntax, data type, structure or run time.

When an error is encountered the subroutine ERROR is called. A flowchart of subroutine ERROR is shown in Figure 2. Subroutine ERROR prints the line in which the error was found and the error number. The following menu is then displayed.

```
ERROR MENU:

1    DISPLAY VARIABLE VALUES

2    EDIT TEXT

3    RESTART EXECUTION FROM BEGINNING

4    QUIT OR EXECUTE DIFFERENT PROGRAM

PLEASE ENTER THE NUMBER OF YOUR CHOICE
```

If the user chooses option 1, subroutine VAROUT is called. The flowchart of VAROUT is shown in Figure 3. The user then has the following choices.

```
VARIABLE MENU

1    DISPLAY ALL VARIABLE VALUES

2    DISPLAY ONE VARIABLE VALUE

3    RETURN

PLEASE ENTER THE NUMBER OF YOUR CHOICE
```

If the user chooses to display all variables, subroutine VAROUT scans all user defined variables and prints out each variable name and its value. Simple variables are printed first in the order of integer, real, double precision, character and logical. The array variables are then printed in the same order. All user defined variables are printed whether a value has been assigned to it or not. The variable menu is then displayed again. The storage and structure of these variables is discussed in Chapter III.

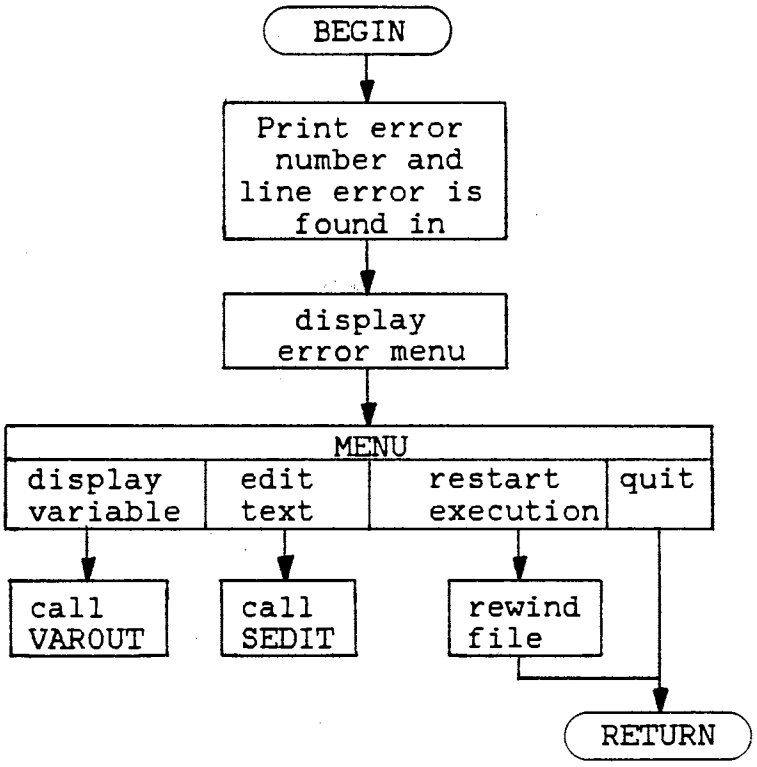


Figure 2. Flowchart of Subroutine ERROR

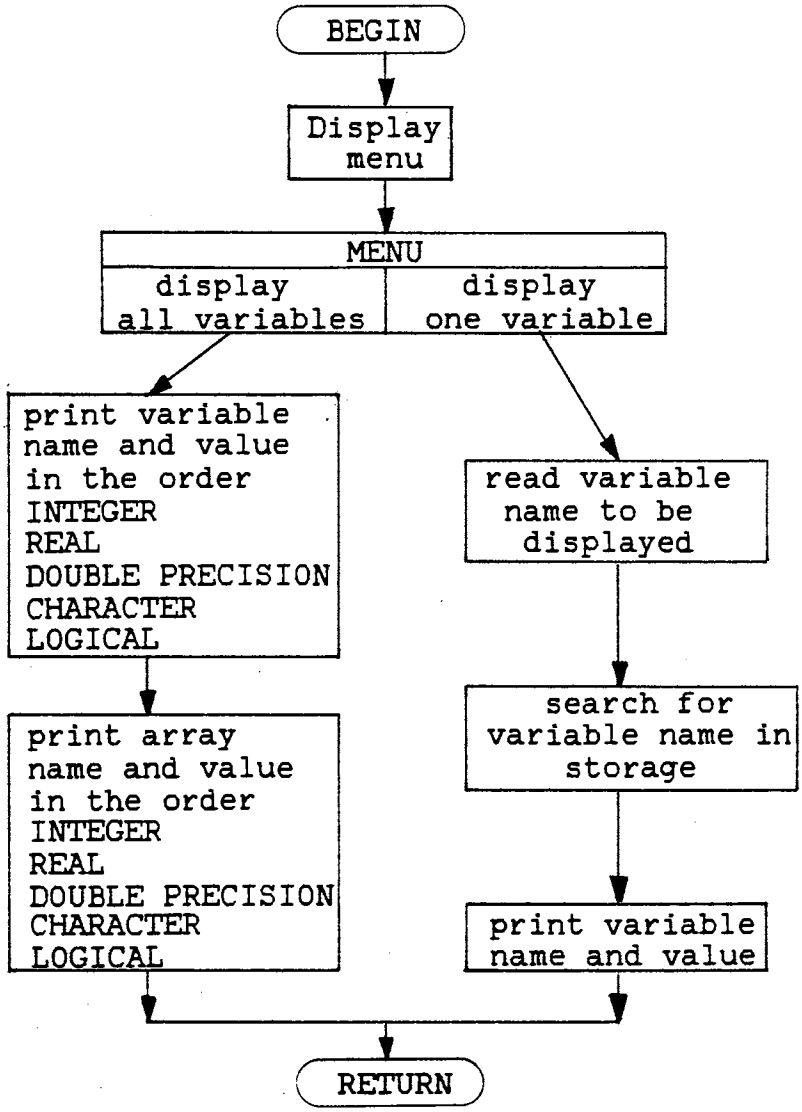


Figure 3. Flowchart of Subroutine VAROUT

When only one variable is to be displayed, the user is asked to 'enter the name of the variable to be displayed'. The variable name is searched for and the variable name along with its value is printed out. The variable menu is then displayed.

If the user chooses the return option, the error menu is displayed. If the user chooses the second option on the error menu, (edit text) the subroutine SEDIT is called. The flowchart for subroutine SEDIT is shown in Figure 4. Subroutine SEDIT will display the 10 lines of text nearest the line with the error. If INTERP has not been beyond the line with the error, the 10 lines displayed will be the 10 lines immediately preceding the line with the error. If INTERP has been past the error line, with an IF THEN ELSE or a GOTO or similar structure, the five lines before and after the error line are displayed. The line containing the error is marked with an arrow for easy identification. The following menu is displayed below the lines of text.

```
EDIT MENU
1    DISPLAY DIFFERENT TEXT
2    INSERT A LINE
3    DELETE A LINE
4    CHANGE A LINE
5    QUIT (RETURN TO ERROR MENU)
PLEASE ENTER THE NUMBER OF YOUR CHOICE
```

The INTERP editor, will not allow any editing beyond the farthest advancement in the user's program. INTERP reads the users program line by line and has no knowledge of the upcoming code. If the user attempts to edit a line that has not been read, he is informed that this cannot be done and is asked to make a different choice.

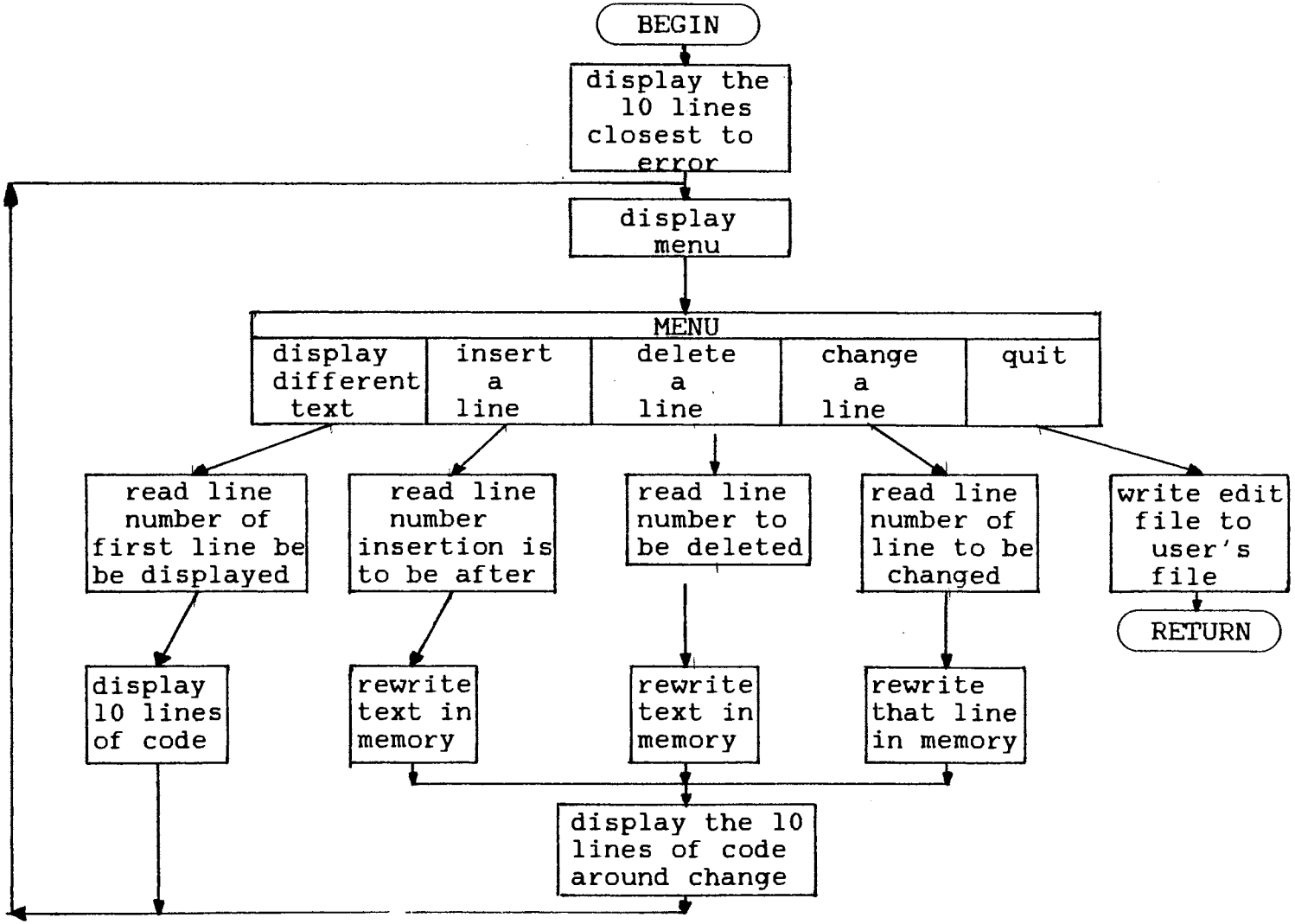


Figure 4. Flowchart of Subroutine SEDIT

When option one is chosen the user is asked to 'enter the line number at which you wish to begin the display'. That line and up to nine lines following it are displayed, and the edit menu is displayed below the text.

If the user chooses to insert a line, he is asked to 'enter line number after which you wish to make the insertion'. Once an appropriate number is entered, INTERP moves all of the interpreted code through the line number indicated to a SAVE file. The user is then asked to 'enter text of line to be inserted'. The format of the line of text must be in the same format as the program being interpreted. This includes spacing over at least seven columns before coding a statement. The new line is read into a character array called SAVE and the remainder of the interpreted text is written to SAVE. All of SAVE is then placed back in storage in INTERP. The 10 lines bracketing the new line are then displayed along with the edit menu.

If the user wishes to delete a line (option 3) he is asked to 'enter line number you wish to delete'. Once an appropriate number is entered INTERP writes all of the interpreted code up to the line to be deleted to a SAVE file. The deleted line is skipped and the remaining lines are written to SAVE. The entire SAVE file is then placed back in storage in INTERP. The 10 lines bracketing the deleted line are displayed and again the error menu is shown.

If option 4 is chosen the user is asked to 'enter line number of the line you wish to change'. Once an appropriate number is entered the user is asked to 'enter line as desired'. The entire line of code must be reentered and must match the format used when the program was written. Once the new line of code has been entered, the 10 lines

bracketing it are displayed along with the edit menu.

When option five is chosen the rest of the user's file which has not been interpreted is read by INTERP. The new edited version of the program is then written over the top of the user's file. The error menu is then displayed.

If the 'restart execution from beginning' option is chosen from the error menu, INTERP rewinds the users file being interpreted, clears all tables and registers, and returns to the RUN State.

If the last option of the error menu is chosen the user is pushed into the PATHWAY State which will be discussed later.

PAUSE State. If a pause statement is encountered while in the RUN State, the subroutine SPAUSE is called and the user is pushed into the PAUSE State. The flowchart of subroutine SPAUSE is shown in Figure 5. The PAUSE State, although similar to the pause statement used in standard complied FORTRAN, is much more flexible in INTERP. The user is given the following options in the PAUSE State.

PAUSE MENU

1 DISPLAY VARIABLE VALUES

2 CHANGE A VARIABLE VALUE

3 CONTINUE EXECUTION FROM THIS POINT

4 START EXECUTION FROM BEGINNING OF PROGRAM

5 QUIT

PLEASE ENTER THE NUMBER OF YOUR CHOICE

If option one is chosen, subroutine VAROUT is called and the program progresses exactly as it did when the display variable value option was chosen from the error menu. This option is quite useful in

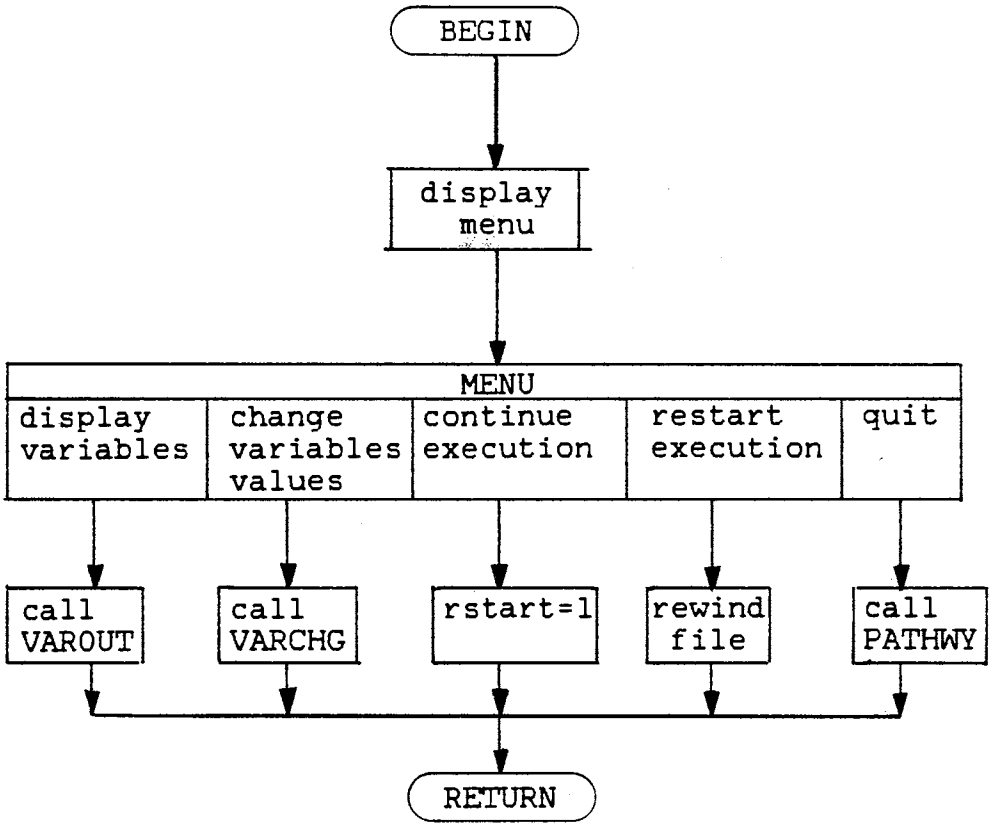


Figure 5. Flowchart of Subroutine SPAUSE

attempting to debug computational programs.

If the change variable option (option 2) is chosen, subroutine VARCHG is called. The flowchart for VARCHG is shown in Figure 6. In subroutine VARCHG the user is asked to 'enter the name of variable you wish to change.' The variable name is searched for and the name along with its value are printed. The user is then asked to 'please enter the new value for "variable name."' Once a value is entered the user is asked 'do you want "variable name" = "variable value" (yes/no).' If the response is no the user again is asked to enter the new value for "variable name." If the response is yes the new value is stored under the variable name and the pause menu is redisplayed.

If the user chooses the third option and wishes to continue execution from this point, INTERP pushes the user back to the RUN State and interpretation continues. If option 4 is chosen (to start execution from the beginning of the program) the user file is rewound, all tables and registers are cleared and the user is pushed back into the RUN State. If the last option to quit is selected, the user is pushed into the PATHWAY State.

END State. If neither an error nor a pause statement are encountered and the end of the program is reached properly, the user is pushed into the END State and informed that 'program "filename" has been interpreted (press enter to continue).' Requiring an input from the user allows any output on a screen to remain displayed until the user is ready to proceed. Once the user presses enter, the user is advanced to the PATHWAY State.

PATHWAY State. When the PATHWAY State is entered, subroutine PATHWY is called. The flowchart for subroutine PATHWY is shown in

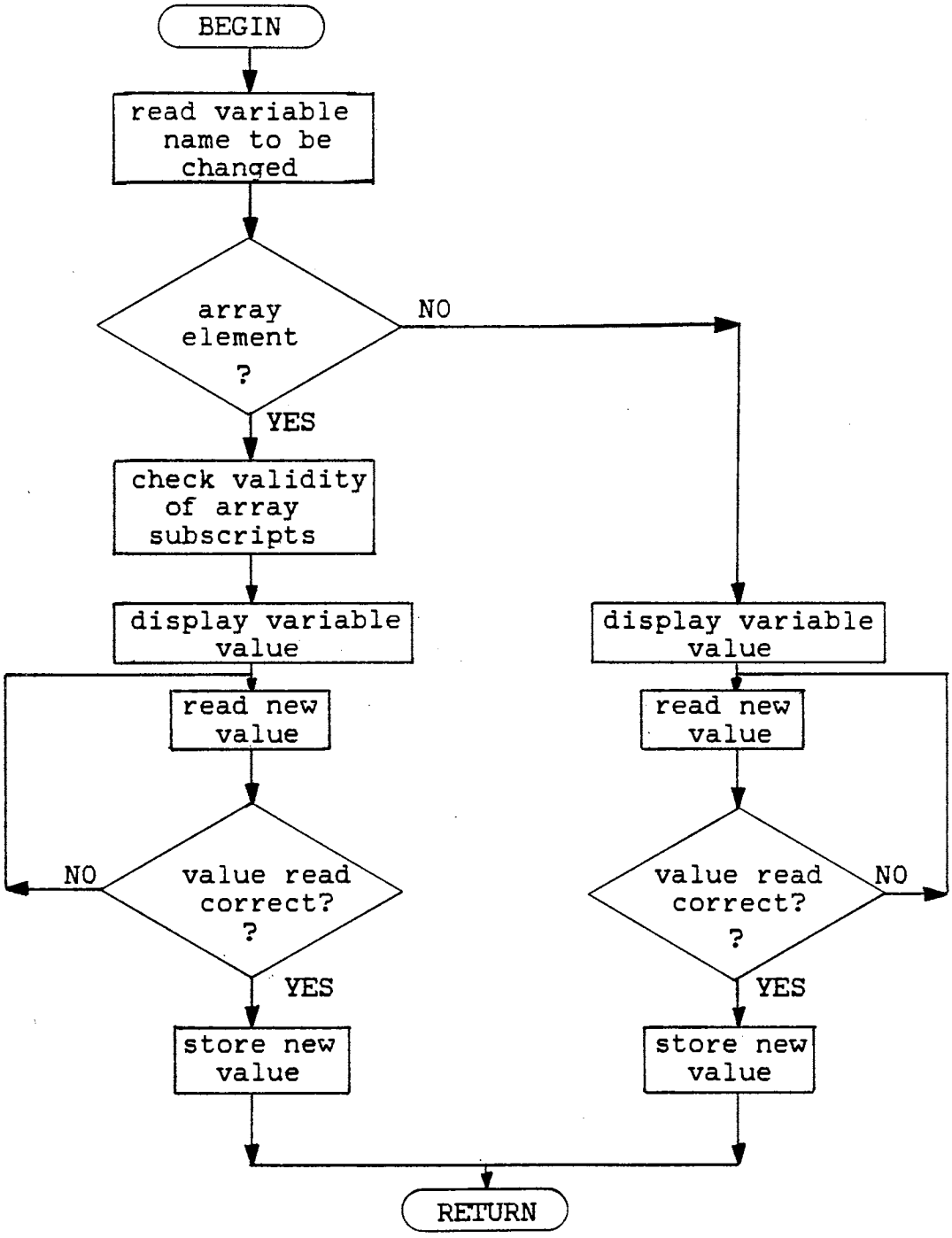


Figure 6. Flowchart of Subroutine VARCHG

Figure 7. Subroutine PATHWY gives the user the following options.

PATHWAY MENU

1 RE-EXECUTE PROGRAM

2 EXECUTE A DIFFERENT PROGRAM

3 QUIT WORKING

PLEASE ENTER THE NUMBER OF YOUR CHOICE

If The user chooses to re-execute the program, the user's file is rewound, all tables and registers are cleared and the user is pushed into the RUN State.

If the interpreting of a different program is desired, the user's file is closed and the user is pushed into the FILE SELECT State. If the last option (to quit working) is selected, the user's file is closed and INTERP informs the user 'you are leaving INTERP, Have a nice day.' INTERP then completes its own execution.

General Program Structure

The general program structure of the interpreter is a main program and many subroutines, each of which is called for a particular type of function.

The main program begins by welcoming the user to 'INTERP' and reads the filename of the program to be interpreted. A flowchart of the main program is shown in Figure 8. Calls are then made to a variety of subroutines to clear registers and tables to begin interpreting a new program. The user is then informed that interpretation of his program has begun and the first line of the user's program is read. As each line is read it is stored in a character array.

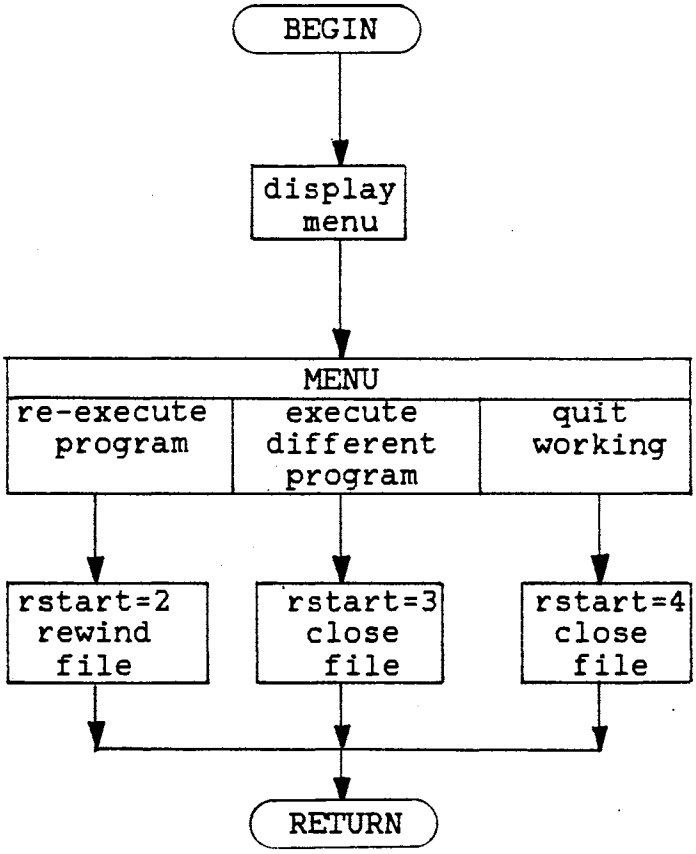


Figure 7. Flowchart of Subroutine PATHWY

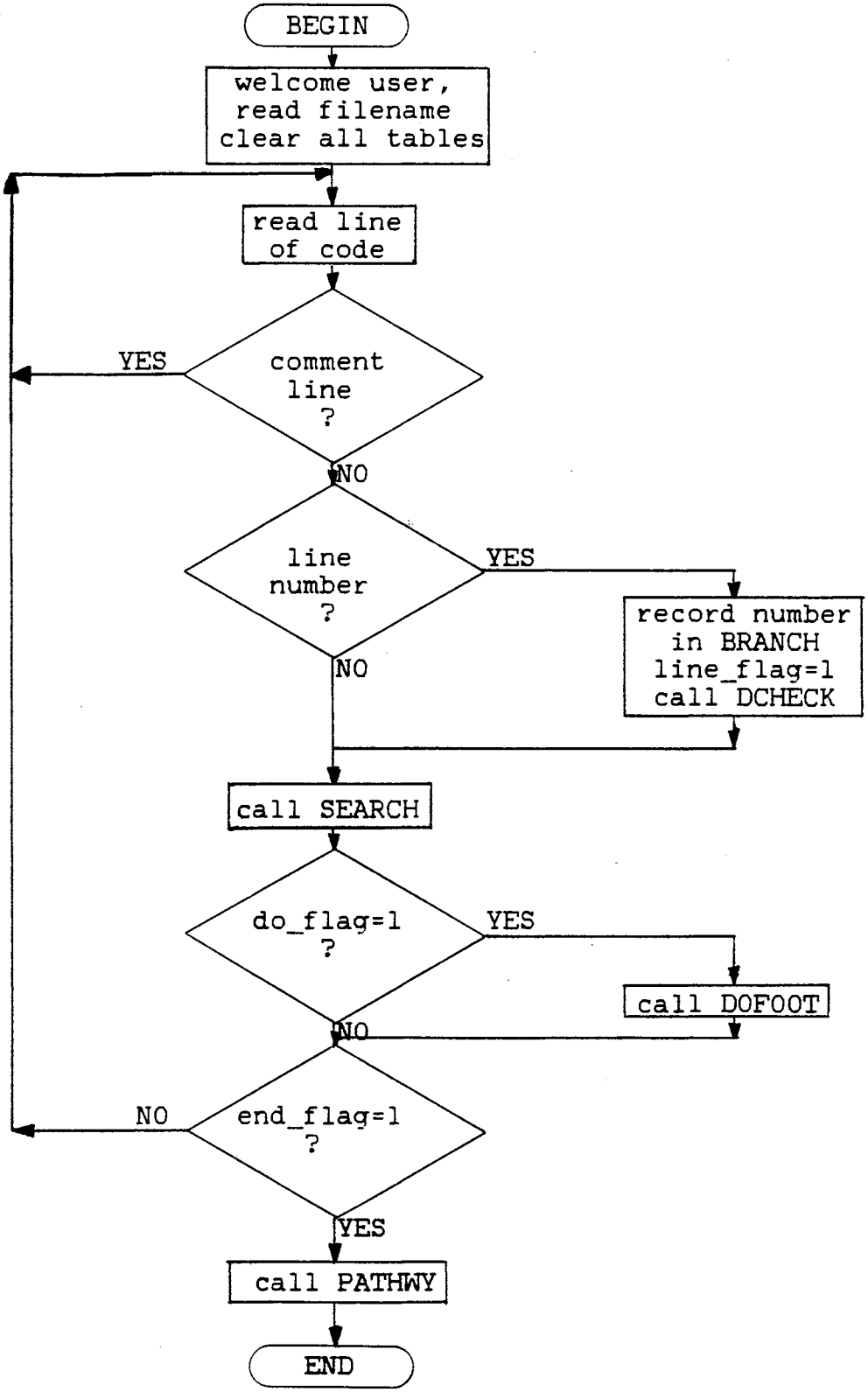


Figure 8. Flowchart of INTERP

The line is checked to see if it is a comment line. If it is, the program advances to the next line. If the first column is empty, the first five columns are then checked for a statement number. If a statement number is found, it is recorded in a table called BRANCH. This is done by calling subroutine MAP. A flowchart of subroutine MAP is shown in Figure 9. Subroutine MAP is called to either clear the BRANCH table or to record a statement number. The BRANCH table is explained more fully in Chapter IV where control statements are discussed.

If there was a statement number in the line, subroutine DCHECK, standing for DO loop check, is called. Subroutine DCHECK checks to see if this line is a DO foot. If it is a DO foot a DO-flag is set and checks are made on the statement type and the DO structure. This is also explained more fully in Chapter V.

Subroutine SEARCH is then called. A flowchart of subroutine SEARCH is shown in Figure 10. Subroutine SEARCH is the subroutine that initially scans the line of code and directs the path of execution the program is to take. Subroutine SEARCH begins by eliminating blanks at the beginning of the line of code. A key word is then searched for. If that key word is not found, the next key word is searched for until one is found. The appropriate subroutine is then called. If no key words are found an error is given. The order that the key words and symbols are searched for is shown below:

PROGRAM

INTEGER

REAL

DOUBLE

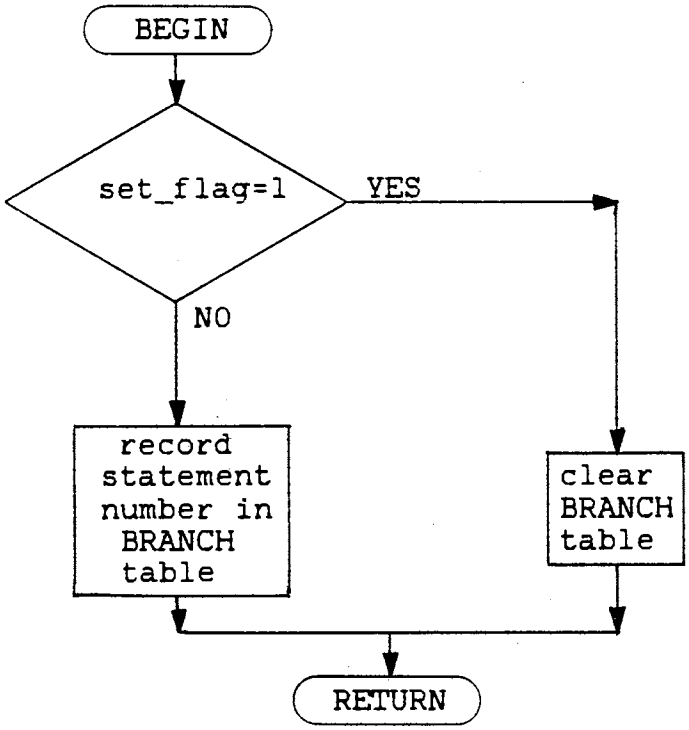


Figure 9. Flowchart of Subroutine MAP

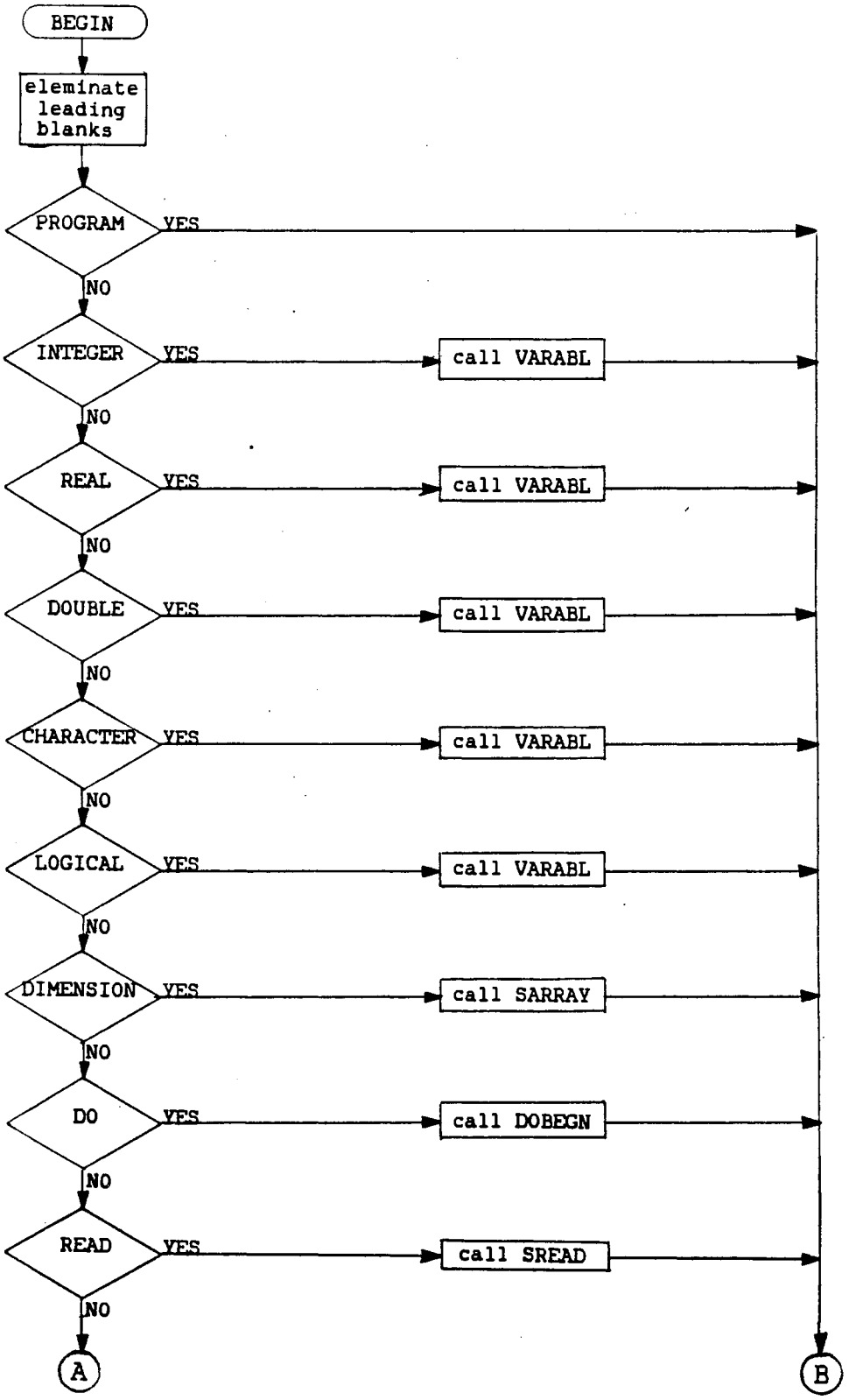


Figure 10. Flowchart of Subroutine SEARCH

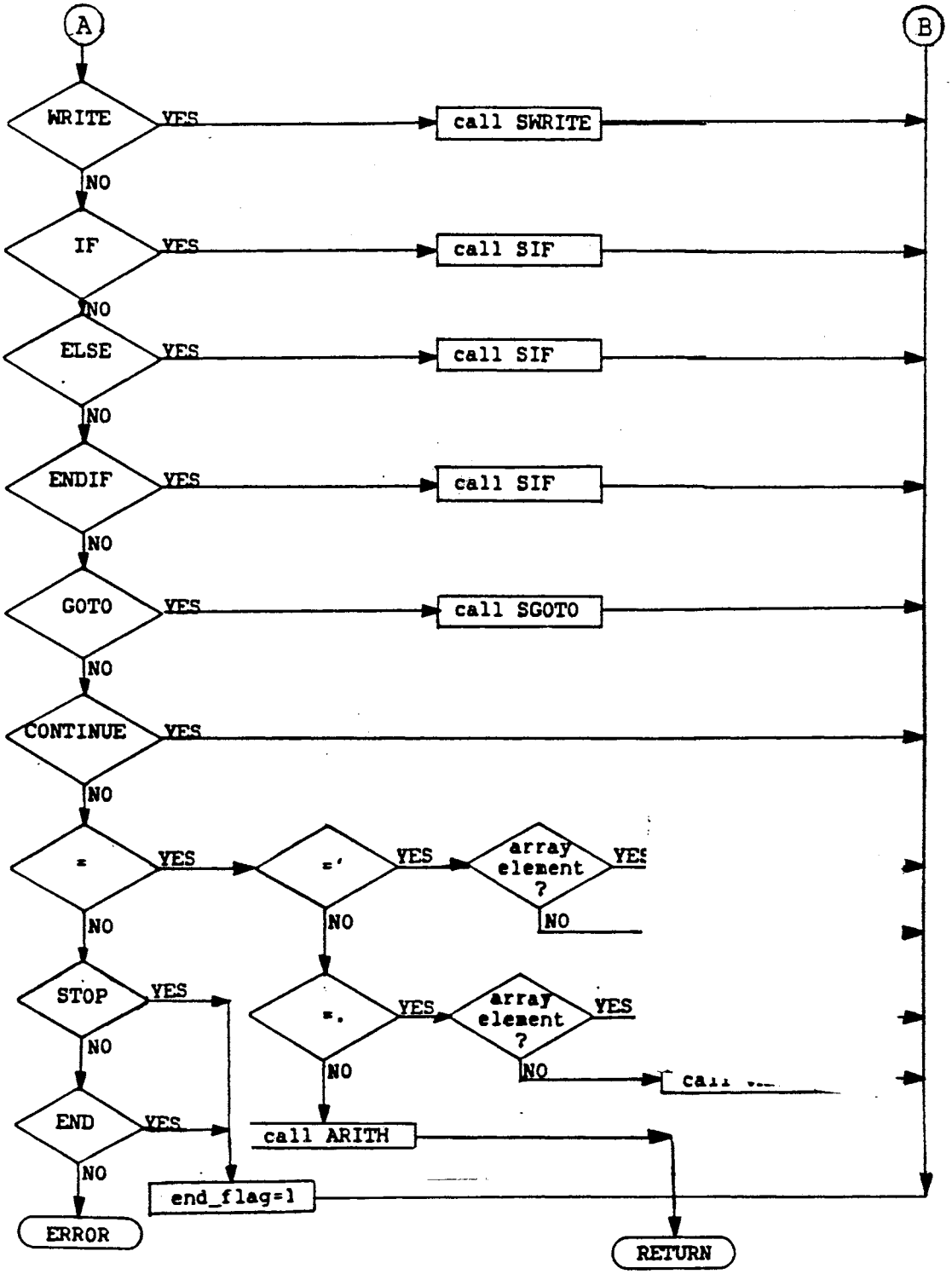


Figure 10. Continued

CHARACTER

LOGICAL

DIMENSION

DO

READ

WRITE

IF

ELSE

ENDIF

GOTO

CONTINUE

=' (character assignment)

=. (logical assignment)

= (arithmetic expression)

STOP

END

Once a line of code has been found further down the list than the DIMENSION Statement, a flag is set and the first seven key words are no longer searched for. There is no reason for this order of search and the search could be done in any order with one exception. The WRITE Statement must be searched before the '=' Statement. If this is not done any WRITE Statement with a '=' in it would be mistakenly interpreted as an assignment statement. If the key word found is STOP or END the end-flag is set.

After subroutine SEARCH has called the appropriate subroutine and control has been returned to subroutine SEARCH, SEARCH returns the control to the main program.

If the DO-flag has been set in the main program, the subroutine DOFOOT is called which updates DO loop parameters and will be further explained in Chapter V. A check is then made of the end-flag. If the end-flag has not been set, the next line of code is read. Each time a line of code is read, it must be checked to see whether or not this line has already been read. This bookkeeping is done by the two variables, LPOINT and LINDEX. LPOINT is the line number of the line being interpreted and LINDEX is the line number of the maximum progression that has been made through the program. If the end-flag had been set, the user is informed that his program has been interpreted and he is advanced into the PATHWAY State.

Throughout the entire program a direct pathway back from the ERROR State is needed. This requires that after every call to subroutine ERROR, the subroutine that called ERROR must return to its calling program and that calling program must return to the one that called it. This is accomplished by checking the variable RSTART value after every call that is made. The RSTART value will be something other than one if a direct return to the main program is needed.

CHAPTER III

HANDLING OF VARIABLES

Data Types Allowed and Data Restrictions

It was desirable for the interpreter to handle all data types needed by the average programmer. The following data types are allowed by the interpreter:

INTEGER

REAL

DOUBLE PRECISION

CHARACTER

LOGICAL

In each data type the user is allowed up to 20 variable names. All variable names used in the program must be declared in a type statement and cannot exceed six characters in length. The length of user defined character variables cannot exceed sixty characters.

All arrays must be dimensioned in the DIMENSION statement and the array name must have been previously declared in a type statement. When consideration was given to the amount of space the INTERP program was to occupy, it was decided to limit the dimensions of the user's arrays. For the types integer, real and double precision the user is allowed up to ten - one or two dimensional arrays with maximum size of each to be 100 by 100.

Character arrays have up to 60 characters in each element and are allowed five one dimensional arrays of up to 100 elements each. Logical data is also allowed five one dimensional arrays of up to 100 elements each. It is believed that these limits should be adequate for most users. However, if the user of the interpreter wishes to alter any of the above restrictions, he can do it easily be done by altering the array storage dimension in subroutine VARABL or SARRAY. These subroutines are explained below.

Variables

The storage of simple variable names and their values is handled by subroutine VARABL. A flowchart of VARABL is shown in Figure 11. Subroutine VARABL can be called for one of the four reasons below.

0. Initialize a variable name
1. Store a variable value
2. Retrieve a variable value
3. Remove variable name to make it an array name

The first parameter in the parameter list passed to the subroutine indicates which task is to be performed. If subroutine VARABL was called because a data statement was being interpreted, the initializing of a variable name would occur. All variable names are stored in the character array VARTBL, standing for variable table. This table is illustrated in Figure 12. Array VARTBL is dimensioned to 20 by 5. Each of the five columns are used for a different data type. For example, if the variable name ROOT was of type real the character string 'ROOT' would be stored in the second column of VARTBL. Array VINDEX is of length five and keeps track of how many variable names of each type have

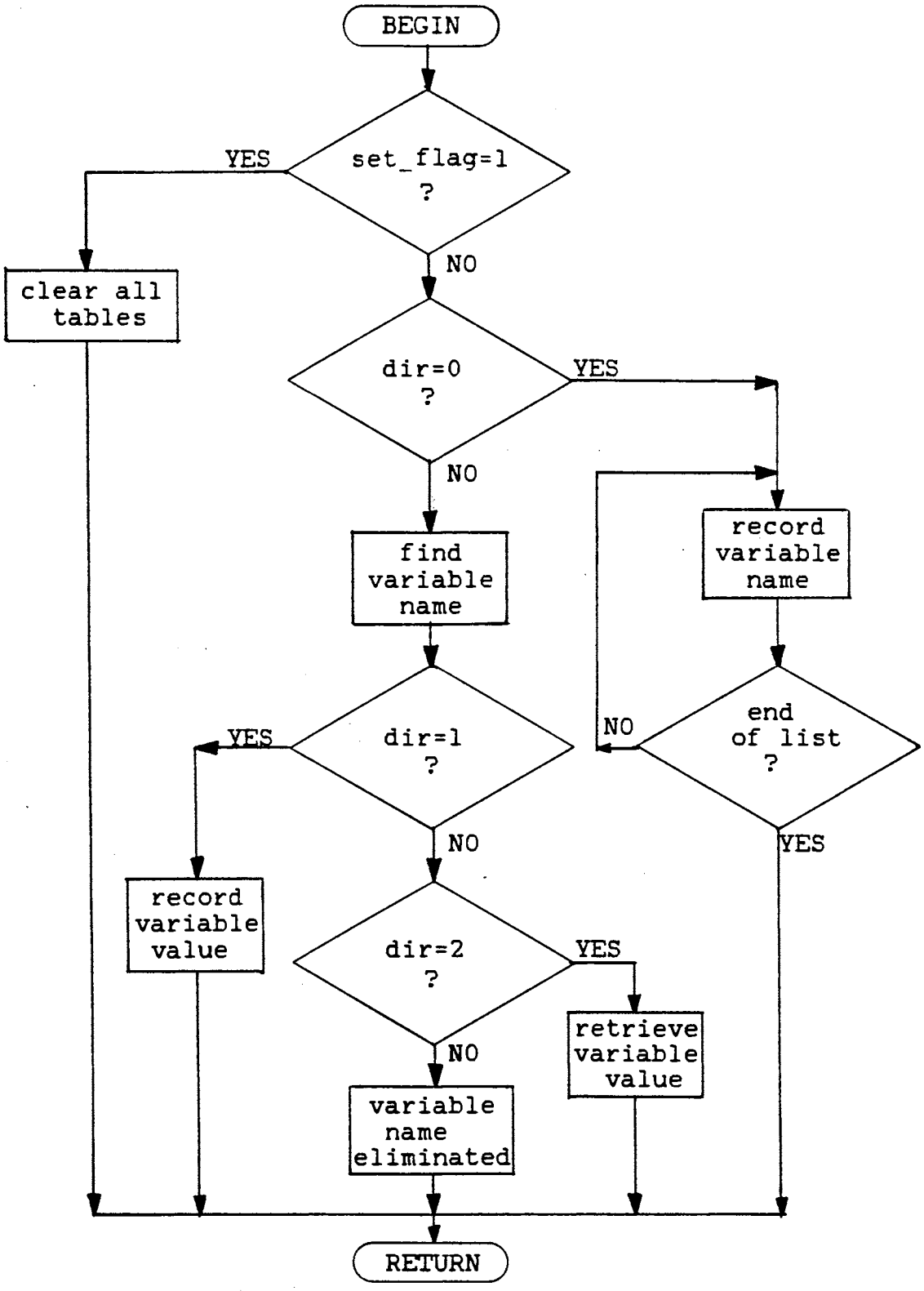
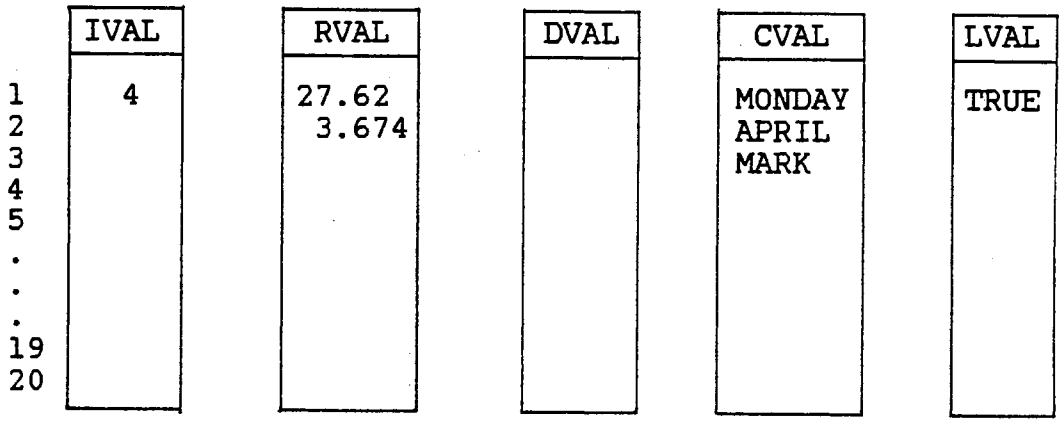


Figure 11. Flowchart of Subroutine VARABL

	INTEGER	REAL	DOUBLE PRECISION	CHARACTER	LOGICAL
1	SUM	ROOT		DAYS	FLAG
2		ANSWER		MONTHS	
3				NAMES	
4					
5					
.					
.					
.					
19					
20					

Character Array VARTBL



Arrays used to store variable values

Figure 12. Example of Single Variable Storage

been declared.

If the task to be performed by subroutine VARABL is to assign a variable value, the variable name, variable type and the value to be assigned are all passed in the parameter list. Five of the variables in the parameter list are to provide pathways for values of each type to be passed to and from the subroutine. The user defined variable name is searched for in array VARTBL. When located, the row and column, or index and type respectively, are recorded. The index and type values are used to link the variable name to its value.

Subroutine VARABL provides five arrays, each of length 20 to store variable values. Each array stores a different type of data. This is demonstrated in Figure 12. If the user defined character variable Months has a value of 'APRIL' the variable would have an index of 2 and type of 4. In an assignment statement the character string 'APRIL' would be stored in the character array CVAL(2).

If the variable name type and the type of the value sent to be stored are the same, the value is stored in the appropriate linked array. If the variable name type and the value type are different, checking must be done. If the value is of type character or logical an error is called. If the types are integer, real or double precision the value is converted to the variable name type. When making this assignment the interpreter uses the rules shown in Table II.

If the task of the subroutine VARABL is to return a variable value, the subroutine would receive the variable name as a parameter. The variable is searched for and the index and type linking values are found. Through linking of these values, the variable value is found. The variable value and type are then returned to the calling program.

TABLE II

MIXED MODE ASSIGNMENTS

VN = Variable Name
VV = Variable Value

VN \ VV	INTEGER	REAL	DOUBLE PRECISION	CHARACTER	LOGICAL
INTEGER	VN=VV	TRUNCATE VV	TRUNCATE VV	ERROR	ERROR
REAL	APPEND.0 TO VV	VN=VV	VV=MSP * ROUND LSP	ERROR	ERROR
DOUBLE PRECISION	APPEND.0 TO MSP *	MSP=VV * LSP=0	VN=VV	ERROR	ERROR
CHARACTER	ERROR	ERROR	ERROR	VN=VV	ERROR
LOGICAL	ERROR	ERROR	ERROR	ERROR	VN=VV

* MSP : most significant portion
LSP : least significant portion

If the call to subroutine VARABL is from subroutine SARRAY, the task of VARABL will be to eliminate a variable name. This allows SARRAY to make that name an array name. The variable name is passed to VARABL. The variable name is searched for and eliminated. The type of the variable is returned to SARRAY and is used in creating the array.

Arrays

Arrays are handled in a similar fashion as simple variables. Subroutine SARRAY will be called if one of the following tasks needs to be performed.

0. initialize array name
1. store an array value
2. retrieve array values

If the interpreter encounters a DIMENSION statement, subroutine SARRAY is called to initialize the array name. A flowchart of subroutine SARRAY is shown in Figure 13. The name of the array is read and VARABL is called to find the type of the array name that had been declared and to remove the name from the simple variable list. The array names are stored in the character array, ARAYTB which is constructed exactly as VARABL was for simple variable name storage.

The subscripts are then read for the user's dimensions. The subscript values are stored in linked fashion in a three dimensional array called AINFO. The row and column or index and type respectively, are the same in ARAYTB as well as in AINFO. The AINFO table is three locations deep with each level holding the following values:

AINFO(index,type,1) = dimension (1 or 2)

AINFO(index,type,2) = maximum number of rows

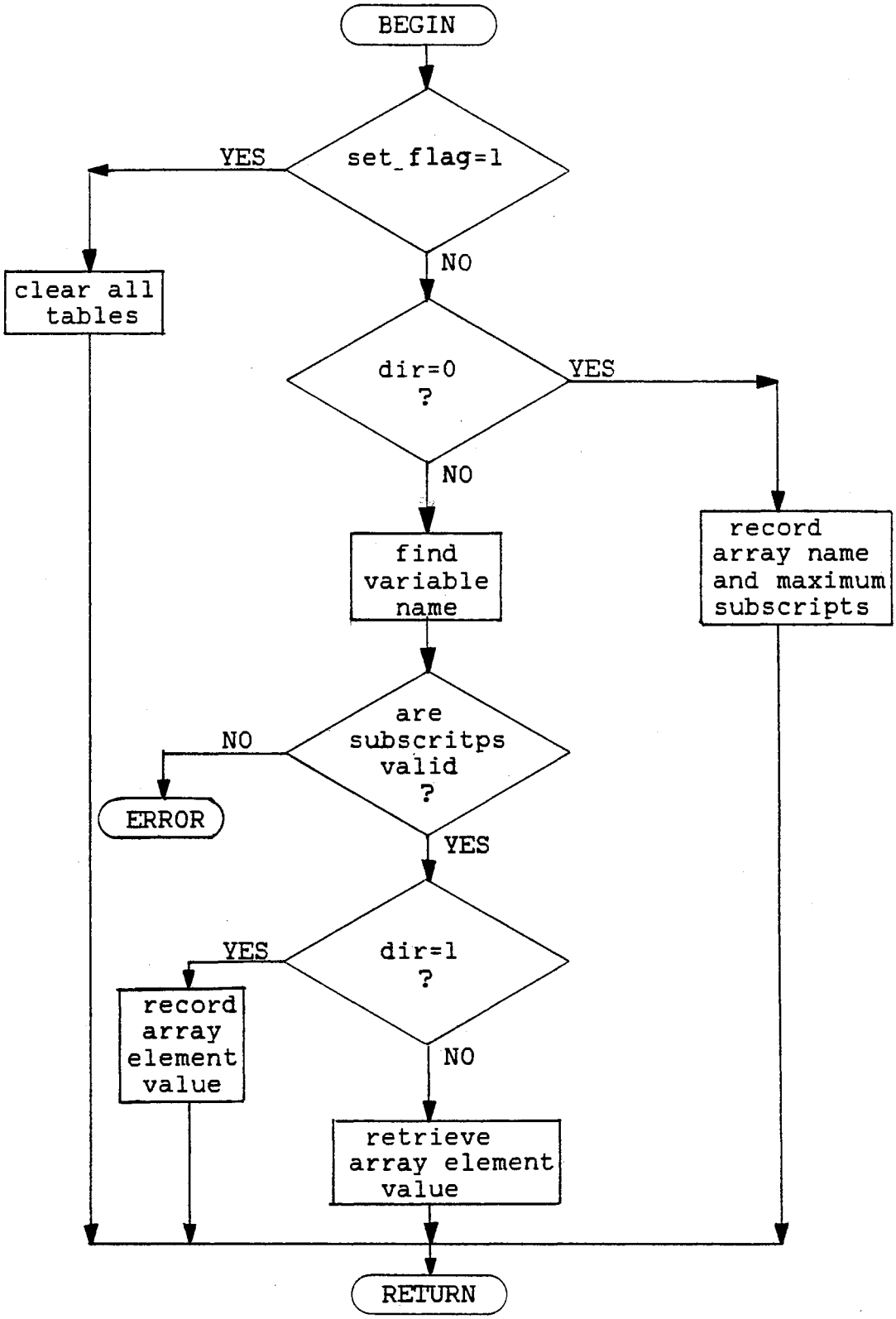


Figure 13. Flowchart of Subroutine SARRAY

AINFO(index,type,3) = maximum number of columns

This is illustrated in Figure 14.

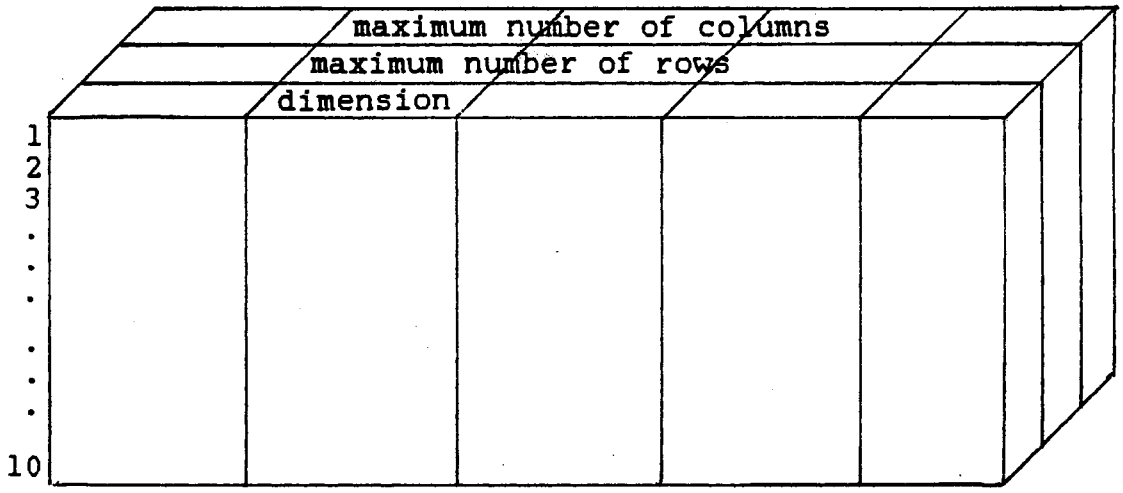
If subroutine SARRAY was called to store or retrieve a value, the array name is searched for and the subscript values are read. A check is made to ensure the subscript values are within the user's defined dimensions.

The storage of the array values is in five separate arrays. The integer, real and double precision arrays are three dimensional arrays of dimension 100 by 100 by 10. This gives the user a maximum of 10 arrays each up to 100 by 100. The index of the array name links the values to the correct array. If a value was to be stored in a user defined array named STRESS it would be placed in the desired row and column and in the second plane of the real array RARAY.

The character and logical arrays are stored in two dimensional arrays since the user is only allowed one dimensional arrays of the type. The first subscript of the character or logical array is the user's defined subscript. The second is the index used to link the array name to the correct array location. It is seen that if the user wishes to change the storage space available to the programmer it can easily be done.

	INTEGER	REAL	DOUBLE PRECISION	CHARACTER	LOGICAL
1		STRAIN			
2		STRESS			
3					
.					
.					
.					
10					

Character Array ARAYTB



Array AINFO

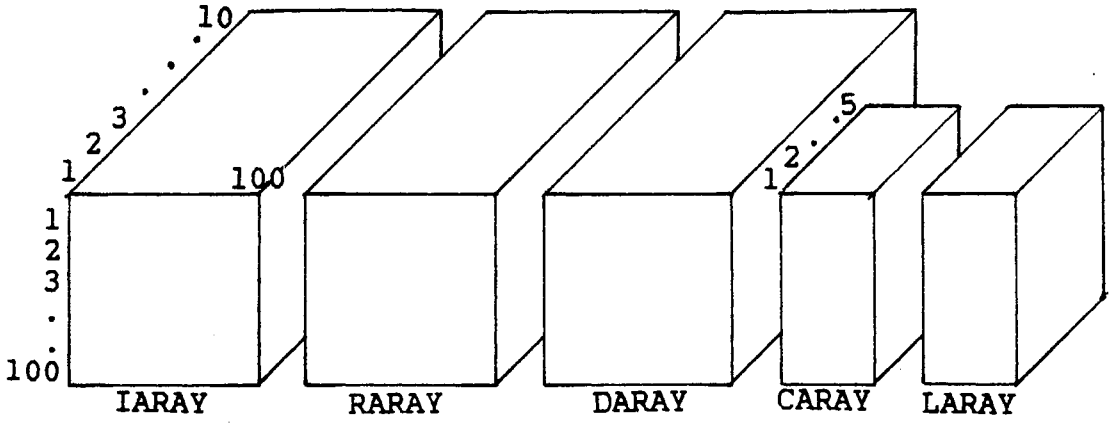


Figure 14. Example of Array Variable Storage

CHAPTER IV

ASSIGNMENT STATEMENTS

Introduction

The assignments of variable values and mathematical calculations are made with assignment statements. During the search for keywords in subroutine SEARCH, any statement that contained an equal sign and was not a write statement or an IF statement, was considered an assignment statement. The assignment of a logical or character variable is handled differently than an arithmetic assignment.

Character Assignments

Character assignments are handled completely in subroutine SEARCH. Once an equal sign is found it is checked to see if a quotation mark follows the equal sign. If there is a quotation, the character string is isolated and the left of the equal sign is read to see whether or not the variable is an array. A call to either subroutine VARABL or SARRAY is made and the character string is stored under the variable name. Once a character variable has an assigned value it is handled like any other variable with a value.

Logical Assignments

Logical assignments are performed identically to character assignments except that periods are looked for instead of quotations. A

check is also made to ensure that the character string between the periods is either 'TRUE' or 'FALSE'. If something else is found an error is called.

Arithmetic Assignments

If an equal sign is found and no quotation mark or period follows it, it is assumed to be an arithmetic assignment statement. Arithmetic assignment statements include both constant values being assigned to variables and mathematical expressions. The evaluations of the arithmetic expression begins by subroutine SEARCH calling subroutine ARITH. A flowchart of subroutine ARITH is shown in Figure 15.

Subroutine ARITH generates a table that represents the mathematical expression to be evaluated. Upon entering subroutine ARITH the table called ARTHTB, standing for arithmetic table, is cleared. The character string to the left of the equal sign (a variable name) is then read and stored in a character variable RESULT. A variety of flags are cleared and the first operator is searched for. The operators that are searched for include: +,-,/,(,),*. If one of these operators is found, it and its priority are placed in array ARTHTB. The priority of the operators is shown below:

(,)	2
**	3
*,/	4
+,-	5
intrinsic function	7
numeric value	9

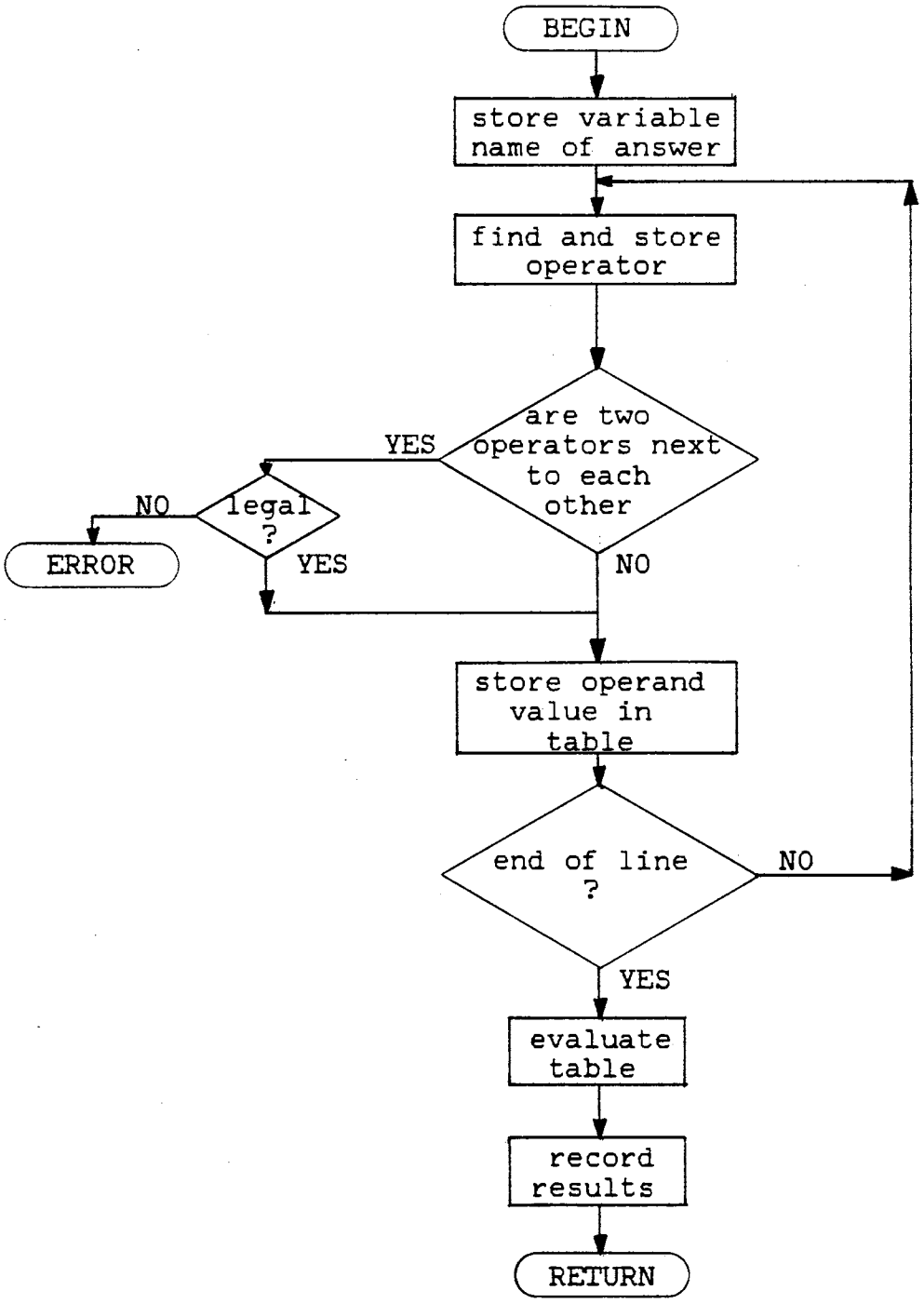


Figure 15. Flowchart of Subroutine ARITH

An example of ARTHTB is shown in Figure 16. ARTHTB is an integer array of 72 rows and 4 columns. Each row represents a location in the arithmetic expression. The first column is the priority of the operator recorded. The second column stores the code for the type of operator. Codes for the operators are as follows:

+	1
-	2
/	3
(	4
)	5
*	6
**	the flag POWFLG

A flag is used in the raising of a power because of the two spaces it occupies rather than the one the other operators occupy. The third column of the ARTHTB tells what type of numeric value is being stored. The codes for this column are:

INTEGER	1
REAL	2
DOUBLE PRECISION	3

Column four holds the integer value that is to be operated on. Two more arrays RARTH and DARTH hold real and double precision values respectively. These arrays are 1 by 72 in size and are used in the same manner as the fourth column of ARTHTB.

If, when an operator is looked for and none is found, it is assumed that the end of the expression has been encountered. The blank following the last operand is found and a flag is set to indicate the

```
7  8  9  10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
A  =  S  Q  R  T  (  B  *  *  2  -  5  .  1  )  /  C
```

WHERE: B=3.2
C=4

	PRORITY	CODE	TYPE	INTEGER VALUE	REAL VALUE	DOUBLE PRECISION VALUE
7	0	0	0	0	0	0
8	0	0	0	0	0	0
9	7	8	0	0	0	0
10	0	0	0	0	0	0
11	0	0	0	0	0	0
12	0	0	0	0	0	0
13	2	4	0	0	0	0
14	9	0	2	0	3.2	0
15	3	0	0	0	0	0
16	0	0	0	0	0	0
17	9	0	1	2	0	0
18	5	2	0	0	0	0
19	9	0	2	0	5.1	0
20	0	0	0	0	0	0
21	0	0	0	0	0	0
22	2	5	0	0	0	0
23	4	3	0	0	0	0
24	9	0	1	4	0	0
25	0	0	0	0	0	0

PRIORITY:

(,)	2
**	3
* , /	4
+ , -	5
INTRINSIC FUNCTION	7
NUMERIC VALUE	9

CODE:

+	1
-	2
/	3
(	4
)	5
*	6
**	7

TYPE:

INTEGER	1
REAL	2
DOUBLE PRECISION	3

Figure 16. Example of ARITH Table

end of the line.

The width between the last operator or equal sign and the newly found operator is measured. If the width is zero, indicating no operand between two operators, syntax checks are made. The only operators that can have another operator right next to it are the parenthesis. Any of the following operators would be illegal before or after the parenthesis:

If the width between the operators is zero and everything is legal, the next operator is located.

Once two operators are found that are spread apart, the operand (character string) is evaluated. If the operator following the operand (character string) is an open parenthesis, the operand must be an intrinsic function (operator) or an array name (operand). It is first checked to see if the character string is an intrinsic function. This is done by calling subroutine FCNAME standing for function name. A flowchart of subroutine FCNAME is shown in Figure 17. The subroutine FCNAME compares the character string with a list of intrinsic names. If a match is found, a seven is recorded in column one of the ARTHTB and the code for the intrinsic function is recorded in column two. The intrinsic function and their codes recognized by INTERP are:

SIN	1
COS	2

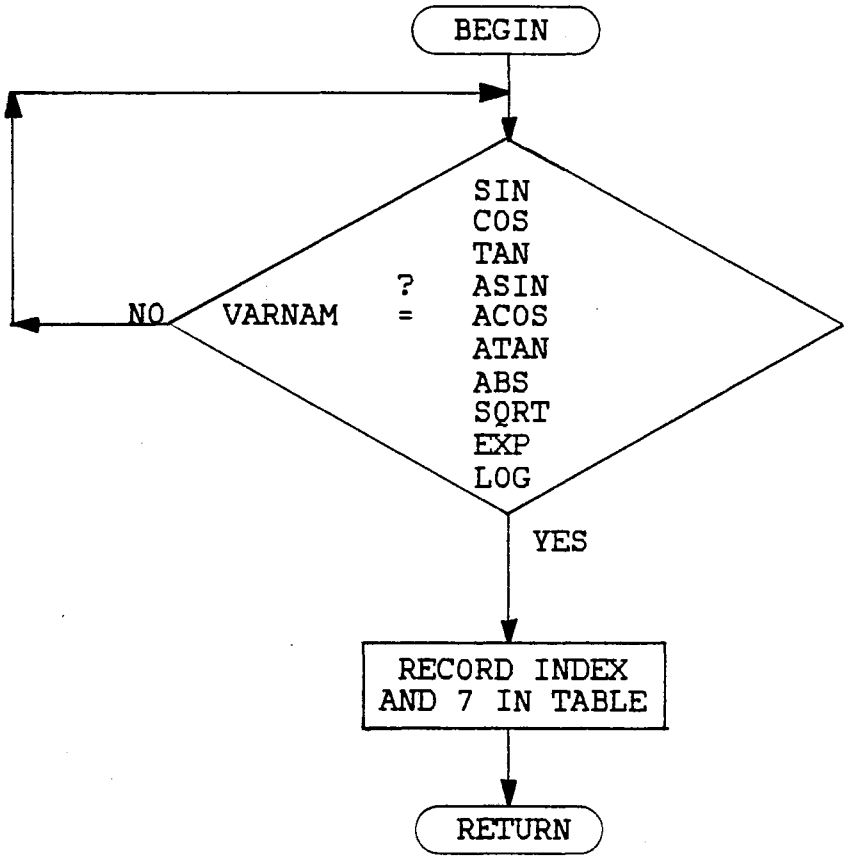


Figure 17. Flowchart of Subroutine FCNAME

TAN	3
ASIN	4
ACOS	5
ATAN	6
ABS	7
SQRT	8
EXP	9
LOG	10

If the character string is an operand and is not an intrinsic function, the close parenthesis is found and the operand along with the contents of the parenthesis is sent to subroutine SARRAY. Subroutine SARRAY returns the value of the array variable sent.

If the operator to the right of the operand is not an open parenthesis, subroutine NUMBER is called. A flowchart of NUMBER is shown in Figure 18. Subroutine NUMBER receives a character string that could be a series of digits, a variable name or an array name with parameters. Subroutine NUMBER first searches for an open parenthesis and if one is found, calls subroutine SARRAY to find its value. This is not needed for the case being discussed but is needed when subroutine NUMBER is called from other parts of the program.

Subroutine NUMBER next checks to see if the character string sent is a string of digits. If it is a string of digits a flag is set and a decimal point is searched for. If a decimal point is found the value of the real number is determined. If no decimal point is found the value of the integer number is determined.

If the character string sent is not a string of digits, subroutine VARABL is called and the character string is sent as a variable name.

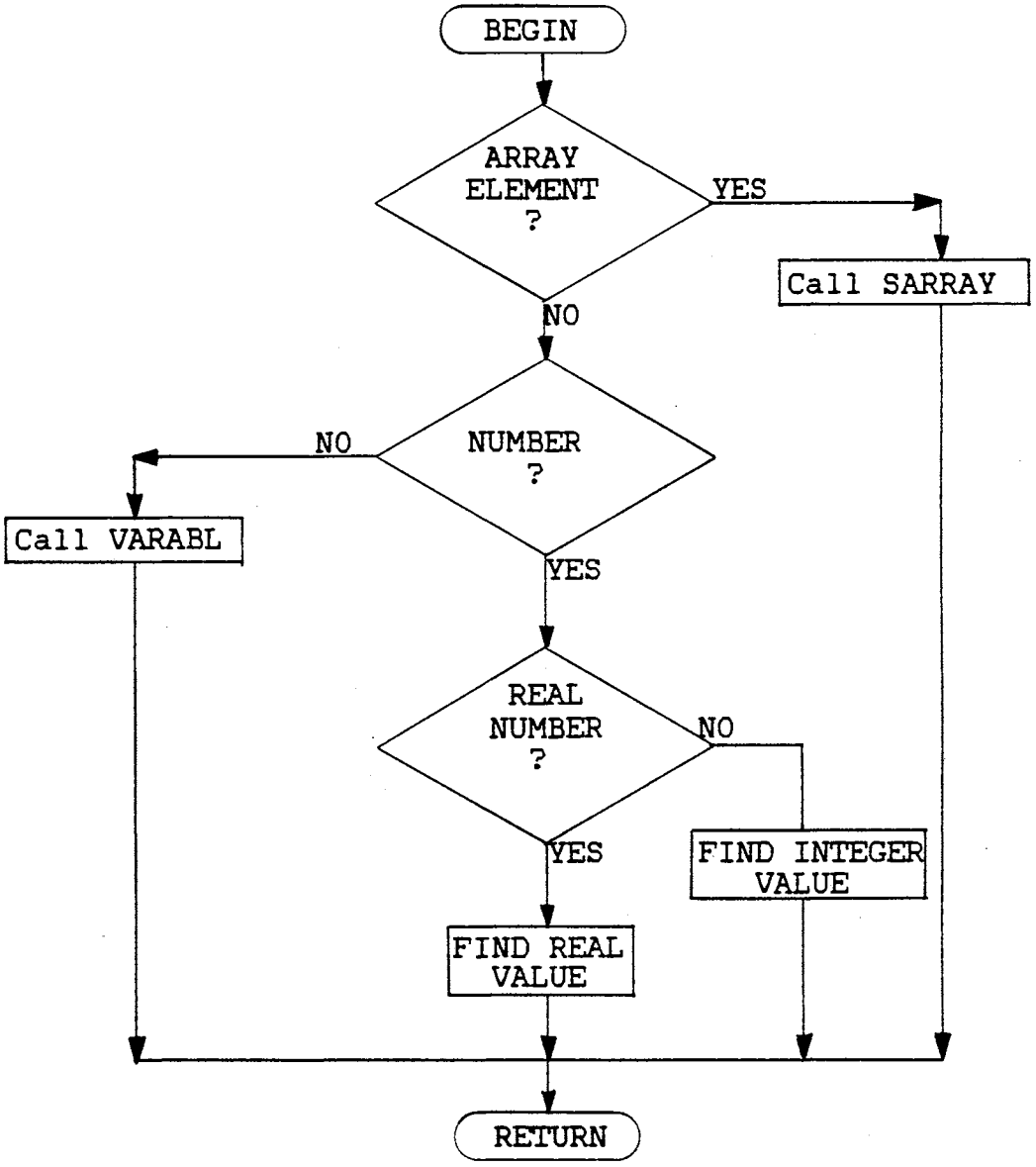


Figure 18. Flowchart of Subroutine NUMBER

Subroutine VARABL returns the value of the variable. This value must be a numeric value.

Once the numeric value of the character string is evaluated, it is returned to subroutine ARITH along with the type of the value. The type of the value is recorded in array ARTHTB 'column three' and the value is recorded in either ARTHTB column four, RARTH or DARTH if its type is integer, real or double precision, respectively. A nine is recorded in ARTHTB column one to indicate a value being stored. If the end of the line has not been reached, the location of the next operator is found. If the end of the line has been reached, the table is evaluated.

A scan is made though the ARTHTB array and a close parenthesis is searched for. If a close parenthesis is found, a bottom pointer is indexed to that location. The scan then searches back up the table for an open parenthesis. When the open parenthesis is found a top pointer is indexed to that location. The portion of the table between the top and bottom pointers is then evaluated.

This evaluation is done by scanning the range of ARTHTB for priority level three. If a priority three is found, subroutine MATH is called. Subroutine MATH performs the seven operations used in mathmetical expressions. A flowchart of subroutine MATH is shown in Figure 19. Subroutine MATH searches above where the priority level three was found for a level nine, indicating the storage of a value. The value and its type is then stored and a search is made below the location of the level three until a priority nine is found. This value and type is also stored. The nines indicating a stored value are changed to zeros, indicating the values have been used.

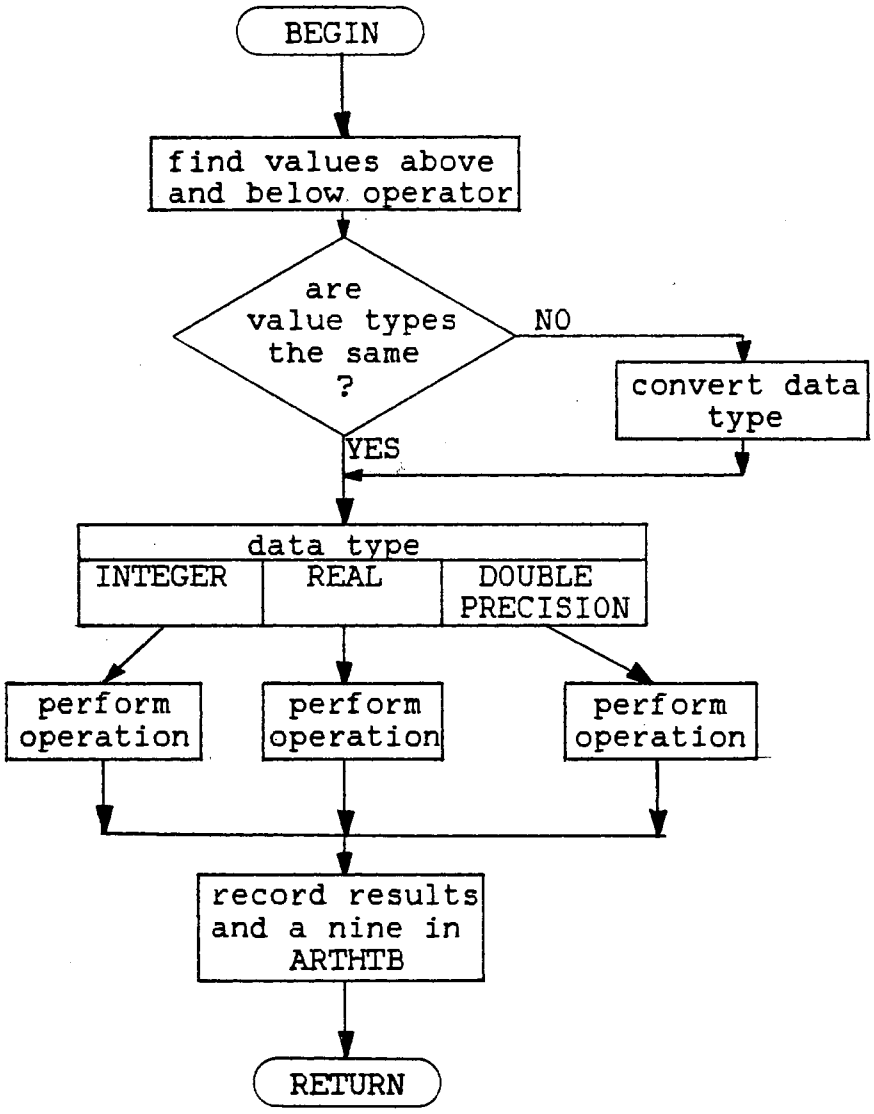


Figure 19. Flowchart of Subroutine MATH

If the types of values found are different, a conversion is made as discussed in Chapter III. Once both values are the same type, a computed GOTO is used to direct the values to the correct operator and the operation is performed. The results of the operation are stored on the row of ARTHTB on which the operator was stored. The priority of this row is reset to nine. Control is then returned to subroutine ARITH.

Subroutine ARITH continues searching for any more items with priority three. If no priority three items are found, priority fours are searched for and then priority five. Any time a priority is encountered, subroutine MATH is called and the evaluation is performed. This procedure observes both the priority of the operators evaluated as well as the order in which the operators appear.

After the range has been evaluated there will be only one value left in the range and all operators will have been eliminated. A check is then made directly before the open parenthesis to see if an intrinsic function is to be performed. If the evaluation of an intrinsic function is needed, subroutine DOFUNC standing for do function, is called. A flowchart of subroutine DOFUNC is shown in Figure 20.

Subroutine DOFUNC begins by searching for the value following it in ARTHTB. Depending on the type of the value, the intrinsic function is evaluated and results in a value of the same type. The only intrinsic function that can be performed on integers is the ABS function. All others result in errors. Other checks are made to ensure valid values before a function is attempted. The resulting value is stored on the same row as the intrinsic function. Control is then returned to ARITH.

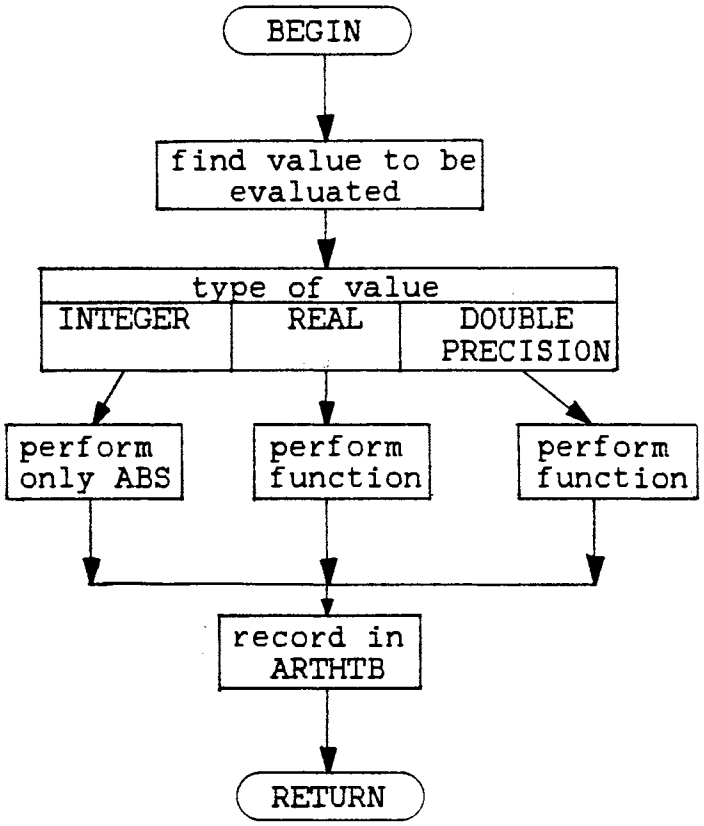


Figure 20. Flowchart of Subroutine DOFUNC

A scan is made further down the table for another set of parentheses and if found, the procedure is repeated. If no parentheses are found, the procedure is repeated with the entire table being the range of the priority scans. This will leave only one value in ARTHTB and all operators will be eliminated.

The character string to the left of the equal sign of the original line of code is checked to see whether or not it is an array. The character string and the resulting value are sent to subroutine SARRAY or VARABL depending on whether or not the variable is an array element. Control is then returned to subroutine SEARCH.

CHAPTER V

CONTROL STATEMENTS

Introduction

Control of a program, written in FORTRAN 77, is maintained by the following statements.

IF Statement

DO Statement

CONTINUE Statement

GOTO Statement

PAUSE Statement

STOP/END Statements

Each of these control statements will be described separately. However, some share the same subroutines and tables.

IF Statement

When an IF statement is encountered in subroutine SEARCH, subroutine SIF is called. The flowchart of subroutine SIF is shown in Figure 21. The line of code and the number one are sent as parameters. The one indicates that the entrance to the subroutine SIF is from an IF statement rather than from an ELSE or ENDIF statement.

Subroutine SIF first searches the line of code for the character string 'THEN'. IF no 'THEN' is found an IF-flag is set indicating a logical IF statement. Parsing is then performed to isolate the logical

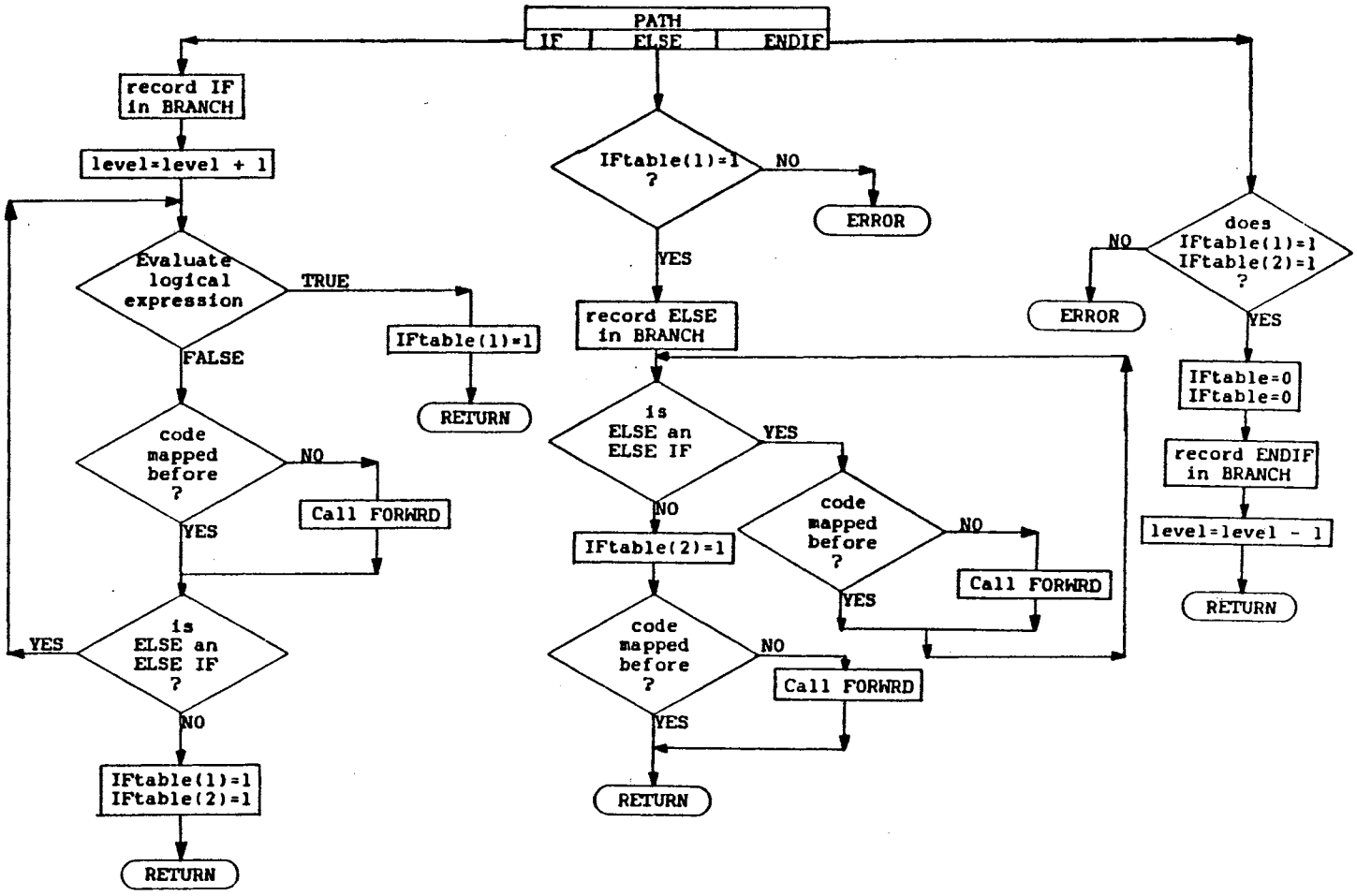


Figure 21. Flowchart of Subroutine SIF

expression and subroutine EVALIF, standing for evaluate IF, is called. The flowchart of subroutine EVALIF is shown in Figure 22. The line of code, the parsed logical expression and a variable to return the value of the logical expression are sent as parameters to subroutine EVALIF.

Subroutine EVALIF parses the logical expression for the first operand. The first period is found and checks are made to see if this is the end of the first operand or a decimal point. If indicated, a second period is searched for. Once the first operand is found it is sent to subroutine NUMBER. Subroutine NUMBER was discussed earlier and will return the value of what was sent whether it was a variable, an array element or a number character string.

The relational operator is then parsed and subroutine OPERAT is called. A flowchart of subroutine OPERAT is shown in Figure 23. The line of code, the parsed relational operator and a variable to indicate which relational operator is found are sent as parameters. Subroutine OPERAT determines which relational operator the character sent represents. The relational operators allowed and their codes are:

1	.LT.	less than
2	.LE.	less than or equal
3	.EQ.	equal
4	.NE.	not equal
5	.GT.	greater than
6	.GE.	greater than or equal

The second operand is then found as was the first. A check is made beyond the second operand to see if this logical expression is in two parts. If a second part is found a double-flag is set.

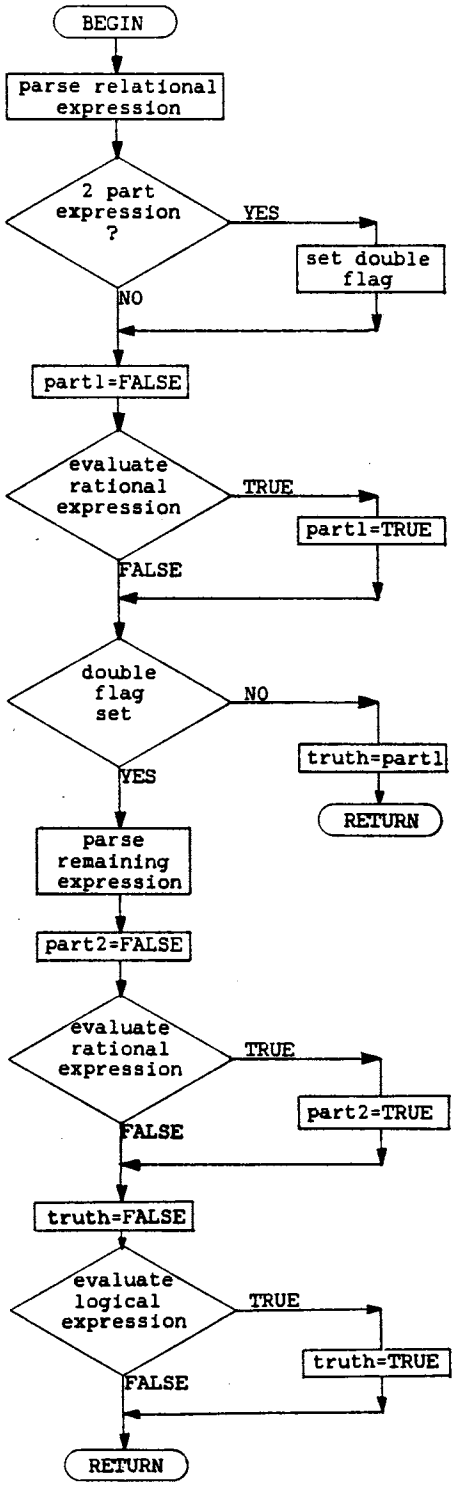


Figure 22. Flowchart of Subroutine EVALIF

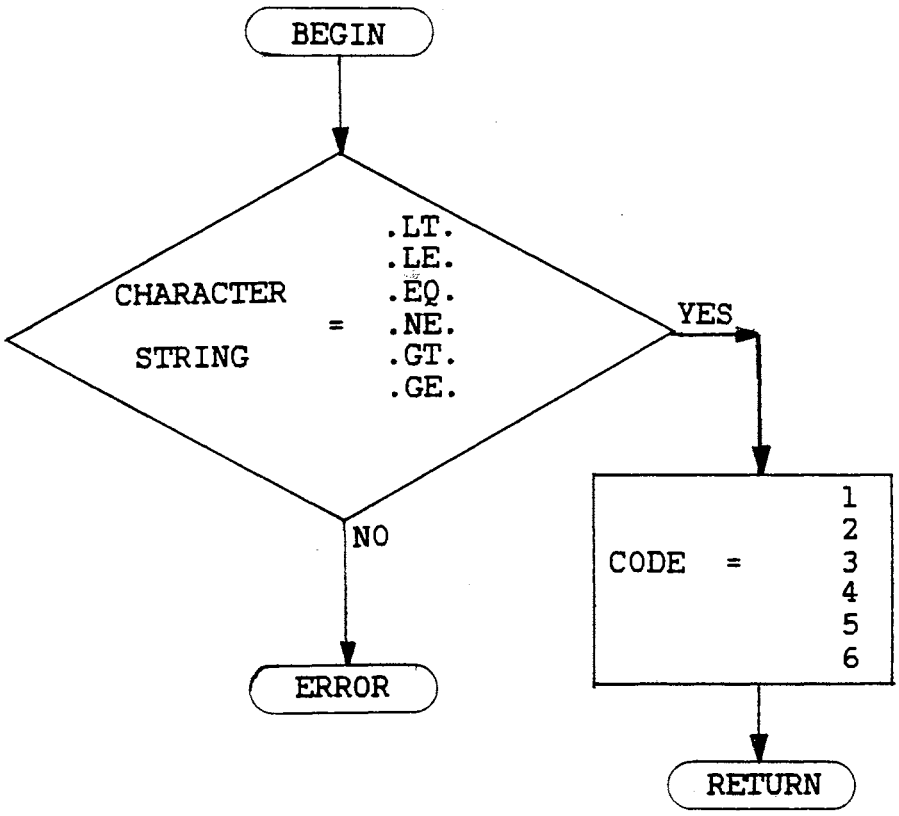


Figure 23. Flowchart of Subroutine OPERAT

An evaluation of the two operands and relational operators is performed. If this is a single logical expression, control is returned to subroutine SIF. If the logical expression is in two parts, the logical operator is parsed. The logical operators allowed and their codes are:

1 .OR.

2 .AND.

The second relational expression is parsed and evaluated exactly as the first. The type of the logical operator is then determined by calling subroutine LOPER standing for logical operator. A flowchart of LOPER is shown in Figure 24. Subroutine LOPER is structured exactly as OPERAT except that it finds the logical operator used rather than the relational operator.

A logical evaluation is performed on the results of the two relational evaluations and a logical value is returned to subroutine SIF.

If the logical expression is false, the IF-flag is turned off and control is returned to subroutine SEARCH, and interpretation of the program continues as normal. If the logical expression is true the statement following the logical expression is stored in the common variable IFLINE and control is returned to subroutine SEARCH. With the IF-flag set, subroutine SEARCH takes IFLINE and interprets it as it would any other line of code. The interpretation then continues normally. The IF-flag is not reset until subroutine SIF is called again.

If, when subroutine SIF looks for the word THEN, and a THEN is found, the IF-flag is not set, indicating an IF THEN ELSE block

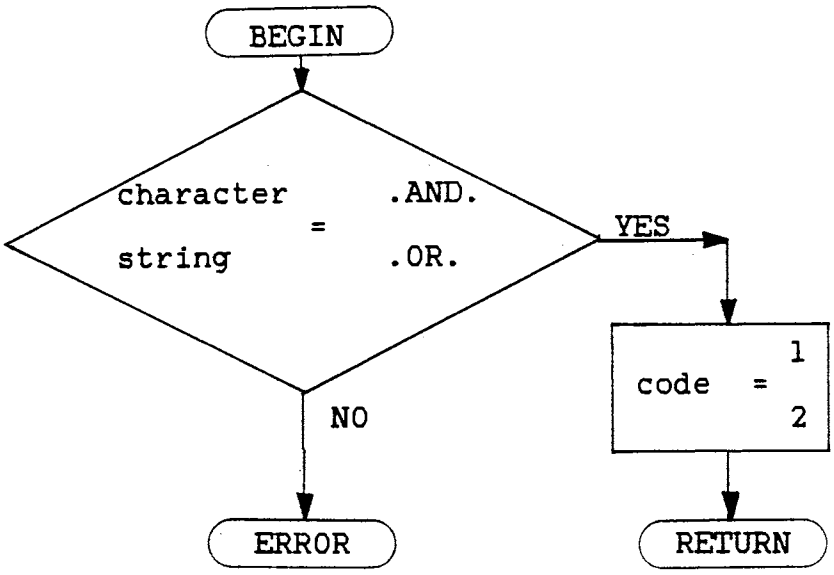


Figure 24. Flowchart of Subroutine LOPER

structure. Since an IF block structure can be nested, the level or depth of the IF block must be accounted for. A map of where the IF statement, ELSE statement and ENDIF statements are must be generated. This is needed to control the branching that must occur as well as detecting errors in illegal branching. Detection of these errors will be considered when the GOTO statement is discussed.

The map that is generated is an integer array table called BRANCH. An example of a BRANCH table is shown in Figure 25. The BRANCH table is used with other control statements such as the DO and GOTO statements. Entries are made in BRANCH whenever there is a statement number, or when an IF,ELSE,ENDIF or DO statement is encountered.

The first column of BRANCH is the user's statement number. Locations of these statement numbers are needed in DO loops and GOTO statements. The second column is the line number of what is being entered. The third column is the type of statement being entered. The codes of statement types are shown below:

0	statement number
1	DO statement
2	DO foot
3	IF statement
4	ELSE statement
5	ENDIF statement

Column four is the statement number of the matching DO-foot of the DO statement being entered. Columns five and six are the levels of the DO and IF structures.

When the IF THEN statement is encountered, the variable level is incremented by one. A check is made to see if the IF THEN statement had

```
                .
                .
                .
1     100   J=3
2           DO 200 HOLD=2.3,6
3              IF(J.LT.HOLD)THEN
4                 GOTO 300
5              ELSE
6                 GOTO 100
7              ENDIF
8     200   CONTINUE
9     300   HOLD=0
                .
                .
                .
```

STRUCTURE CODE:

0	statement number
1	DO statement
2	DO foot
3	IF statement
4	ELSE statement
5	ENDIF statement

BRANCH:

STATEMENT NUMBER	LINE NUMBER	STRUCTURE TYPE	DO FOOT	DO LEVEL	IF LEVEL
0	0	0	0	0	0
100	1	0	200	0	0
0	2	1	0	1	0
0	3	3	0	1	1
0	5	5	0	1	1
0	7	5	0	1	0
200	8	2	0	0	0
300	9	0	0	0	0

Figure 25. Example of BRANCH Table

a statement number that has already been entered in BRANCH. If there was a statement number, the entries for the IF THEN statement would be on the same row. If there was no statement number, entries are made on the next row. The following entries are made:

column 2 line number

column 3 3

column 4 no entry

column 5 same as column 5 in row above

column 6 entry in column 6 in row above +1

The logical expression of the IF statement is then parsed and sent to subroutine EVALIF. Subroutine EVALIF evaluates the logical expression exactly as it did for the logical IF statement.

If the logical expression is true then it is noted that a true expression has been found. This is not necessary in a single IF THEN ELSE structure but in the multiple ELSE IF structure it is necessary to know if a block has been executed or not. This is noted in an array named IFTABL where each row represents a different level of IF structure. This array also notes when the ELSE statement of an IF structure is found. After the true expression has been noted, control is returned to subroutine SEARCH and execution continues normally until an ELSE statement is encountered.

When an ELSE statement is encountered subroutine SEARCH calls subroutine SIF sending the line of code and a number two, indicating the entrance from an ELSE statement. In subroutine SIF a check is made to ensure a block of statements has been executed. This is done by checking array IFTABL. The ELSE statement is then recorded in the BRANCH table. A check is also made to see if the ELSE statement had a

statement number and the following entries are made accordingly:

column 2 line number

column 3 4

column 4 no entry

column 5 same entry as column 5 above

column 6 same entry as column 6 above

The ELSE statement is then parsed to see if it is an ELSE IF structure. If it is an ELSE IF structure, a check is made to see if this code has already been mapped. If it has, a scan is made through the BRANCH table until another ELSE statement of the same level is found. That statement is parsed to see if it is an ELSE IF statement and the procedure continues as above until an ELSE with no IF is found.

If the code had not been mapped before, subroutine FORWRD, standing for forward is called. A flowchart of subroutine FORWRD is shown in Figure 26. Subroutine FORWRD is called when it is necessary to move forward in the lines of code without interpreting the lines. While moving though the code the BRANCH table is built. Statement numbers, DO-foots, DO Statement, IF Statement, ELSE Statement, and ENDIF statements are recorded. When subroutine FORWRD is called a particular line number or type of statement is being searched for. A code is sent to subroutine FORWRD to indicate what is being search for, the codes being:

1 statement number

2 ELSE statement

3 ENDIF statement

The two parameters that are sent to subroutine FORWRD are this code and if applicable, the statement number being sought.

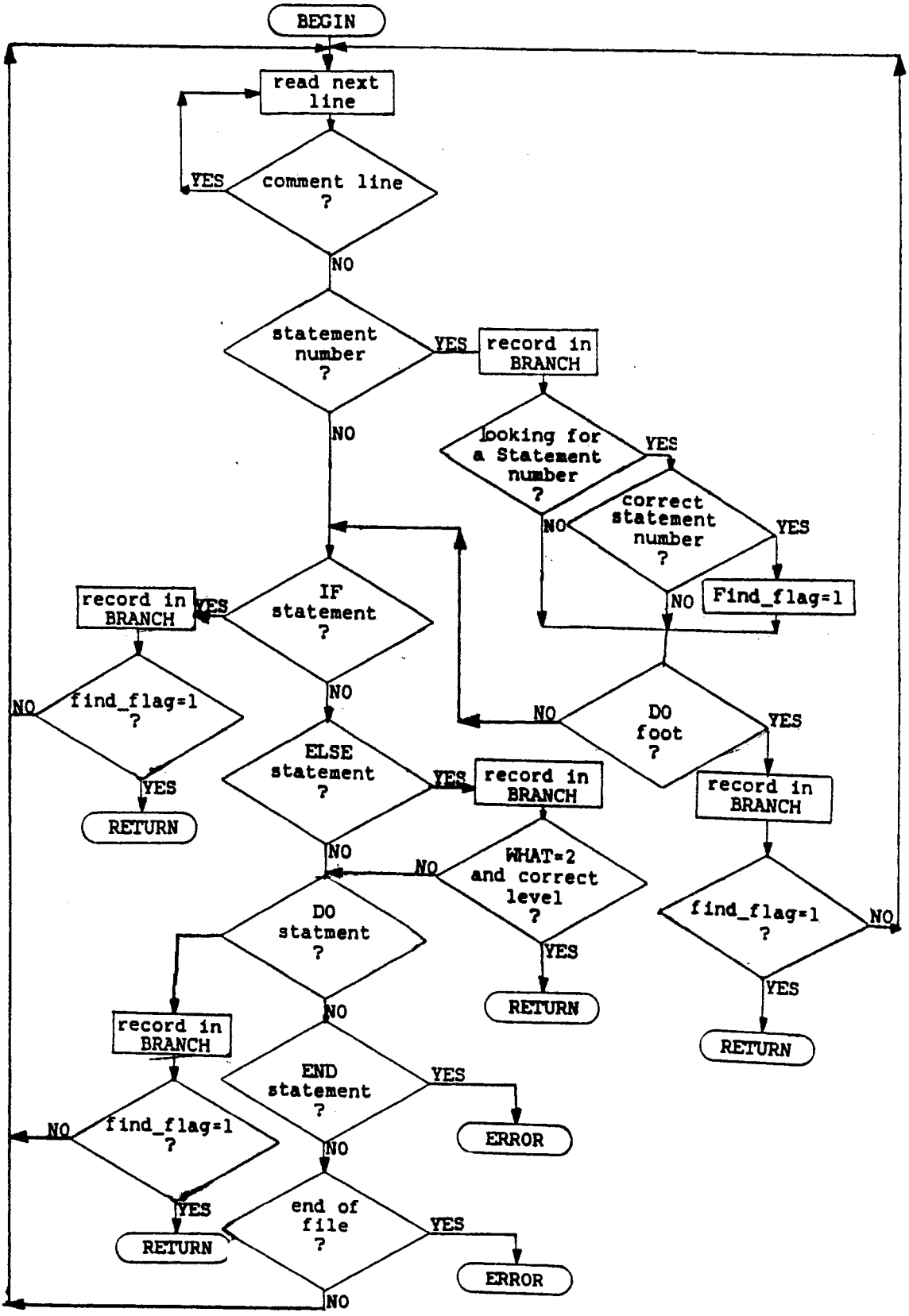


Figure 26. Flowchart of Subroutine FORWRD

After subroutine FORWRD is called and an ELSE statement is found, it is checked to see if it is an ELSE IF structure as was explained above. If it is an ELSE IF structure, subroutine FORWRD is called again and this continues until an ELSE with no IF is found.

When found, this ELSE with no IF is recorded in the array IFTABL to indicate this ELSE was found. The ENDIF statement is then searched for. Again if this code had already been mapped a scan is made though the BRANCH table until an ENDIF is found or subroutine FORWRD is called and the ENDIF statement is searched for. Once the ENDIF statement is found, control is returned to subroutine SEARCH. The ENDIF statement, unlike the ELSE and ELSE IF statements, is executed. The execution of the ENDIF statement will be discussed below.

If, when the logical expression of the IF THEN block was evaluated it was false, a different chain of events would occur. An ELSE statement is searched for in either the BRANCH table or by calling subroutine FORWRD depending upon whether or not the code had been previously mapped. The ELSE statement is checked to see if it is an ELSE IF structure. If it is an ELSE IF structure, the logical expression is evaluated and is handled as if it were a simple IF THEN statement.

If the ELSE statement found is not an ELSE IF, it is noted in the array IFTABL that the ELSE has been found and that a block is about to be executed. Control is returned to subroutine SEARCH and interpretation continues until an ENDIF is encountered.

When an ENDIF statement is encountered, whether it is the ENDIF that was just found by subroutine FORWRD or encountered in line by line interpretation, subroutine SIF is called. The line of code and a number

three are sent as parameters. The three indicates entrance because of the ENDIF statement.

The array IFTABL is checked to ensure that both a block of statements had been executed and that an ELSE statement was found for this level of IF. The IFTABL is then cleared for this level and the ENDIF statement is recorded in the BRANCH table as shown:

column 2	line number
column 3	5
column 4	no entry
column 5	same entry as column 5 above
column 6	same entry as column 6 above - 1

The variable LEVEL is decreased by one and control is returned to subroutine SEARCH.

DO Statement

The DO loop, unlike all other control structures, is handled in both the main program and in subroutine SEARCH. This is necessary to check the DO foot to ensure it is a legal statement, as well as to update parameters and direct control to the top of the loop if necessary. Subroutine SEARCH is not designed to perform these types of functions so they are done in the main program.

The DO statement is first recognized in subroutine SEARCH. Subroutine SEARCH calls subroutine DOBEGN standing for do begin. A flowchart of subroutine DOBEGN is shown in Figure 27. Subroutine DOBEGN begins by parsing the DO statement to find the statement number of the DO foot. The variable name of the DO index is parsed and its initial value is found. A check is made to see if an incremental value is given

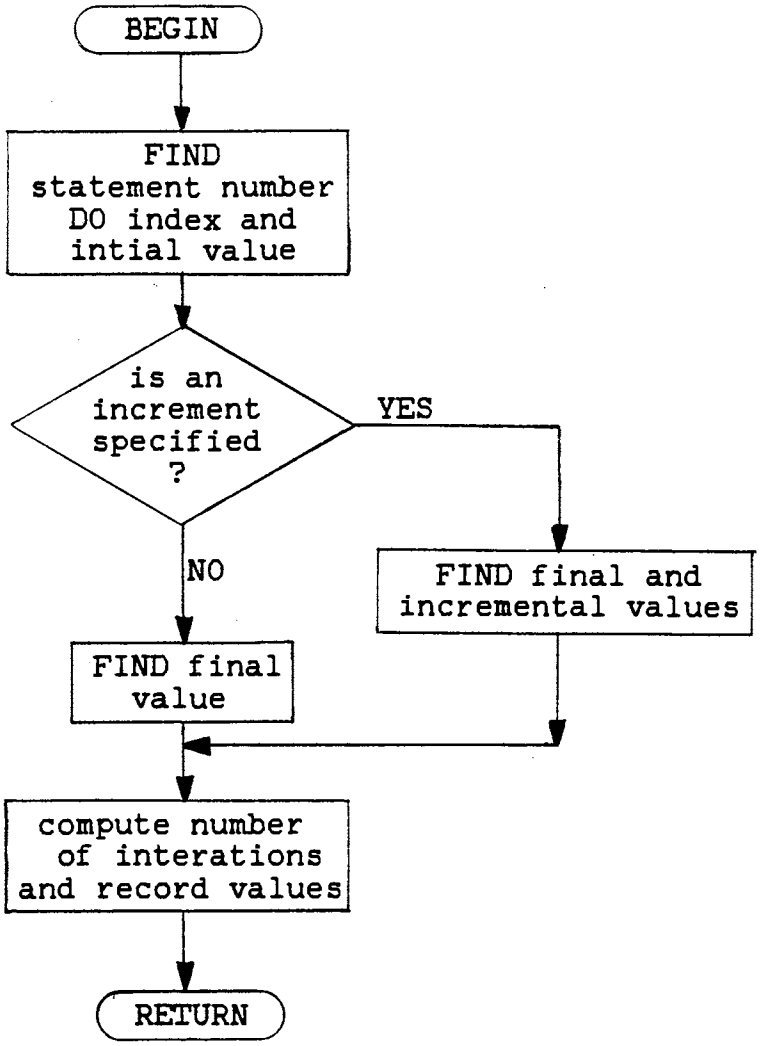


Figure 27. Flowchart of Subroutine DOBEGN

and then the final value and incremental value, if applicable, are found. These values are found with the help of subroutine NUMBER which was discussed in Chapter IV.

The variable name of the DO index is stored in the character array, COUTER. Array COUTER is a one by fifteen array where each storage location holds a string of up to nine characters. The array length of fifteen limits the total number of DO loops allowed in a single program.

The total number of passes through the loop is calculated and stored in a real array. The DO loop parameters can be real numbers and must be stored in a real array. A two by fifteen real array, REALDO holds both the initial and incremental values of the DO loop.

The statement number of the DO foot, the line number of the DO statement and the number of iterations are stored in a three by fifteen array called DO. It is not necessary to save the final values since the iteration value is decremented with each pass through the loop. The loop is finished when this value is zero.

Information on the DO loop is also placed in the array BRANCH. This is done to help with program control and finding illegal branching. A check is made to see if the DO statement has a line number and the following entries are made on the appropriate row of the BRANCH table.

column 2	line number
column 3	1
column 4	Do foot statement number
column 5	same as column 5 above + 1
column 6	same as column 6 above

Control is then returned to subroutine SEARCH and the next line of code is interpreted.

As the program progresses, each time the main program detects a line number, subroutine DCHECK, standing for do check, is called. A flowchart of DCHECK is shown in Figure 28. Subroutine DCHECK, begins by parsing the statement number. It then checks the array DO for a matching statement number. If no match is found, control is returned to the main program and no flags are set. If a match is found and the statement is a DO foot, checks are made to ensure the statement is a legal DO foot. A DO foot cannot be one of the following statements:

GOTO

RETURN

END

STOP

IF

The DO foot is then recorded in the BRANCH table. The following entries are made:

column 2	no entry
column 3	2
column 4	0
column 5	same as column 5 above - 1
column 6	same as column 6 above

The correct structure of combinations of DO and IF structures is then checked. The IF level in the BRANCH table at the DO foot must be equal to zero, this checks for any illegal mixing of the DO and IF structures. The nesting of DO loops is checked to ensure that the DO loop level of the foot (before it was decremented) is the same as the

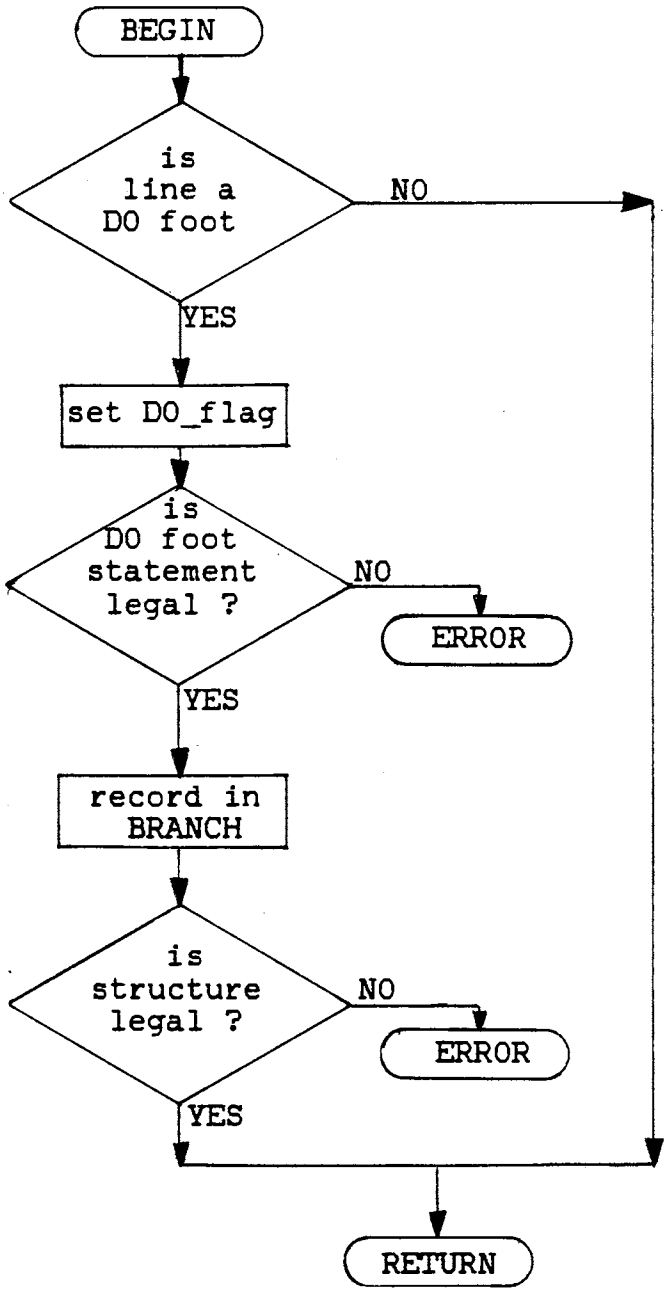


Figure 28. Flowchart of Subroutine DCHECK

DO loop statement. A check is also made to ensure no DO levels lower than the present one exists within that loop. If everything is legal, control is returned to the main program.

The DO foot statement is then executed in a normal manner through subroutine SEARCH. After control is returned to the main program from subroutine SEARCH, a check is made of the DO-flag. If the DO-flag is set subroutine DOFOOT is called. A flowchart of subroutine DOFOOT is shown in Figure 29. Subroutine DOFOOT decrements the iteration counter and checks to see if the DO loop is to be repeated or is complete. If the DO loop is complete, control is returned to the main program. If the loop needs repeating the point of reentry to the program is found. The new incremented value of the DO index is calculated and stored with the help of subroutine VARABL. Control is then returned to the main program.

CONTINUE Statement

The CONTINUE statement, most commonly used as a DO foot, can be used anywhere in a program. When a CONTINUE statement is encountered by subroutine SEARCH, it simply progresses to the next line of code.

GOTO Statement

When the character string 'GOTO' or 'GO TO' is encountered in subroutine SEARCH, subroutine SGOTO is called. A flow chart of SGOTO is shown in Figure 30. Subroutine SGOTO begins by locating where in the BRANCH table it is leaving from.

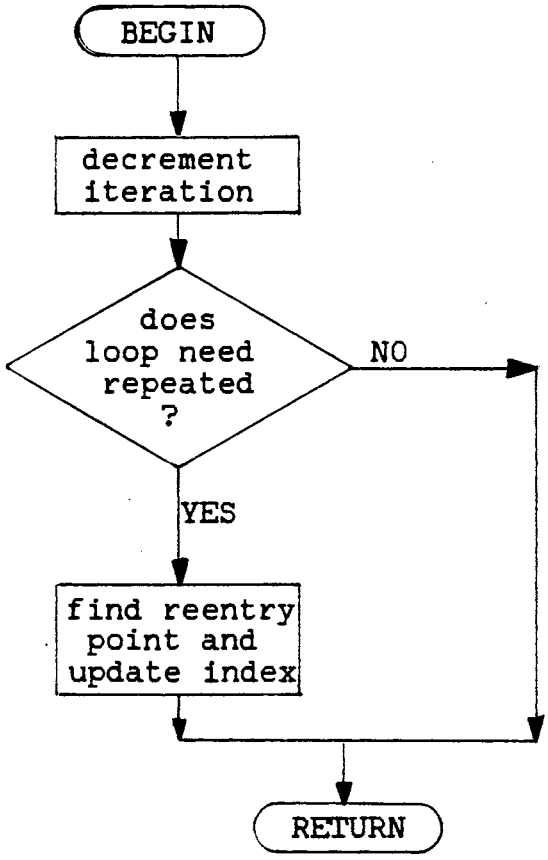


Figure 29. Flowchart of Subroutine DOFOOT

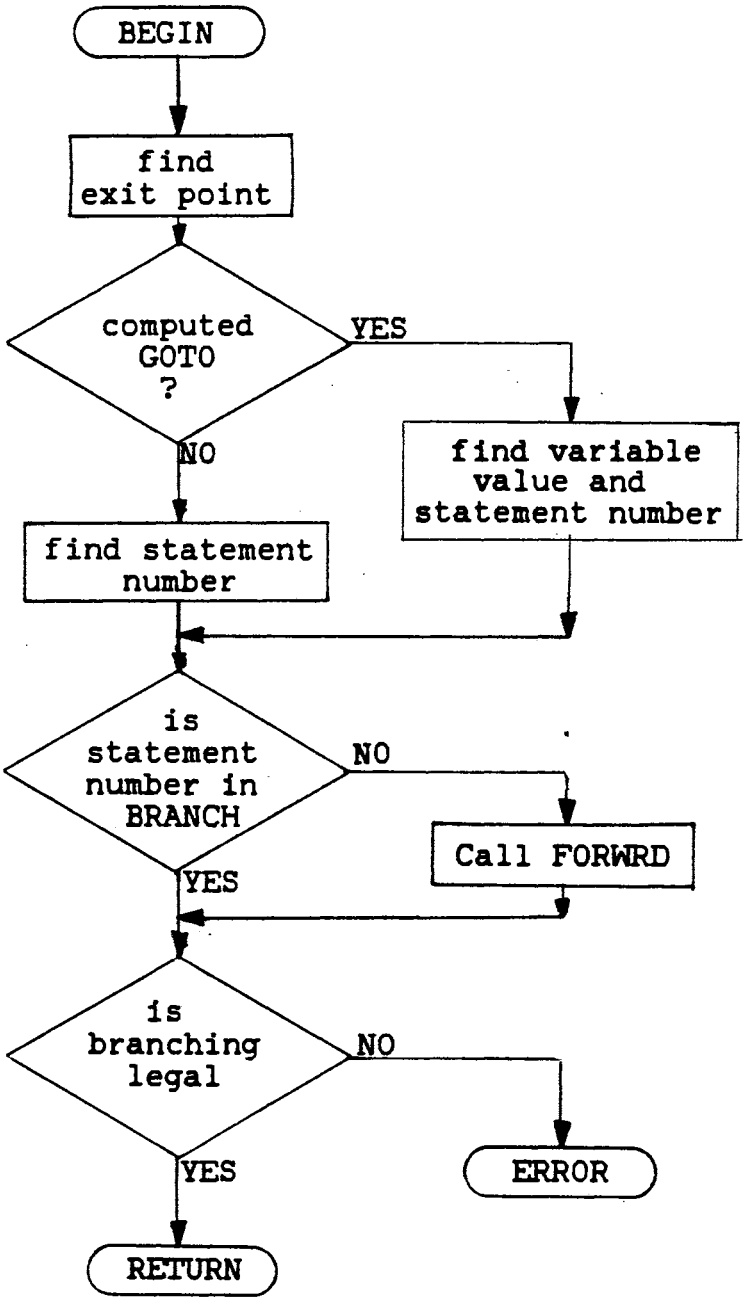


Figure 30. Flowchart of Subroutine SGOTO

The GOTO statement is then checked to see if it is a computed GOTO or an unconditional GOTO statement. If the statement is a computed GOTO, the value of the variable is found and the appropriate statement number is located. Checks are made to ensure that the value of the variable does not exceed the number of statement numbers given.

If the GOTO statement is an unconditional GOTO it is also checked to find the desired statement number.

The statement number is then searched for in the BRANCH table. If it cannot be found, subroutine FORWRD is called. Subroutine FORWRD was discussed in Chapter IV. Once the statement number has been found, checks are performed to ensure the branching is legal.

The checks include making sure the DO and IF levels of the exit point are greater than or equal to the entry point. The DO and IF levels between the entry and exit points also cannot be below the levels at the entry point.

Once all checks are made, control is returned to subroutine SEARCH and then to the main program. Execution of the line specified in the GOTO statement is performed and execution continues normally.

PAUSE Statement

The PAUSE statement is a valuable debugging tool that can be placed anywhere in the program. When the PAUSE statement is encountered, subroutine SEARCH calls subroutine SPAUSE and a menu is displayed. The menu and the options available have been discussed in Chapter II.

STOP/END Statements

The STOP statement is recognized by the interpreter but has no meaning. The END statement is necessary to indicate the end of the program. When an END statement is encountered in subroutine SEARCH an end-flag is set to indicate the end of the program. The main program then advances the user into the PATHWAY State which was discussed in Chapter II.

CHAPTER VI

INPUT/OUTPUT

Introduction

It is not in the concept of the interpreter to provide input/output formating. It was also believed that for a student programmer, the formating techniques used in FORTRAN would be easier to learn once the rest of the language was mastered. For these reasons only free format READ and WRITE statements are allowed.

READ Statement

Interpretation of the READ statement begins with subroutine SEARCH looking for the character string 'READ ' or 'READ('. Once the character string is found, the subroutine SREAD is called with the entire read statement being sent as a parameter. The flowchart for subroutine SREAD is shown in Figure 31.

In subroutine SREAD the beginning of the line is parsed and syntax checks are performed up to the beginning of the variable list. The unit number is also checked to ensure it is an integer value.

Parsing of the variable list begins by finding the left most comma and left most open parenthesis. If the variable being parsed is an array element one of the following will be true:

open parenthesis exists and is more left than the comma, or

open parenthesis exists and there is no comma.

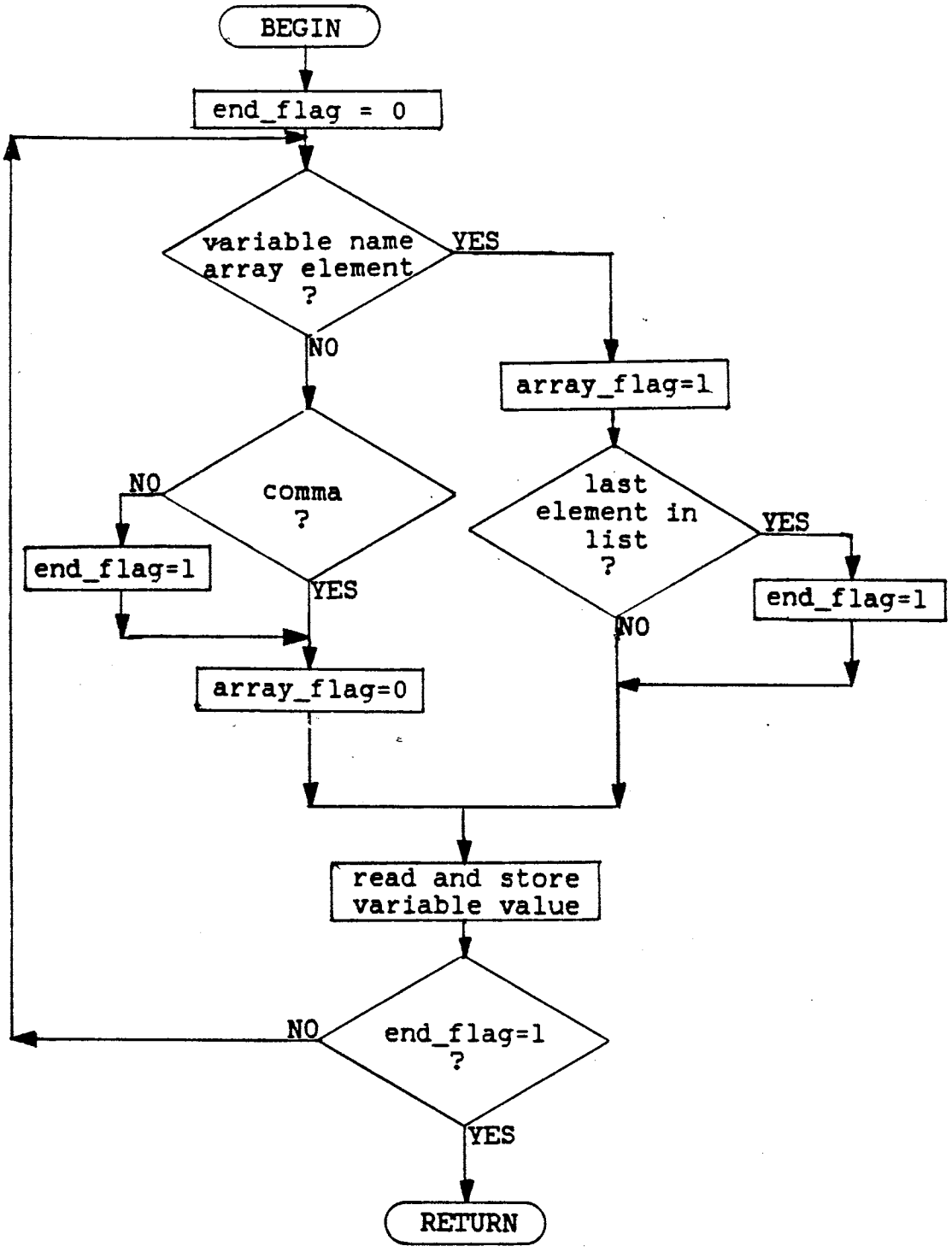


Figure 31. Flowchart of Subroutine SREAD

If the variable is found to be an array the variable name, its subscripts and parentheses are all stored in character variable WORD, and the array flag is set to one. The space immediately following the close parenthesis is checked for a comma or blank. If a blank is found, the end-flag is set to one to indicate the last variable to be read.

If the variable is not an array element, a slightly different procedure is followed. If there was no comma in the variable list, the right most blank is found to determine the length of the variable name. The variable name is stored in character variable WORD and the end-flag is set to one. If there is a comma following the variable name, the variable name is stored in variable WORD. Regardless of the comma status the array flag is set to zero.

Depending on the status of the array flag, either subroutine SARRAY or VARABL is called. The parameter sent to the subroutine is the variable name and the parameter returned is the variable type. Once the variable type has been determined, a computed GOTO is used to read the input into a temporary storage location of the correct type.

Subroutine SARRAY or VARABL (depending upon the array flag) is called a second time. This time the parameters sent are the variable name, type and value to be stored. The subroutine then stores the value under the variable name.

Depending on the status of the end-flag, control will go to the top of the subroutine and begin evaluating the next variable name or control will return to subroutine SEARCH.

WRITE Statement

The interpretation of the WRITE statement begins with subroutine SEARCH looking for the character string 'WRITE ' or 'WRITE('. Once this string is found, the subroutine SWRITE is called with the entire write statement sent as a parameter. The flow chart for subroutine SWRITE is shown in Figure 32.

In subroutine SWRITE the beginning of the line is parsed and syntax checks are performed up to the beginning of the variable list. The unit number is also checked to ensure that it is an integer value.

The parsing of the variable list in a WRITE statement differs from a READ statement since the WRITE statement allows for character strings in the output list. The parsing begins by finding the left most comma and the left most quote. The variable list element is a variable name if one of the following is true:

a comma exists and is more left than the quote, or

no quote exists.

If no comma is found, the end-flag is set to one, indicating the last element in the list. If the element is found to be a variable the same logic is followed as in the read statement to determine if the variable is an array element. If the variable is an array element, subroutine SARRAY is called with the variable name being sent as a parameter and the return of the variable type and the value stored in the variable name. A check is made to see if this element is the last element in the list, since a comma could have been in an array subscript and this element still be the last element. The end-flag is set accordingly.

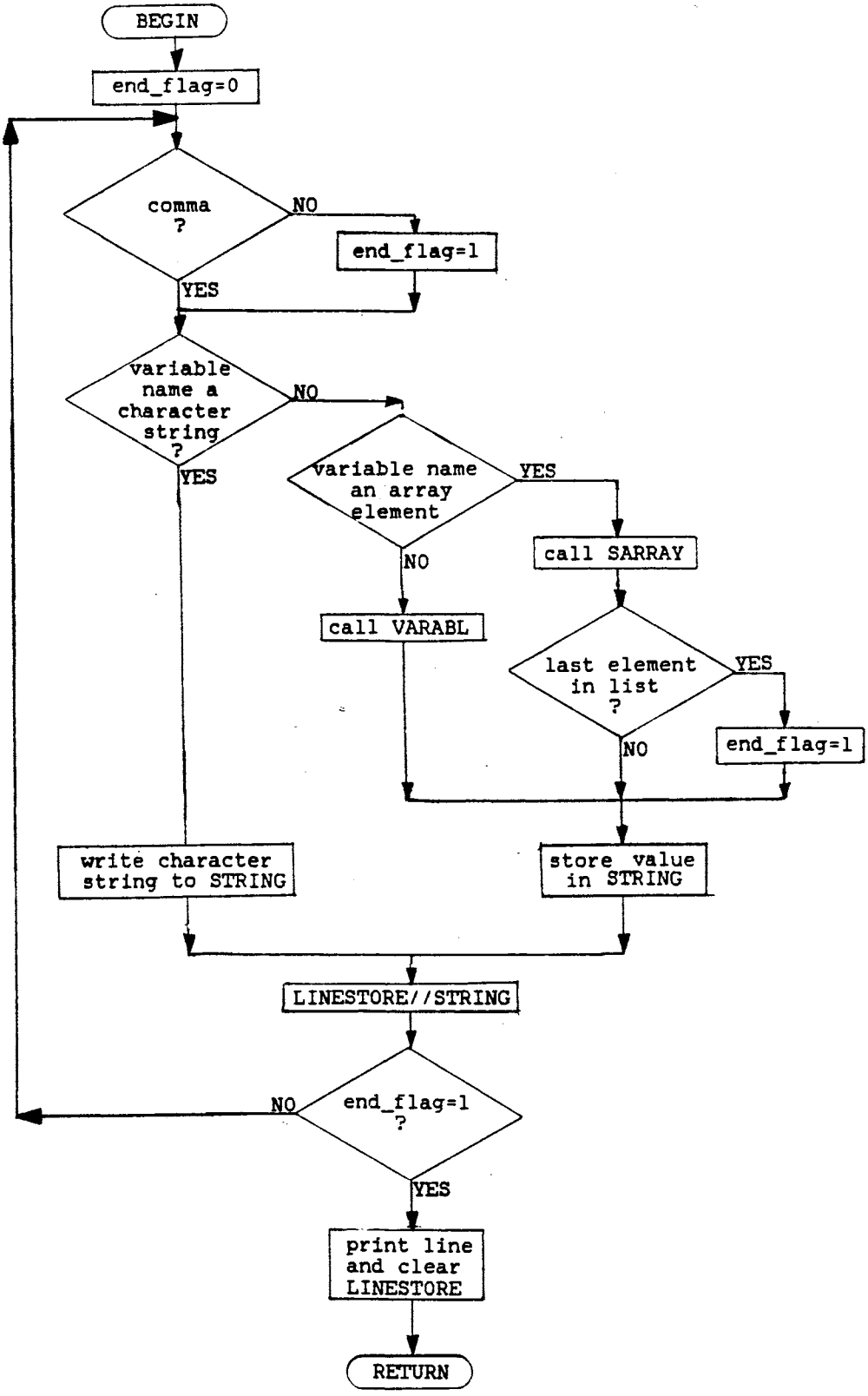


Figure 32. Flowchart of Subroutine SWRITE

If the element was found to be a variable and not an array element, subroutine VARABL is called with the variable name sent and variable type and value returned.

Whether or not the variable is an array, the value is then written to a character string called STRING. This is done with a computed GOTO with the direction being determined by the variable type. This character string is a temporary storage location for the output of the variable elements.

Due to the unknown size of the values to be printed, STRING is large enough to hold up to 80 characters. The values are right justified so the leading blanks are eliminated and the length of STRING can then be determined.

If the element was determined to be a character string, the length is determined and the character string is written to STRING. Regardless of what the element was in the variable list, the value of STRING is concatenated to a character string called LNSTOR, standing for line storage. LNSTOR is the character string that is built by concatenation of the outputs of each element in the variable list. This one character string eliminates the unwanted carriage returns between variable list elements that would otherwise be difficult to avoid.

If at this point the end-flag is equal to one, LNSTOR is printed to the desired device. LNSTOR is then cleared and control is returned to subroutine SEARCH. If the end-flag is equal to zero, control is returned to the top of the program and the next element of the variable list is evaluated.

CHAPTER VII

DISCUSSION AND CONCLUSIONS

Testing of the FORTRAN 77 interpreter, INTERP, has proven that the programming states are very helpful in writing and debugging programs. The testing has also proven to save the user time in not having to recompile after each editing session.

The testing of the interpreter has been performed primarily by the author. Additional testing by student programmers is planned. The author anticipates that some unforseen bugs in the interpreter will be found. First time programmers are likely to write code with errors that the author did not anticipate.

The size of the program is the biggest disadvantage of the interpreter. The program consists of 28 subroutines, approximately 5000 lines of code and uses 338 sectors of storage. It is believed that with additional work the size could be reduced, however, a substantial reduction would not be expected.

Additional work could also be done to increase the size of the FORTRAN 77 subset that is allowed by the interpreter and to allow for the use of expressions throughout the program. These expressions would be useful in places such as the subscripts of an array variable or a DO loop parameter.

It is believed that in time, the FORTRAN 77 interpreter 'INTERP' will prove to be very useful not only here at Oklahoma State University but in may other situations. The need for this program is difficult to overstate.

SELECTED BIBLIOGRAPHY

Acker, Johnny. "Language, Interpeters and Compilers." On Computing Inc., Fall 1981. 47-49.

Davis, Gordon B. and Thomas R. Hoffmann. FORTRAN 77: A Structured, Disciplined Style Based on 1977 American National Standard FORTRAN and Compatible with WATFOR, WATFIV, WATFIV-S, and M77 FORTRAN Compilers. 2nd Ed. New York: McGraw-Hill Book Co., 1983.

Fritzson, Richard. "Write Your Own FORTH Interpreter." Microcomputing., February 1981. 76-92.

Katzan, Harry. FORTRAN 77. New York: VanNorstrand Reinhold, 1978.

PDP-11 FORTRAN Language Reference Manual. Nashua, New Hampshire: Digital Equipment Corporation, 1983.

Reference Manual Harris FORTRAN. Fort Lauderdale, Flordia: Harris Corporation, 1982.

APPENDIXES

APPENDIX A

USER'S MANUAL

GETTING STARTED WITH INTERP

To begin using the Fortran interpreter INTERP you must first have written and stored the program you wish to execute in a file. To begin to execution of your program type INTERP:X and press the enter key. You will be prompted with:

WELCOME TO 'INTERP' YOUR FORTRAN INTERPRETER

PLEASE ENTER NAME OF PROGRAM TO BE INTERPRETED

Type the name of your file and press enter. The screen will be cleared and printed across the top will be:

THE PROGRAM 'filename' IS BEING EXECUTED

Your program is now being interpreted. Any input or output your program has in it will be executed. If no errors are found during the interpretation and the program END statement is reached you will be prompt:

PROGRAM 'filename' HAS BEEN INTERPRETED
(PRESS ENTER TO CONTINUE)

Do not press enter until you are done viewing any output your program has generated. Once the enter key is pressed the screen is cleared and the following menu is displayed:

EXECUTION CHOICE MENU:

1 REXECUTE PROGRAM
2 EXECUTE DIFFERENT PROGRAM
3 QUIT WORKING

PLEASE ENTER THE NUMBER OF YOUR CHOICE

Type either 1,2 or 3 and press the enter key.

REXECUTE PROGRAM: You will be prompted that your program is being interpreted and events will proceed as before.

EXECUTE DIFFERENT PROGRAM: You will be prompted with:

PLEASE ENTER THE NAME OF PROGRAM TO BE INTERPRETED

Type the filename of the program to be interpreted and press the enter key. The prompt that your program is being interpreted will return and interpretation will begin.

QUIT WORKING: If this option is chosen you will no longer be in INTERP. You will be prompted with:

YOU ARE LEAVING 'INTERP' HAVE A NICE DAY

If you want another program interpreted you must renter INTERP by typing INTERP:X.

ERRORS

If while your program is being interpreted an error is detected you will be prompted with:

***** ERROR ***** XXX 'line of code'

ERROR MENU:

1 DISPLAY VARIABLES
2 EDIT TEXT
3 RESTART EXECUTION FROM BEGINNING
4 QUIT OR EXECUTE DIFFERENT PROGRAM

PLEASE ENTER NUMBER OF YOUR CHOICE

The number XXX is the number of the type of error found in the line of code printed to the right. Look in your Table of Error Numbers to find what needs to be corrected. Displaying variables and editing text will be discussed below.

If you think you have corrected the error and wish to rexecute your program type 3 and press enter. You will be prompted that your program is being interpreted. If you can not find what is wrong and you wish to terminate interpretation of this program type 4 and press enter. You will then be given the same options as if your program had been interpreted properly.

DISPLAY VARIABLES

If you wish to see what the current values of your variables are choose the Display Variables option. The following will appear:

VARIABLE MENU:

1 DISPLAY ALL VARIABLES
2 DISPLAY ONE VARIABLE
3 RETURN

PLEASE ENTER THE NUMBER OF YOU CHOICE

If you enter 1 all of your variable names and their values will be displayed. If you enter 2 you will be prompted with:

ENTER THE NAME OF VARIABLE TO BE DISPLAYED

Type the variable name and press enter. The variable name and value will be printed and the Variable Menu will be displayed. If 3 is entered you will return to the previously displayed menu (Error or Pause).

EDIT TEXT

If, from the Error Menu, you chose to edit text the 10 lines closest to the error line will be printed along with the following menu:

```
EDIT MENU

1    DISPLAY DIFFERENT TEXT
2    INSERT A LINE
3    DELETE A LINE
4    CHANGE A LINE
5    QUIT (RETURN TO ERROR MENU

PLEASE ENTER THE NUMBER OF YOU CHOICE
```

When you are in the edit mode (working from the Edit Menu) you can view and change the portion of your program up to and including the line in which the error was found. You will not have access to any text that has not already been interpreted. Any changes you make in your text are also made in your program file and will be there when you leave INTERP.

DISPLAY DIFFERENT TEXT: This option will prompt you with:

ENTER LINE NUMBER YOU WISH TO BEGIN DISPLAY

Once you have entered the number, that line and the nine lines following it will be displayed.

INSERT A LINE: This option will prompt you with:

ENTER LINE NUMBER YOU WISH TO INSERT AFTER

Once you have entered the number you are prompted:

ENTER TEXT OF LINE TO BE INSERTED

Type the line you want inserted. This line entered must have the same format as when you wrote your program, including spacing over seven spaces before beginning the statement (unless you want a statement number). Once the line is complete press the enter key. The 10 lines around the line entered will be displayed. You cannot enter more then one line at a time.

DELETE A LINE: This option will prompt you with:

ENTER LINE NUMBER YOU WISH DELETED

Enter the number and press enter. The 10 lines around the deleted line will be displayed.

CHANGE A LINE: This option prompts you with:

ENTER LINE NUMBER OF THE LINE YOU WISH TO CHANGE

Once the number has been entered you are prompted:

ENTER LINE AS DESIRED

You must type the entire line using the same format as the original line. When the line you type is complete press the enter key. The 10 lines around the line changed will be displayed.

QUIT: If you have edited all you want to and you chose to quit, the Error Menu will be redisplayed.

PAUSE

The pause statement can be used anywhere in your program. As your progrm is being interpreted each line is read and executed. When a pause statement is found the interpretaion of your porgram is stopped and the following menu is shown:

PAUSE MENU:

1 DISPLAY VARIABLE VALUES
2 CHANGE VARIABLE VALUES
3 CONTINUE EXECUTION FROM THIS POINT
4 START EXECUTION FROM BEGINNING OF PROGRAM
5 QUIT

PLEASE ENTER THE NUMBER OF YOUR CHOICE

DISPLAY VARIABLE VALUES: This option has been discussed and behaves as it did in the Error Menu.

CHANGE VARIABLE VALUES: If you wish to change a variable value you will be prompted:

ENTER THE NAME OF VARIABLE YOU WISH TO CHANGE

Once the variable name is entered the variable name and its value will be printed and you will be prompted with:

PLEASE ENTER THE NEW VALUE FOR 'variable name'

Once the value is entered you will be prompted:

DO YOU WANT 'variable name' = 'new variable value' (YES/NO)

If the new value is the value you want, enter YES. If you entered the wrong value enter NO and you will be asked to enter a new value. Once you have accepted a value entered the Pause Menu will be redisplayed.

CONTINUE EXECUTION: If you want to continue with the interpretation of your program enter 3 and interpretation will resume.

START EXECUTION FROM BEGINNING: If you chose this option, interpretation will begin again with your first line of code and continue through your program.

QUIT: This option allows you to stop interpretatin of your program but does not kick you out of INTERP. The Execution Menu is displayed.

DATA TYPES AND LIMITATIONS

Components of an expressions can be any of the following:

CONSTANT A fixed value such as a number

VARIABLE Symbolic name that represents a stored value, This value can be changed.

ARRAY A group of values stored under one symbolic name. Each stored value is referred to by the subscripts of the symbolic mane.

A symbolic name is a string of characters (letters or numbers) totaling a maximum of six. THe first character must be a letter.

EXAMPLES OF SYMBOLIC NAMES

Valid	Invalid	
SUM	5TRY	first character must be a letter
TOTAL7	AC#7	character not a letter or number

DATA TYPES: Each of the components above will be of a particuliar data type. The data types available are:

INTEGER	whole numbers
REAL	decimal numbers
DOUBLE PRECISION	real numbers with twice as many significant digits as a normal real
CHARACTER	a sequence of characters
LOGICAL	has a value of true or false

EXAMPLES OF DATA TYPES

12	INTEGER
170.762	REAL
3.1476543D+2	DOUBLE PRECISION
APRIL	CHARACTER (up to 60 characters)
.TRUE.	LOGICAL (must include periods)

To assign a data type to a symbolic name, the type statements are used. The syntax of these statements are:

Datatype symbolicname [,symbolicname]

EXAMPLES OF TYPE STATEMENTS

```
INTEGER SUM,TOTAL
REAL ROOT6
CHARACTER MONTHS,WEEKDY
LOGICAL FLAG
```

Every symbolic name that is going to be used in a program must be declared in a type statement. A maximum of 20 symbolic names are allowed for each data type.

ARRAYS: To specify the size of the group of values stored under an array symbolic name a dimension statement is needed. An array can be thought of as a group of boxes (storage locations). These boxes can either be a tall stack, a one dimensional array or a wall of boxes, a two dimensional array.

1
2
3
4

one dimensional array
A(4)

1,1	1,2	1,3
2,1	2,2	2,3
3,1	3,2	3,3
4,1	4,2	4,3

two dimensional array
B(4,3)

The first subscript indicates how tall the array is. The second subscript (if it is a two dimensional array) indicates how wide the array is. Before an array can be dimensioned the symbolic name must be declared.

EXAMPLES OF DIMENSIONING ARRAYS

```
INTEGER A
REAL B
DIMENSION A(4),B(4,3)
```

In this example, you have allocated four integer storage locations in array A and twelve real storage locations in array B.

Of the twenty symbolic names allowed in the data types integer,real,and double precision, ten of them can be dimensioned to arrays. The maximum array size allowed is 100 by 100. In the character and logical data types only five symbolic names can be dimensioned to arrays. These arrays are limited to one dimensional arrays with a maximum size of 10.

INPUT/OUTPUT

Whenever it is necessary to put information into the computer or pull information out, you need to use input/output statements. These I/O statements are the READ and WRITE statements as explained below.

READ

READ(unum,*)[varname list]

READ(3,*)SUM,TAB(3),X

Where:

unum -- is the number of the unit device to be read from. This number must be an integer

varname --is a list of variable names that the information read is to be stored in . Each name must be seperated by a comma (SUM,TAB and X are previously declared variables).

* -- The star represents a free format. Free format is the only format allowed by INTERP.

In the above example the computer would go to device unit number 3 (say the keyboard) and wait for three (the number of names in variable list) separate values to be entered before continuing exection.

WRITE

WRITE(unum,*)[varname list]

WRITE(3,*)'THE VALUE OF SUM=',SUM,'I=',I

Where:

unum -- is the number of the unit device to be written to. This number must be an integer.

varname list -- is a list of variable names and/or character strings who's value are to be the output. Each element(variable name or character string) in the list are separated by a comma. Each character string must be enclosed with in quotes. (SUM and I are previously declared variables) If the variable list is omitted a blank line is written.

* -- The star represents a free format. Free format is the only format allowed by INTRERP.

In the above example the computer will go to device unit number 3 (say the CRT) and write the first character string followed by the value of SUM, then the second

character string followed by the value of I. The output of the example above could be:

```
SUM= 13  I= 1.375
```

ASSIGNMENT STATEMENTS

There are three types of assignment statements, they are:

CHARACTER

LOGICAL and

ARITHMETIC.

CHARACTER ASSIGNMENT STATEMENTS: This statement is used to assign a string of, up to 60 characters, to a variable name. This string must be bracketed by quotation marks.

EXAMPLE OF CHARACTER ASSIGNMENT

```
MONTHS='APRIL'
```

The character string APRIL is assigned to character variable MONTHS.

LOGICAL ASSIGNMENT STATEMENT: This statement is used to assign a logical value of TRUE or FALSE to a variable. To make this assignment a variable name is set equal to the character string .TRUE. or .FALSE. This character string must include the periods.

EXAMPLE OF LOGICAL ASSIGNMENT

```
FLAG=.TRUE.
```

A value of TRUE is assigned to the logical variable FLAG.

ARITHMETIC ASSIGNMENT STATEMENT: An arithmetic assignment can either be a value being assigned to a variable name or an arithmetic expression being evaluated and the results being assigned to a variable name.

If the data types of the variable name and the value to be assigned differ, the value is converted to the same data type as the variable name.

EXAMPLES OF MIXED MODE DATA TYPES

```
I=3.76     I => INTEGER  I=3
A=2        A => REAL     A=2.0
```

In an arithmetic expression the following symbols are recognized and are performed in the following order.

(,)	parenthesis
**	raised to a power
* , /	multiply and divide
+ , -	add and subtract

If two operators on the same line appear in an expression, they are performed in the order at which they appear.

EXAMPLE

X=A+B/C*D is equal to X=A+((B/C)*D)

There are certain intrinsic functions available for uses in arithmetic expressions. They include:

SIN, COS, TAN, ASIN, ACOS, ATAN	Trigonometric Functions
ABS	Absolute Value
SQRT	Square Root
EXP	exponential
LOG	logarithm

These intrinsic functions are used by placing the value to be evaluated in parentheses following the function name. The value in the parentheses can be a single value or a result of an expression.

EXAMPLES OF INTRINSIC FUNCTION USES

```
X=SIN(PI)
Y=SQRT(A**2+B**2)
```

CONTROL STATEMENTS

The control and branching that occur in a program is performed by the control statements listed below:

IF

DO

CONTINUE

GOTO

STOP/END

IF STATEMENT: There are two types of IF statements, the logical IF and the block IF. They both evaluate a logical expression then take appropriate action. The relational and logical operators available are:

RELATIONAL OPERATORS:

.LT.	LESS THAN
.LE.	LESS THAN OR EQUAL
.EQ.	EQUAL
.NE.	NOT EQUAL
.GT.	GREATER THAN
.GE.	GREATER THAN OR EQUAL

LOGICAL OPERATORS:

.AND.
.OR.

The logical expressions may contain up to two relational operators and one logical operator. The operands in the expression may be numbers or variables.

EXAMPLE OF LOGICAL EXPRESSION

```
(A.LT.5.AND.A.GE.2)

(A.EQ.B)
```

Each of these expressions would be evaluated to TRUE or FALSE. In the logical IF, the statement following the logical expression would be executed only if the logical expression is TRUE.

EXAMPLE OF LOGICAL IF

```
IF(A.EQ.B)B=7
```

If A is equal to B, B would then be assigned the value 7. If A is not equal to B, the value of B would remain unchanged.

In the block IF when the logical expression is evaluated it determines if a block of statements will be executed or not.

EXAMPLE OF BLOCK IF

```
IF(A.EQ.B)THEN
    BLOCK1
ELSE
    BLOCK2
ENDIF
```

If A is equal to B, all the statements in BLOCK1 will be executed and none in BLOCK2. If A is not equal to B no statements in BLOCK1 will be executed but all the statements in BLOCK2 will be executed.

The IF-block can also be structured in an ELSE IF THEN manner.

EXAMPLE OF ELSE IF THEN

```
IF(A.EQ.B)THEN
   BLOCK1
ELSE IF(A.LT.B)THEN
   BLOCK2
ELSE IF(C.GT.D)THEN
   BLOCK3
ELSE
   BLOCK4
ENDIF
```

The block of statements following the first TRUE logical expression will be executed. If all logical expressions are FALSE the block of statements following the ELSE will be executed. Only one block of statements will be executed.

DO STATEMENT: The DO statement provides a way to repeat the same lines of code.

EXAMPLE OF DO STATEMENT

```
DO 100 I=A,B,C
```

The DO statement initates the DO loop. All statements following the DO statement will be executed up to and including the statement number 100. Statement number 100 cannot be a GOTO, DO, RETURN, END or STOP statement. The variable I is given the value of A and the loop is executed. Once statement number 100 is reached, the value of C is added to the value of I and the loop is executed again. If C is omitted, it is assumed to be one. The loop continues execution until the value of I is equal to the value of B.

Do loops can also be nested, by placing one loop inside another. The maximum depth of nesting is five.

EXAMPLE OF DO LOOP

```
      DO 100 J=1.0,3.2,0.2
           X=X+1
           Y=X+J
100   CONTINUE
```

This loop would be executed 11 times.

EXAMPLE OF NESTED DO LOOP

```
      DO 200 I=1,2
          DO 100 J=1,3
          WRITE(3,8)I,J
100       CONTINUE
200   CONTINUE
```

The output of this DO loop would be :

```
      11
      12
      13
      21
      22
      23
```

The outer loop would executed twice and the inner loop would execute three times in each pass for a total of six times.

CONTINUE STATEMENT: The CONTINUE statement is not an executable statement. The CONTINUE statement is used to help make a program more readable. The most common use of this statement is at the bottom of a DO loop. However, it can be used anywhere.

GOTO STATEMENT: The GOTO statement is used to branch to a different part of the program. There are two types of GOTO statements, the unconditional GOTO and the computed GOTO.

EXAMPLE OF UNCONDITIONAL GOTO

```
      GOTO 120
```

This GOTO statement directs the program to continue execution at statement number 120.

EXAMPLE OF COMPUTED GOTO

```
GOTO(100,120,130,140)SUM
```

This GOTO statement directs the program to continue execution from one of the statement numbers listed. Which of the statement numbers is determined by the value of the variable SUM. If SUM=1 execution goes to statement 100 if SUM=2 execution goes to statement 120 and so on. The value of SUM cannot exceed the number of statement numbers listed.

END OR STOP STATEMENT: A STOP or END statement must be located at the end of a program. This indicates the end of the program and tells the interpreter to stop execution of the program and to return to the PATHWAY state.

APPENDIX B

PROGRAM LISTING

```
C*****************************   PROGRAM INTERP  **********************
C
C
C       PURPOSE:        THIS IS THE MAIN PROGRAM OF THE INTERRETER
C                       IT CLEARS TABLES AN REGISTERS TO START NEW
C                       INTERPRETATION, READS THE USERS PROGRAM-LINE
C                       BY LINE, CHECKS FOR COMMENT LINES AND STATEMENT
C                       NUMBERS, RUNS CHECKS ON DO LOOPS AND CALLS
C                       SUBROUTINE SEARCH TO LOOK FOR KEY WORDS
C
C
C       PARAMETERS:
C
C               USED FOR CONTROL:
C
C                       COMENT: VARIABLE USED TO CHECK FOR COMMENT
C                               LINES
C                       ENDFLG: FLAG THAT IS SET WHEN END OF USER'S
C                               PROGRAM IS REACHED
C                       LINFLG: FLAG THAT IS SET IF THE LINE HAS A
C                               STATEMENT NUMBER
C                       DOFLG:  FLAG THAT IS SET IF LINE IS A DO FOOT
C                       ENTRY:  THE LINE NUMBER OF THE NEXT LINE OF
C                               CODE TO BE INTERPRETED- USED WITH DO
C                               AND GOTO STATEMENTS
C                       FILNAM: THE NAME OF FILE TO BE INTERPRETED
C
C               FOR PARSING:
C
C                       BLANK:  INTEGER LOCATION OF A BLANK
C                       TYPE:   DATA TYPE OF NUMBER
C                       NUM:    STATEMENT NUMBER FOUND
C                       EXPRES: LINE OF CODE BEING PARSED
C                       WORD:   CHARACTER STRING OF STATEMENT NUMBER
C                       IVAL,R,D,C,L:  PATHWAYS TO PAST VALUES OF
C                               INTEGER,REAL, DOUBLE PRECISION,
C                               CHARACTER AND LOGICAL VALUES TO AND
C                               FROM SUBROUTINES
C
C               COMMON PARAMETERS:
C
C                       SETFLG: FLAG WHEN SET INDICATES THE NEED TO
C                               CLEAR ALL TABLES AND REGISTERS
C                       RSTART: A VARIABLE THATS VALUE INDUCATES
C                               WHICH DIRECTION VMAIN IS TO TAKE:
C                                   1:  CONTINUE EXECUTION
C                                   2:  RESTART EXECUTION FROM START
C                                   3:  EXECUTE DIFFERENT PROGRAM
C                                   4:  STOP
C
C                       SCLINE: CHARACTER ARRAY THAT HOLDS THE LINES
C                               OF CODE READ
C                       CLINE,LINE:  THE LINE OF CODE BEING INTERPRETED
C                       LINDEX: THE LINE NUMBER OF THE MOST ADVANCED
C                               LINE THAT HAS BEEN INTERPRETED
C                       LPOINT: THE LINE NUMBER OF THE LINE BEING
C                               INTERPRETED
C                       LEVEL:  THE LEVEL OF IF NESTING
C                       MAXLIN: THE MAXIMUM NUMBER OF LINES ALLOWED
C                               IN THE USERS PROGRAM
C                       DO:     INTEGER ARRAY THAT HOLDS DO-FOOT
C                               STATEMENTS NUMBERS AND LINE NUMBERS
C                               AND THE NUMBER OF ITERATIONS LEFT
C                       REALDO: REAL ARRAY OF THE INITIAL AND
C                               INCREMENTAL VALUES OF THE DO PARAMETERS
C                       DINDEX: NUMBER OF ROW LAST ENTRY TO DO AND
C                               REALDO WAS MADE
C                       FINDEX: NUMBER OF THE ROW THAT THE INFO RELATING
C                               TO A PARTICULIAR DOFOOT IS IN
C
C
```

```
C       ERROR NUMBERS CALLED:                                                   72
C                       303                                                     73
C                       406                                                     74
C                       407                                                     75
C                                                                               76
C       SUBROUTINES CALLED:                                                     77
C                       SARRAY                                                  78
C                       VARABL                                                  79
C                       NUMBER                                                  80
C                       ERROR                                                   81
C                       SEARCH                                                  82
C                       MAP                                                     83
C                       DOBEGN                                                  84
C                       DCHECK                                                  85
C                       DFOOT                                                   86
C                       PATHWY                                                  87
C                                                                               88
C*************************************************************************      89
        PROGRAM VMAIN                                                           90
        INTEGER COMENT,ENDFLG,BUG,BLANK,LINFLG,DOFLG,ENTRY,TYPE,                91
     $          CLNUM,SETFLG,LPOINT,IVAL,NUM,FINDEX,DINDEX,DO,                  92
     $          LINDEX,LEVEL,RSTART,MAXLIN                                      93
        REAL R,REALDO                                                           94
        CHARACTER C*60,SCLINE*72,FILNAM*8,EXPRES*72,CLINE*72,WORD*10,           95
     $          CONTER*9,LINE*72,DUMMY*1                                        96
        DOUBLE PRECISION D                                                      97
        LOGICAL L                                                               98
        COMMON/SETUP/SETFLG                                                     99
        COMMON/STATE/RSTART                                                    100
        COMMON/DOBLK/DO(15,3),REALDO(15,2),DINDEX,FINDEX                       101
        COMMON/BLOCKC/CONTER(15)                                               102
        COMMON/CONTRL/LINDEX,LPOINT,LEVEL,MAXLIN                               103
        COMMON/CONTR2/SCLINE(50)                                               104
        COMMON/TRAP/CLINE,LINE                                                 105
        COMMON/CHECK/BUG                                                       106
        PARAMETER(IR=3,IW=3)                                                   107
        MAXLIN=50                                                              108
        WRITE(3,*)'BUG=?'                                                      109
        READ(3,*)BUG                                                           110
        REWIND 3                                                               111
        WRITE(IW,*)"WELCOME TO 'INTERP', YOUR FORTRAN INTERPRETER"             112
 100    WRITE(IW,*)                                                            113
        WRITE(IW,*)'PLEASE ENTER THE NAME OF PROGRAM TO BE INTERPRETED'        114
        READ(IR,1150,ERR=150)FILNAM                                            115
        OPEN(UNIT=100,FILE=FILNAM,ERR=160)                                     116
C                                                                              117
C               CLEAR AND SET UP REGISTERS TO BEGIN NEW PROGRAM                118
C                                                                              119
 200    SETFLG=1                                                               120
        CALL DOBEGN(CLINE,CLNUM,SETFLG)                                        121
        CALL VARABL(0,C,TYPE,IVAL,RVAL,D,C,L)                                  122
        CALL SARRAY(0,C,TYPE,IVAL,RVAL,D,C,L)                                  123
        CALL MAP(IVAL,LPOINT)                                                  124
        CALL SEARCH(CLNUM,ENDFLG)                                              125
        SETFLG=0                                                               126
        LEVEL=0                                                                127
        ENDFLG=0                                                               128
        LPOINT=0                                                               129
        LINDEX=0                                                               130
        RSTART=1                                                               131
        REWIND 3                                                               132
        WRITE(IR,*)'THE PROGRAM NAMED ***',FILNAM,'*** IS BEING',              133
     $                  'INTERPRETED'                                         134
 10     GO TO(11,200,100,1000)RSTART                                           135
 11     LPOINT=LPOINT+1                                                        136
        IF(LPOINT.LE.LINDEX)THEN                                               137
                CLINE=SCLINE(LPOINT)                                           138
        ELSE                                                                   139
                LINDEX=LINDEX+1                                                140
                READ(100,1250,END=170,ERR=180)SCLINE(LINDEX)                   141
                CLINE=SCLINE(LINDEX)                                           142
        ENDIF                                                                  143
```

```
C                                                                       144
C              CHECK FOR COMMENT LINE                                   145
C                                                                       146
      LINE=CLINE                                                        147
      COMENT=INDEX(CLINE,'C')                                           148
      IF(COMENT .EQ. 1)GO TO 10                                         149
C                                                                       150
C              LOOK FOR LINE #                                          151
C                                                                       152
      EXPRES=CLINE(1:72)                                                153
      DO 15 J=1,5                                                       154
              BLANK=INDEX(EXPRES,' ')                                   155
              IF(BLANK.NE.1) THEN                                       156
                      LINFLG=1                                          157
                      WORD=CLINE(1:5)                                   158
                      CALL NUMBER(CLINE,WORD,TYPE,IVAL,R,D)             159
                      GO TO(210,200,100,1000)RSTART                     160
 210              IF(TYPE.NE.1)CALL ERROR(CLINE,303)                   161
                  GO TO (211,200,100,1000)RSTART                        162
 211              NUM=IVAL                                              163
                  IF(LINDEX.EQ.LPOINT)CALL MAP(NUM,LPOINT)              164
                  GO TO (30,200,100,1000)RSTART                         165
                  GO TO 30                                              166
              ELSE                                                      167
                  EXPRES=CLINE(J:72)                                    168
              ENDIF                                                     169
 15     CONTINUE                                                       170
 30     IF(LINFLG.EQ.1)CALL DCHECK(SCLINE,LPOINT,DOFLG,ENTRY)          171
        GO TO(31,200,100,1000)RSTART                                    172
C                                                                       173
C              LOOK FOR TYPE OF STATEMENT                               174
C                                                                       175
 31     CALL SEARCH(LPOINT,ENDFLG)                                     176
        GO TO (32,200,100,1000)RSTART                                   177
 32     IF(DOFLG.EQ.1)CALL DOFOOT(DOFLG,ENTRY)                         178
        GO TO (33,200,100,1000)RSTART                                   179
 33        IF(ENDFLG.EQ.1) GO TO 20                                     180
           IF(DOFLG.EQ.1)LPOINT=ENTRY                                   181
           DOFLG=0                                                      182
           LINFLG=0                                                     183
           GO TO 10                                                     184
 20     WRITE(IW,*)'PROGRAM ***',FILNAM,'*** HAS BEEN INTERPRETED'     185
        WRITE(IW,*)                                                     186
        WRITE(IW,*)'PRESS ENTER TO CONTINUE'                            187
        READ(IR,FMT='(A)')DUMMY                                         188
        CALL PATHWY                                                     189
        GO TO (1000,200,100,1000)RSTART                                 190
 150    WRITE(IW,*)FILNAM,'is not reconizable'                          191
        GO TO 100                                                       192
 160    WRITE(IW,*)'Program',FILNAM,'cannot be found'                   193
        GO TO 100                                                       194
 170    CALL ERROR(SCLINE(LINDEX-1),406)                                195
        GO TO (1000,200,100,1000)RSTART                                 196
 180    CALL ERROR(SCLINE(LPOINT),407)                                  197
        GO TO (1000,200,100,1000) RSTART                                198
 1150   FORMAT(A8)                                                      199
 1250   FORMAT(A72)                                                     200
 1000   WRITE(IW,*)"YOU ARE LEAVING 'INTERP' HAVE A NICE DAY"           201
        STOP                                                            202
        END                                                             203
```

```
C*****************************  SUBROUTINE MAP    ************************
C
C       CALLING FORMAT ----- MAP(USENUM,LPOINT)
C
C       PURPOSE:        TO CLEAR THE BRANCH TABLE AT BEGINNING OF
C                       NEW INTERPRETATIONS AND TO RECORD STATEMENT
C                       NUMBERS
C
C       I/O PARAMETERS:
C
C               INPUT:
C                       USENUM: THE STATEMENT NUMBER TO BE RECORDED
C                       LPOINT: THE LINE NUMBER OF THE LINE OF CODE WITH
C                               THE STATEMENT NUMBER
C
C       COMMON PARAMETERS:
C
C               SETFLG: A FLAG TO INDICATE TO CLEAR BRANCH TABLE AND
C                       BINDEX
C               BRANCH: A TABLE THAT MAPS THE PROGRAM BEING INTERPRETED
C               BINDEX: WHAT LINE OF TABLE BRANCH THE LAST ENTRY WAS
C                       MADE ON
C
C       ERROR NUMBERS CALLED:
C               NONE
C
C       SUBROUTINES CALLED:
C               NONE
C
C       SUBROUTINES THAT CALL MAP:
C               VMAIN
C
C*******************************************************************************
C
        SUBROUTINE MAP(USENUM,LPOINT)
        INTEGER BRANCH(20,6),SETFLG,BINDEX,BUG,USENUM
        COMMON/SETUP/SETFLG
        COMMON/BLOCK2/BRANCH,BINDEX
        COMMON/CHECK/BUG
        IF (SETFLG.EQ.1) THEN
                BINDEX=1
                DO 20 I=1,20
                        DO 10 J=1,6
                        BRANCH(I,J)=0
  10                    CONTINUE
  20            CONTINUE
                GO TO 1000
        ELSE
                CONTINUE
        ENDIF
        IF(LPOINT.LE.BRANCH(BINDEX,2))GO TO 1000
        BINDEX=BINDEX+1
        BRANCH(BINDEX,1)=USENUM
        BRANCH(BINDEX,2)=LPOINT
        BRANCH(BINDEX,5)=BRANCH(BINDEX-1,5)
        BRANCH(BINDEX,6)=BRANCH(BINDEX-1,6)
 1000   RETURN
        END
```

```
C*****************************   SUBROUTINE SEARCH   ********************
C
C       FORMAT  ------ SEARCH(CLNUM,ENDFLG) ----
C
C       PURPOSE:        TO SEARCH FOR KEY WORDS AND CALL APPROPRATE
C                       SUBROUTINE.
C
C       PARAMETERS:
C
C               USED FOR CONTROL:
C
C                       DATAFG: FLAG TO INDICATE USER'S PROGRAM HAS
C                               ADVANCED PAST DECLARATION SECTION
C                       ENDFLG: FLAG THAT IS SET WHEN END OF USER'S
C                               PROGRAM IS REACHED
C                       CLNUM:  THE LINE NUMBER OF THE LINE BEING
C                               INTERPRETED
C
C               FOR PARSING:
C
C                       TYPE:   DATA TYPE OF NUMBER
C                       EXPRES: LINE OF CODE BEING PARSED
C                       ASSIGN: ARRAY OF CHARACTER STRINGS OF DATA
C                               TYPE NAMES
C                       LOOK,LOOK1: INTEGER LOCATIONS OF ELEMENTS
C                               BEING PARSED
C                       COUNT:  SAFETY CHECK IN DO-LOOP USED TO FIND
C                               BLANK LINES
C                       VTYPE:  DATA TYPE OF VARIABLE BEING SENT
C                       I,R,D,C,L:  PATHWAYS TO PAST VALUES OF INTEGER,
C                               REAL, DOUBLE PRECISION, CHARACTER, AND
C                               LOGICAL VALUES TO AND FROM SUBROUTINES
C
C               COMMON PARAMETERS:
C
C                       SETFLG: FLAG WHEN SET INDICATES THE NEED TO
C                               CLEAR ALL TABLES AND REGISTERS
C                       CLINE,LINE:  THE LINE OF CODE BEING INTERPRETED
C                       LINDEX: THE LINE NUMBER OF THE MOST ADVANCED
C                               LINE THAT HAS BEEN INTERPRETED
C                       LPOINT: THE LINE NUMBER OF THE LINE BEING
C                               INTERPRETED
C                       LEVEL:  THE LEVEL OF IF NESTING
C                       MAXLIN: THE MAXIMUM NUMBER OF LINES ALLOWED
C                               IN THE USERS PROGRAM
C                       BRANCH: TABLE THAT MAPS THE USERS PROGRAM AS
C                               IT IS INTERPRETED
C                       BINDEX: NUMBER OF THE LAST LINE ENTERED IN
C                               BRANCH
C                       IFTABL: TABLE THAT KEEPS TRACK OF IF A BLOCK OF
C                               STATEMENTS  HAVE BEEN EXECUTED AND IF
C                               AN ELSE STATEMENT HAS BEEN FOUND IN A
C                               IF STRUCTURE
C                       IFFLAG: FLAG INDICATING A LOGICAL IF WITH A
C                               STATEMENT TO BE EXECUTED
C                       IFLINE: STATEMENT TO BE EXECUTED FOLLOWING A
C                               LOGICAL IF
C
C
C       ERROR NUMBERS CALLED:
C                       112
C                       113
C                       401
C                       406
C
C       SUBROUTINES CALLED:
C                       ERROR
C                       VARABL
C                       SARRAY
C                       DOBEGN
```

```
C                       SREAD                                                   71
C                       SWRITE                                                  72
C                       SIF                                                     73
C                       SGOTO                                                   74
C                       ARITH                                                   75
C                       SPAUSE                                                  76
C                                                                               77
C*******************************************************************************78
C                                                                               79
        SUBROUTINE SEARCH(CLNUM,ENDFLG)                                         80
        INTEGER I,VTYPE,ENDFLG,LOOK1,LOOK,BUG,CHECK,TYPE,                       81
     $   SETFLG,LEVEL,BRANCH(20,6),BINDEX,LINDEX,LPOINT,IFTABL(5,2),            82
     $   DATAFG,IFFLAG,MAXLIN,COUNT,CLNUM                                       83
        REAL R                                                                  84
        DOUBLE PRECISION D                                                      85
        CHARACTER C*60,LINE*72,WORD*9,ASSIGN*9,VAR*15,EXPRES*72,                86
     $                  IFLINE*72,CLINE*72                                      87
        DIMENSION ASSIGN(5)                                                     88
        LOGICAL L                                                               89
        COMMON/SETUP/SETFLG                                                     90
        COMMON/BLOCK2/BRANCH,BINDEX                                             91
        COMMON/CONTRL/LINDEX,LPOINT,LEVEL,MAXLIN                                92
        COMMON/TRAP/CLINE,LINE                                                  93
        COMMON/IFBLK/IFTABL,IFFLAG                                              94
        COMMON/IFBLKC/IFLINE                                                    95
        COMMON/CHECK/BUG                                                        96
        IF(SETFLG.EQ.1)THEN                                                     97
                DATAFG=0                                                        98
                GO TO 10000                                                     99
        ELSE                                                                   100
                CONTINUE                                                       101
        ENDIF                                                                  102
        ASSIGN(1)='INTEGER'                                                    103
        ASSIGN(2)='REAL'                                                       104
        ASSIGN(3)='DOUBLE'                                                     105
        ASSIGN(4)='CHARACTER'                                                  106
        ASSIGN(5)='LOGICAL'                                                    107
C                                                                              108
C               IS THE LINE TO BE EXECUTED FROM A LOGICAL IF STATEMENT         109
C                                                                              110
        LINE=CLINE(7:72)                                                       111
 10     IF(IFFLAG.EQ.1) LINE=IFLINE(1:72)                                      112
C                                                                              113
C               ELEMINANT BLANK CHARACTERS FROM BEGINNING OF LINE              114
C                                                                              115
        COUNT=0                                                                116
 100    LOOK=INDEX(LINE,' ')                                                   117
        IF (LOOK.EQ.1) THEN                                                    118
                LINE=LINE(2:72)                                                119
                COUNT=COUNT+1                                                  120
                IF(COUNT.GT.72)THEN                                            121
                        CALL ERROR(LINE,406)                                   122
                        GO TO 10000                                            123
                ELSE                                                           124
                        CONTINUE                                               125
                ENDIF                                                          126
                GO TO 100                                                      127
        ELSE                                                                   128
                WORD=LINE(1:LOOK-1)                                            129
        ENDIF                                                                  130
        EXPRES=LINE                                                            131
        IF(DATAFG.EQ.1) GO TO 400                                              132
C                                                                              133
C               LOOK FOR 'PROGRAM' STATEMENT                                   134
C                                                                              135
        LOOK=INDEX(EXPRES,'PROGRAM')                                           136
        IF(LOOK.NE.1)GO TO 200                                                 137
        GO TO 10000                                                            138
```

```
C                                                                       139
C               LOOK FOR 'DATA TYPE' STATEMENTS                         140
C                                                                       141
 200    DO 210 I=1,5                                                    142
               IF (WORD .EQ. ASSIGN(I)) THEN                            143
                       VTYPE=I                                          144
                       CALL VARABL(0,VAR,VTYPE,I,R,D,C,L)               145
                       GO TO 10000                                      146
               ELSE                                                     147
                       CONTINUE                                         148
               ENDIF                                                    149
 210    CONTINUE                                                        150
C                                                                       151
C               LOOK FOR 'DIMENSION' STATEMENT                          152
C                                                                       153
 300    LOOK=INDEX(EXPRES,'DIMENSION')                                  154
        IF(LOOK.NE.1) GO TO 400                                         155
        CALL SARRAY(0,VAR,TYPE,I,R,D,C,L)                               156
        GO TO 10000                                                     157
C                                                                       158
C               LOOK FOR THE BEGINNING OF A DO LOOP                     159
C                                                                       160
 400    DATAFG=1                                                        161
        LOOK=INDEX(EXPRES,'DO')                                         162
        IF(LOOK.NE.1)GO TO 500                                          163
        CALL DOBEGN(EXPRES,CLNUM,SETFLG)                                164
        GO TO 10000                                                     165
C                                                                       166
C               LOOK FOR 'READ' STATEMENT                               167
C                                                                       168
 500    LOOK=INDEX(EXPRES,'READ ')                                      169
        LOOK1=INDEX(EXPRES,'READ(')                                     170
        IF(LOOK.NE.1.AND.LOOK1.NE.1)GO TO 600                           171
        CALL SREAD(EXPRES)                                              172
        GO TO 10000                                                     173
C                                                                       174
C               LOOK FOR 'WRITE' STATEMENT                              175
C                                                                       176
 600    LOOK=INDEX(EXPRES,'WRITE ')                                     177
        LOOK1=INDEX(EXPRES,'WRITE(')                                    178
        IF (LOOK.NE.1.AND.LOOK1.NE.1) GO TO 800                         179
        CALL SWRITE(EXPRES)                                             180
        GO TO 10000                                                     181
C                                                                       182
C               LOOK FOR AN 'IF' STATEMENT                              183
C                                                                       184
 800    LOOK=INDEX(EXPRES,'IF ')                                        185
        LOOK1=INDEX(EXPRES,'IF(')                                       186
        IF(LOOK.NE.1.AND.LOOK1.NE.1) GO TO 900                          187
        CALL SIF(EXPRES,1)                                              188
        IF(IFFLAG.EQ.1)GO TO 10                                         189
        GO TO 10000                                                     190
C                                                                       191
C               LOOK 'ELSE' STATEMENT                                   192
C                                                                       193
 900    LOOK=INDEX(EXPRES,'ELSE')                                       194
        IF(LOOK.NE.1)GO TO 1000                                         195
        CALL SIF(EXPRES,2)                                              196
        GO TO 10000                                                     197
C                                                                       198
C               LOOK FOR 'ENDIF' STATEMENT                              199
C                                                                       200
 1000   LOOK=INDEX(EXPRES,'ENDIF')                                      201
        IF(LOOK.NE.1) GO TO 1100                                        202
        CALL SIF(EXPRES,3)                                              203
        GO TO 10000                                                     204
```

```
C               LOOK FOR 'GO TO' STATEMENT
C
 1100   LOOK=INDEX(EXPRES,'GO TO ')
        LOOK1=INDEX(EXPRES,'GOTO ')
        IF(LOOK.NE.1.AND.LOOK1.NE.1)GO TO 1200
        CALL SGOTO(EXPRES)
        GO TO 10000
C
C               LOOK FOR 'CONTINUE' STATEMENT
C
 1200   LOOK=INDEX(EXPRES,'CONTINUE')
        IF(LOOK.NE.1)GO TO 1300
        GO TO 10000
C
C               LOOK FOR AN ARITHMATIC '=' EXPRESSION
C
 1300   LOOK=INDEX(EXPRES,'=')
        IF (LOOK .EQ. 0) GO TO 1400
        CHECK=INDEX(EXPRES,"='")
        IF(CHECK.NE.0) THEN
                VAR=EXPRES(1:CHECK-1)
                EXPRES=EXPRES(CHECK+2:72)
                LOOK=INDEX(EXPRES,"'")
                IF(LOOK.EQ.0) THEN
                        CALL ERROR(LINE,112)
                        GO TO 10000
                ELSE
                        CONTINUE
                ENDIF
                C=EXPRES(1:LOOK-1)
                CHECK=INDEX(VAR,'(')
                IF(CHECK.EQ.0)THEN
                        CALL VARABL(1,VAR,4,I,R,D,C,L)
                        GO TO 10000
                ELSE
                        CALL SARRAY(1,VAR,4,I,R,D,C,L)
                        GO TO 10000
                ENDIF
        ELSE
                CONTINUE
        ENDIF
        CHECK=INDEX(EXPRES,'=.')
        IF(CHECK.NE.0) THEN
                VAR=EXPRES(1:CHECK-1)
                EXPRES=EXPRES(CHECK+2:72)
                LOOK=INDEX(EXPRES,'.')
                IF(LOOK.EQ.0) THEN
                        CALL ERROR(LINE,113)
                        GO TO 10000
                ELSE
                        CONTINUE
                ENDIF
                EXPRES=EXPRES(1:LOOK-1)
                IF(EXPRES.EQ.'TRUE')THEN
                        L=.TRUE.
                ELSE IF (EXPRES.EQ.'FALSE')THEN
                        L=.FALSE.
                ELSE
                        CALL ERROR(LINE,113)
                        GO TO 10000
                ENDIF
                CHECK=INDEX(VAR,'(')
                IF(CHECK.EQ.0)THEN
                        CALL VARABL(1,VAR,5,I,R,D,C,L)
                        GO TO 10000
                ELSE
                        CALL SARRAY(1,VAR,5,I,R,D,C,L)
                        GO TO 10000
                ENDIF
        ELSE
                CONTINUE
        ENDIF
```

```
      CALL ARITH(EXPRES)                                                278
      GO TO 10000                                                       279
C                                                                       280
C             LOOK FOR 'PAUSE' STATEMENT                                281
C                                                                       282
 1400 IF(EXPRES.NE.'PAUSE')GO TO 1500                                   283
      CALL SPAUSE                                                       284
      GO TO 10000                                                       285
C                                                                       286
C             LOOK FOR 'STOP' STATEMENT                                 287
C                                                                       288
 1500 IF(EXPRES .EQ.'STOP ') THEN                                       289
             ENDFLG=1                                                   290
             GO TO 10000                                                291
      ELSE                                                              292
             GO TO 1600                                                 293
      ENDIF                                                             294
C                                                                       295
C             LOOK FOR 'END' STATEMENT                                  296
C                                                                       297
 1600 IF(EXPRES.EQ.'END') THEN                                          298
             ENDFLG=1                                                   299
             GO TO 10000                                                300
      ELSE                                                              301
             CONTINUE                                                   302
      ENDIF                                                             303
C                                                                       304
C             NO KEY WORDS FOUND                                        305
C                                                                       306
      CALL ERROR(LINE,401)                                              307
10000 IF(IFFLAG.EQ.1)IFFLAG=0                                           308
      RETURN                                                            309
      END                                                               310
```

```
C*****************************   SUBROUTINE SREAD    ********************
C
C       CALLING FORMAT ------  SREAD(LINE) ------
C
C       PURPOSE:        TO EXECUTE A FORTRAN READ STATEMENT
C
C       I/O PARAMETERS:
C
C               INPUT:
C                       LINE:   A FORTRAN 77 FREE FORMATED READ
C                               STATEMENT
C               OUTPUT: READING OF VALUES OF CHARACTERS AND VARIABLES
C                       THROUGH THE DEVICE SPECIFIED
C
C       OTHER PARAMETERS:
C
C               EXPRES: THE CHARACTER STRING OF THE READ STATEMENT THAT
C                       IS SHORTENED AS IT IS PARSED
C               RPATH:  INTEGER VALUE OF INPUT DEVICE DESIRED
C               I,R,D,C,L:  PATHWAYS TO SEND INTEGER,REAL,DOUBLE
C                       PRECISION,CHARACTER AND LOGICAL VALUES FROM
C                       SUBROUTINES NUMBER,SARRAY AND VARABL
C               TYPE:   THE VARIABLE TYPE RETURNED FROM SUBROUTINE
C                       NUMBER,SARRAY,AND VARABL
C               WORD:   A CHARACTER STRING THAT REPRESENTS A NUMERIC
C                       VALUE
C               LPARTH,RPARTH: LOCATION OF LEFT AND RIGHT PARENTHESIS
C                       IN ARRAY VARIABLES
C               COMMA:  LOCATION OF LEFT MOST COMMA IN EXPRES
C               BLANK:  LOCATION OF BLANK AFTER LAST ELEMENT TO BE READ
C               ARAYFG: A FLAG TO MARK THAT THE ELEMENT BEING READ
C                       IS AN ARRAY ELEMENT
C               ENDFLG: A FLAG TO INDICATE THE LAST ELEMENT IN THE
C                       VARIABLE LIST
C       COMMON PARAMETERS:
C
C                       RSTART:  A VARIABLE THATS VALUE INDUCATES
C                                WHICH DIRECTION VMAIN IS TO TAKE
C                                     1:  CONTINUE EXECUTION
C                                     2:  RESTART EXECUTION FROM START
C                                     3:  EXECUTE DIFFERENT PROGRAM
C                                     4:  STOP
C
C       ERROR NUMBERS CALLED:
C                       101
C                       102
C                       103
C                       105
C                       111
C
C       SUBROUTINES CALLED:
C                       ERROR
C                       NUMBER
C                       SARRAY
C                       VARABL
C
C       SUBROUTINES THAT CALL SREAD:
C                       SEARCH
C
*******************************************************************************
C
C
        SUBROUTINE SREAD(LINE)
        INTEGER LOOK,COMMA,I,TYPE,BLANK,RPATH,ENDFLG,BUG,LPARTH,
     $          RPARTH,ARAYFG,RSTART
        REAL R
        DOUBLE PRECISION D
        CHARACTER LINE*72,EXPRES*72,WORD*10,C*60,CHECK*1
        LOGICAL L
        COMMON/STATE/RSTART
        COMMON/CHECK/BUG
        ENDFLG=0
```

```
C                                                                       73
C               PARSE READ STATEMENT                                    74
C                                                                       75
      LOOK=INDEX(LINE,'(')                                              76
      IF(LOOK.EQ.0) THEN                                                77
              CALL ERROR(LINE,101)                                      78
              RETURN                                                    79
      ELSE                                                              80
              CONTINUE                                                  81
      ENDIF                                                             82
      EXPRES=LINE(LOOK+1:72)                                            83
      COMMA=INDEX(EXPRES,',')                                           84
      IF(COMMA.GT.5.OR.COMMA.EQ.0) THEN                                 85
              CALL ERROR(LINE,105)                                      86
              RETURN                                                    87
      ELSE                                                              88
              CONTINUE                                                  89
      ENDIF                                                             90
C                                                                       91
C               FIND UNIT NUMBER TO READ FROM                           92
C                                                                       93
      WORD=EXPRES(1:COMMA-1)                                            94
      CALL NUMBER(LINE,WORD,TYPE,RPATH,R,D)                             95
      IF(RSTART.NE.1)GO TO 1000                                         96
      IF(TYPE.NE.1) THEN                                                97
              CALL ERROR(LINE,102)                                      98
              RETURN                                                    99
      ELSE                                                             100
              CONTINUE                                                 101
      ENDIF                                                            102
C                                                                      103
C               CHECK  FOR FREE FORMAT                                 104
C                                                                      105
      EXPRES=EXPRES(COMMA+1:72)                                        106
      LOOK=INDEX(EXPRES,'*)')                                          107
      IF(LOOK.NE.1) THEN                                               108
              CALL ERROR(LINE,103)                                     109
              RETURN                                                   110
      ELSE                                                             111
              CONTINUE                                                 112
      ENDIF                                                            113
C                                                                      114
C               PARSE VARIABLE NAMES BEING READ                        115
C                                                                      116
      EXPRES=EXPRES(3:72)                                              117
 100  COMMA=INDEX(EXPRES,',')                                          118
      LPARTH=INDEX(EXPRES,'(')                                         119
      IF((LPARTH.LT.COMMA.AND.LPARTH.NE.0).OR.(LPARTH.NE.0.AND.        120
     $                COMMA.EQ.0)) THEN                               121
              RPARTH=INDEX(EXPRES,')')                                 122
              WORD=EXPRES(1:RPARTH)                                    123
              COMMA=RPARTH+1                                           124
              CHECK=EXPRES(COMMA:COMMA)                                125
              IF(CHECK.NE.',')ENDFLG=1                                 126
              ARAYFG=1                                                 127
      ELSE                                                             128
              IF(COMMA.EQ.0) THEN                                      129
                      BLANK=INDEX(EXPRES,' ')                          130
                      IF(BLANK.EQ.0) THEN                              131
                              CALL ERROR(LINE,105)                     132
                              RETURN                                   133
                      ELSE                                             134
                              CONTINUE                                 135
                      ENDIF                                            136
                      WORD=EXPRES(1:BLANK-1)                           137
                      ENDFLG=1                                         138
              ELSE                                                     139
                      WORD=EXPRES(1:COMMA-1)                           140
              ENDIF                                                    141
              ARAYFG=0                                                 142
      ENDIF                                                            143
```

```
C                                                                       144
C              READ IN VALUE FOR VARIABLE IN READ STATEMENT             145
C                                                                       146
      IF(ARAYFG.EQ.1)CALL SARRAY(2,WORD,TYPE,I,R,D,C,L)                 147
      IF(RSTART.NE.1)GO TO 1000                                         148
      IF(ARAYFG.EQ.0)CALL VARABL(2,WORD,TYPE,I,R,D,C,L)                 149
      IF(RSTART.NE.1)GO TO 1000                                         150
      GO TO (110,120,130,140,150)TYPE                                   151
 110  READ(RPATH,*,ERR=300)I                                            152
      GO TO 200                                                         153
 120  READ(RPATH,*,ERR=300)R                                            154
      GO TO 200                                                         155
 130  READ(RPATH,*,ERR=300)D                                            156
      GO TO 200                                                         157
 140  READ(RPATH,FMT='(A60)',ERR=300)C                                  158
      GO TO 200                                                         159
 150  READ(RPATH,*,ERR=300)L                                            160
 200  IF(ARAYFG.EQ.1)CALL SARRAY(1,WORD,TYPE,I,R,D,C,L)                 161
      IF(RSTART.NE.1)GO TO 1000                                         162
      IF(ARAYFG.EQ.0)CALL VARABL(1,WORD,TYPE,I,R,D,C,L)                 163
      IF(RSTART.NE.1)GO TO 1000                                         164
      IF(ENDFLG.EQ.1)GO TO 1000                                         165
      EXPRES=EXPRES(COMMA+1:72)                                         166
      GO TO 100                                                         167
300   CALL ERROR(LINE,111)                                              168
1000  RETURN                                                            169
      END                                                               170
```

```
C*************************  SUBROUTINE SWRITE  ***********************
C
C       CALLING FORMAT------  SWRITE(LINE) -------
C
C       PURPOSE:  TO EXECUTE A FORTRAN WRITE STATEMENT
C
C       I/O PARAMETERS:
C
C               INPUT:
C                       LINE: A FORTRAN 77,FREE FORMATED WRITE STATEMENT
C               OUTPUT:
C                       PRINTING OF CHARACTER STRINGS AND VARIABLE VALUE
C                       TO THE OUTPUT DEVICE SPECIFIED
C
C       OTHER PARAMETERS:
C
C               FOR PARSING:
C
C                       EXPRES: THE CHARACTER STRING OF THE WRITE
C                               STATMENT THAT IS SHORTENED AS IT IS
C                               PARSED
C                       PATH:   CHARACTER STRING SPECIFING OUTPUT DEVICE
C                       WPATH:  THE INTEGER VALUE OF THE OUTPUT DEVICE
C                       AVAR:   THE CHARACTER STRING OF THE VARIABLE
C                               NAME TO BE PRINTED
C                       I,R,D,C,L: PATHWAYS TO SEND INTEGER,REAL,DOUBLE
C                               PRECISION,CHARACTER AND LOGICAL VALUES
C                               FROM SUBROUTINES NUMBER,SARRAY AND
C                               VARABL
C                       TYPE:   THE VARIABLE TYPE RETURNED FROM
C                               SUBROUTINE NUMBER,SARRAY AND VARABL
C                                       1-INTEGER
C                                       2-REAL
C                                       3-DOUBLE PRECISION
C                                       4-CHARACTER
C                                       5-LOGICAL
C                       START: A POINTER OF HOW MUCH OF THE LINE HAS
C                               BEEN PARSED
C                       LPAR,RPAR:  THE LOCATION OF LEFT AND RIGHT
C                               PARENTHESES IN VARIABLE ARRAYS
C                       ENDFLG: FLAG TO INDICATE THE LAST ELEMENT IN THE
C                               VARIABLE LIST
C                       COMMA,QOUT: LOCATION OF LEFT MOST COMMA AND
C                               QUOTE IN EXPRES
C
C                TO CREAT OUTPUT:
C
C                       STRING:A CHARACTER VARIABLE THAT STORES THE
C                               VALUE OF THE VARIABLE TO BE PRINTED
C                       LENGTH: THE LENGTH OR WIDTH OF THE VARIABLE
C                               VALUE OR CHARACTER STRING TO BE PRINTED
C                       CHAR:   A CHARACTER STRING TO BE PRINTED
C                       SAVE:   TEMPORARY STARAGE OF CHARACTER STRING TO
C                               BE PRINTED
C                       HOLD:   TEMPORARY STORAGE OF CHARACTER VARIABLE
C                               FOR CANCATENATION OF OUTPUT LINE
C                       LNSTOR: THE COMPLETE CHARACTER STRING TO BE
C                               PRINTED
C
C                TO DETECT ERRORS:
C
C                       LPARTH,RPARTH:  CHARACTER STRINGS TO CHECK
C                               SYNTAX OF LEFT AND RIGHT PARENTHESES
C                       STAR:   CHARACTER STRING TO CHECK SYNTAX FOR
C                               FREE FORMAT
C
C        ERROR NUMBERS CALLED:
C                         101
```

```
C                       102                                                    69
C                       103                                                    70
C                       104                                                    71
C                       105                                                    72
C                       114                                                    73
C                       115                                                    74
C                                                                              75
C       SUBROUTINES CALLED:                                                    76
C                       NUMBER                                                 77
C                       SARRAY                                                 78
C                       VARABL                                                 79
C                       ERROR                                                  80
C                                                                              81
C       SUBROUTINES THAT CALL SWRITE:                                          82
C                       SEARCH                                                 83
C                                                                              84
C*****************************************************************************85
C                                                                              86
C                                                                              87
        SUBROUTINE SWRITE(LINE)                                                88
        INTEGER COMMA,QUOT,START,TYPE,WPATH,I,ENDFLG,BUG,LPAR,RPAR,            89
     $          LENGTH,CHCK,COUNT,RSTART                                       90
        REAL R                                                                 91
        DOUBLE PRECISION D                                                     92
        CHARACTER LINE*72,LPARTH*1,RPARTH*1,PATH*10,STAR*2,EXPRES*72,          93
     $       C*60,AVAR*15,CHAR*72,CHECK*1,STRING*72,HOLD*(*),                  94
     $       LNSTOR*(*),SAVE*72                                                95
        LOGICAL L                                                              96
        COMMON/STATE/RSTART                                                    97
        COMMON/CHECK/BUG                                                       98
        ENDFLG=0                                                               99
C                                                                             100
C               CHECK SYNTAX OF STATEMENT                                     101
C                                                                             102
        EXPRES=LINE(1:72)                                                     103
        LPARTH=EXPRES(6:6)                                                    104
        IF(LPARTH .NE.'(') THEN                                               105
                CALL ERROR(LINE,101)                                          106
                RETURN                                                        107
        ELSE                                                                  108
                CONTINUE                                                      109
        ENDIF                                                                 110
C                                                                             111
C               FIND NUMBER OF OUTPUT DEVICE TO BE USED                       112
C                                                                             113
        COMMA=INDEX(EXPRES,',')                                               114
        IF(COMMA.EQ.0.OR.COMMA.GT.10) THEN                                    115
                CALL ERROR(LINE,105)                                          116
                RETURN                                                        117
        ELSE                                                                  118
                CONTINUE                                                      119
        ENDIF                                                                 120
        PATH=EXPRES(7:COMMA-1)                                                121
        CALL NUMBER(LINE,PATH,TYPE,WPATH,R,D)                                 122
        IF(RSTART.NE.1)GO TO 1000                                             123
        IF(TYPE .NE. 1) THEN                                                  124
                CALL ERROR(LINE,102)                                          125
                RETURN                                                        126
        ELSE                                                                  127
                CONTINUE                                                      128
        ENDIF                                                                 129
C                                                                             130
C               MAKE SURE USER IS USING FREE FORMAT                           131
C                                                                             132
        STAR=EXPRES(COMMA+1:COMMA+2)                                          133
        IF(STAR.NE.'*)')THEN                                                  134
                CALL ERROR(LINE,103)                                          135
                RETURN                                                        136
        ELSE                                                                  137
                CONTINUE                                                      138
        ENDIF                                                                 139
        START=COMMA+3                                                         140
```

```
C                                                                       141
C               PARSE VARIABLE LIST                                     142
C                                                                       143
 10     EXPRES=EXPRES(START:72)                                         144
        COMMA=INDEX(EXPRES,',')                                         145
        QUOT=INDEX(EXPRES,"'")                                          146
        IF (COMMA .EQ. 0) ENDFLG=1                                      147
        IF (ENDFLG .EQ. 1 .AND. QUOT .NE. 0) GO TO 550                  148
        IF (COMMA .LE. QUOT.OR.QUOT.EQ.0)THEN                           149
C                                                                       150
C               VARIABLE VALUE IS TO BE WRITTEN                         151
C                                                                       152
                IF(COMMA.EQ.0)COMMA=INDEX(EXPRES,' ')                   153
                LPAR=INDEX(EXPRES,'(')                                  154
                IF(LPAR.LT.COMMA.AND.LPAR.NE.0) THEN                    155
C                                                                       156
C               VARIABLE VALUE IS AN ARRAY ELEMENT                      157
C                                                                       158
                        RPAR=INDEX(EXPRES,')')                          159
                        IF(RPAR.EQ.0) THEN                              160
                                CALL ERROR(LINE,101)                    161
                                RETURN                                  162
                        ELSE                                            163
                                CONTINUE                                164
                        ENDIF                                           165
                        AVAR=EXPRES(1:RPAR)                             166
                        CALL SARRAY(2,AVAR,TYPE,I,R,D,C,L)              167
                        IF(RSTART.NE.1)GO TO 1000                       168
                        STRING=EXPRES(RPAR+1:RPAR+1)                    169
                        COMMA=RPAR+1                                    170
                        STRING=EXPRES(RPAR:72)                          171
                        CHCK=INDEX(STRING,',')                          172
                        IF(CHCK.EQ.0)ENDFLG=1                           173
                ELSE                                                    174
C                                                                       175
C               VARIABLE VALUE IS NOT AN ARRAY ELEMENT                  176
C                                                                       177
                        AVAR=EXPRES(1:COMMA-1)                          178
                        CALL VARABL(2,AVAR,TYPE,I,R,D,C,L)              179
                        IF(RSTART.NE.1)GO TO 1000                       180
                ENDIF                                                   181
C                                                                       182
C               PLACE VARIABLE VALUE INTO CHARACTER STRING 'STRING'     183
C                                                                       184
                GO TO (100,200,300,400,500)TYPE                         185
 100            WRITE(STRING,FMT='(I20)')I                              186
                GO TO 510                                               187
 200            WRITE(STRING,FMT='(F20.10)')R                           188
                GO TO 510                                               189
 300            WRITE(STRING,FMT='(E20.10)')D                           190
                GO TO 510                                               191
                WRITE(STRING,FMT='(A72)')C                              192
                GO TO 510                                               193
 500            WRITE(STRING,FMT='(A1)')L                               194
C                                                                       195
C               ELIMINATE LEADING BLANKS OF 'STRING'                    196
C                                                                       197
 510            COUNT=0                                                 198
                CHCK=INDEX(STRING,' ')                                  199
                IF(CHCK.EQ.1)THEN                                       200
                        STRING=STRING(2:60)                             201
                        COUNT=COUNT+1                                   202
                        IF(COUNT.GT.100) THEN                           203
                                CALL ERROR(LINE,114)                    204
                                RETURN                                  205
                        ELSE                                            206
                                CONTINUE                                207
                        ENDIF                                           208
                        GO TO 510                                       209
                ELSE                                                    210
                        CONTINUE                                        211
                ENDIF                                                   212
```

```
C                                                                       213
C               FIND LENGTH OF 'STRING'                                 214
C                                                                       215
                LENGTH=INDEX(STRING,' ')                                216
        ELSE                                                            217
C                                                                       218
C               A CHARACTER STRING IS TO BE WRITTEN                     219
C                                                                       220
 550            IF(QUOT .NE. 1) THEN                                    221
                       CALL ERROR(LINE,104)                             222
                       RETURN                                           223
                ELSE                                                    224
                       CONTINUE                                         225
                ENDIF                                                   226
                CHAR=EXPRES(2:72)                                       227
                QUOT=INDEX(CHAR,"'")                                    228
                IF(QUOT.GT.60.OR.QUOT.EQ.0)THEN                         229
                       CALL ERROR(LINE,115)                             230
                       RETURN                                           231
                ELSE                                                    232
                       CONTINUE                                         233
                ENDIF                                                   234
                LENGTH=QUOT-1                                           235
                SAVE=CHAR(1:QUOT-1)                                     236
C                                                                       237
C               WRITE CHARACTER STRING TO 'STRING'                      238
C                                                                       239
                WRITE(STRING,FMT='(A72)')SAVE                           240
                IF (ENDFLG .EQ. 0) THEN                                 241
                      CHECK=CHAR(QUOT+1:QUOT+1)                         242
                      IF(CHECK.NE.',') THEN                             243
                               CALL ERROR(LINE,104)                     244
                               RETURN                                   245
                       ELSE                                             246
                               CONTINUE                                 247
                       ENDIF                                            248
                      COMMA=QUOT+2                                      249
                ELSE                                                    250
                      CONTINUE                                          251
                ENDIF                                                   252
 600    ENDIF                                                          253
C                                                                       254
C               CANCATENATE 'STRING' TO EXISTING OUTPUT                 255
C                                                                       256
        HOLD=LNSTOR//'  '//STRING(1:LENGTH)                             257
 650    LNSTOR=HOLD                                                     258
C                                                                       259
C               UPDATE TO FIND NEXT ELEMENT TO BE PRINTED               260
C                                                                       261
        START=COMMA+1                                                   262
        IF (ENDFLG .EQ. 0) GO TO 10                                     263
C                                                                       264
C               ENTIRE LINE IS READ---WRITE IT TO DESIRED DEVICE        265
C                                                                       266
        WRITE(IW,*)LNSTOR                                               267
C                                                                       268
C               CLEAR LINE STORAGE FOR NEXT WRITE STATEMENT             269
C                                                                       270
        LNSTOR=' '                                                      271
 1000   RETURN                                                          272
        END                                                             273
```

```
C*****************************   SUBROUTINE VARABL  **********************
C
C       CALLING FORMAT ------  SARRAY(DIR,VAR,VTYPE,IVALUE,RVALUE,DVALUE
C                                    CVALUE,LVALUE)
C
C       PURPOSE:        TO INITIALIZE USER VARIABLE NAMES AND TO STORE
C                       AND RETRIEVE USER VARIABLE VALUES.
C
C       PARAMETERS:
C
C
C               I/O PARAMETERS:
C
C                       DIR: THE CODE OF TASK TO BE PREFORMED
C                               0    INITIALIZES VARIABLE NAME
C                               1    STORES A VARIABLE VALUE
C                               2    RETRIEVES A VARIABLE VALUE
C                               3    REMOVE VARIABLE NAME TO MAKE IT
C                                    AN ARRAY
C
C                       VAR:    CHARACTER STRING OF USER DEFINED
C                               VARIABLE NAME
C
C                       VTYPE:  THE TYPE OF VARIABLE BEING PASSED
C                               1 INTEGER
C                               2 REAL
C                               3 DOUBLE PRECISION
C                               4 CHARACTER
C                               5 LOGICAL
C
C                       IVALUE,RVALUE,DVALUE,CVALUE,DVALUE:  PATHWAYS
C                               TO SEND VALUES OF INTEGER,REAL,DOUBLE
C                               PRECISION,CHARACTER AND LOGICAL TO AND
C                               FROM SUBROUTINE
C
C               FOR PARSING:
C
C                       EXPRES: CHARACTER STRING THAT IS SHORTENED
C                               AS A LINE IS PARSED
C                       WORD:   CHARACTER STORAGE OF ARRAY SUBSCRIPTS
C
C
C               TO STORE AND RETRIEVE VARIABLES:
C
C                       VNTYPE: DATA TYPE VARIABLE NAME HAS BEEN DECLARE
C
C               COMMON PARAMETERS:
C
C                       LINE:   CHARACTER STRING OF LINE BEING
C                               INTERPRETED
C                       IVAL,RVAL,DVAL,CVAL,DVAL:  ARRAYS TO STORE USER
C                               DEFINED VALUES OF TYPES INTEGER,REAL,
C                               DOUBLE PRECISION,CHARACTER AND LOGICAL
C                       VARTBL: A 2-D CHARACTER ARRAYOF ALL USER DEFINED
C                               VARIABLE NAMES
C                       VINDEX: AN INTEGER ARRAY OF THE NUMBER OF
C                               VARIABLES DECLARED IN EACH TYPE
C                       SETFLG: FLAG USED TO CLEAR TABLES TO BEGIN
C                               NEW EXECUTION
C
C       ERROR NUMBERS CALLED:
C                       305
C                       501
C                       502
C                       503
C                       510
C                       601
C
C       SUBROUTINES CALLED:
C                       NONE
C
```

```
C       SUBROUTINES THAT CALL VARABL:                                          72
C                       VMAIN                                                  73
C                       SARRAY                                                 74
C                       SEARCH                                                 75
C                                                                              76
C*************************************************************************     77
C                                                                              78
        SUBROUTINE VARABL(DIR,VAR,VTYPE,IVALUE,RVALUE,DVALUE,                  79
     $                    CVALUE,LVALUE)                                       80
        INTEGER IVALUE,VTYPE,INDX,VINDEX,IVAL,DIR,VNTYPE,BUG,                  81
     $          SETFLG,AINDEX,AINFO,IARAY                                     82
        REAL RVALUE,RVAL,RARAY                                                 83
        DOUBLE PRECISION DVALUE,DVAL,DARAY                                     84
        CHARACTER VARTBL*6,CVALUE*60,VAR*6,CVAL*60,CLINE*72,LINE*72,           85
     $                   EXPRES*72,WORD*10                                     86
        LOGICAL LVALUE,LVAL,LARAY                                              87
        COMMON/TRAP/CLINE,LINE                                                 88
        COMMON/ERVAR1/VARTBL(20,5),CVAL(20)                                    89
        COMMON/EINTEG/VINDEX(5),IVAL(20),AINDEX(5),AINFO(10,5,3),              90
     $                          IARAY(100,100,10)                             91
        COMMON/EREAL/RVAL(20),RARAY(100,100,10)                                92
        COMMON/EDOUBL/DVAL(20),DARAY(100,100,10)                               93
        COMMON/ERVAR5/LVAL(20),LARAY(100,5)                                    94
        COMMON/SETUP/SETFLG                                                    95
        COMMON/CHECK/BUG                                                       96
C                                                                              97
C               CLEAR REGISTER FOR NEW EXECUTION                               98
C                                                                              99
        IF(SETFLG.EQ.1)THEN                                                   100
                DO 12 I=1,5                                                   101
                        DO 11 J=1,20                                          102
                        VARTBL(J,I)='      '                                  103
 11                     CONTINUE                                              104
                VINDEX(I)=0                                                   105
 12             CONTINUE                                                      106
                GO TO 1000                                                    107
        ELSE                                                                  108
                CONTINUE                                                      109
        ENDIF                                                                 110
C                                                                             111
C               ADDING THE VARIABLE NAME TO LIST                              112
C                                                                             113
        IF (DIR.EQ.0)THEN                                                     114
                LOOK=INDEX(LINE,' ')                                          115
                EXPRES=LINE(LOOK+1:72)                                        116
                IF(VTYPE.EQ.3) THEN                                           117
                        LOOK=INDEX(EXPRES,' ')                                118
                        WORD=EXPRES(1:LOOK-1)                                 119
                        IF(WORD .NE. 'PRECISION') THEN                        120
                                CALL ERROR(LINE,502)                          121
                                GO TO 1000                                    122
                        ELSE                                                  123
                                CONTINUE                                      124
                        ENDIF                                                 125
                        EXPRES=EXPRES(LOOK+1:72)                              126
                ELSE                                                          127
                        CONTINUE                                              128
                ENDIF                                                         129
 131            LOOK=INDEX(EXPRES,',')                                        130
                IF(VTYPE.EQ.4)LOOK=INDEX(EXPRES,'*')                          131
                IF (LOOK .GT. 7) THEN                                         132
                        CALL ERROR(LINE,501)                                  133
                        GO TO 1000                                            134
                ELSE                                                          135
                        CONTINUE                                              136
                ENDIF                                                         137
```

```
                IF (LOOK .GT. 0) THEN                                   138
                        VAR=EXPRES(1:LOOK-1)                            139
                        VINDEX(VTYPE)=VINDEX(VTYPE)+1                   140
                        INDX=VINDEX(VTYPE)                              141
                        IF(INDX.GT.20)THEN                              142
                                CALL ERROR(LINE,510)                    143
                                GO TO 1000                              144
                        ELSE                                            145
                                CONTINUE                                146
                        ENDIF                                           147
                        VARTBL(INDX,VTYPE)=VAR                          148
                        IF(VTYPE.EQ.4)LOOK=INDEX(EXPRES,',')            149
                        EXPRES=EXPRES(LOOK+1:72)                        150
                        GO TO 131                                       151
                ELSE                                                    152
                        LINEND=INDEX(EXPRES,' ')                        153
                        IF (LINEND .GT. 7)THEN                          154
                                CALL ERROR(LINE,501)                    155
                                GO TO 1000                              156
                        ELSE                                            157
                                CONTINUE                                158
                        ENDIF                                           159
                        VAR=EXPRES(1:LINEND-1)                          160
                        VINDEX(VTYPE)=VINDEX(VTYPE)+1                   161
                        INDX=VINDEX(VTYPE)                              162
                        IF(INDX.GT.20)THEN                              163
                                CALL ERROR(LINE,510)                    164
                                GO TO 1000                              165
                        ELSE                                            166
                                CONTINUE                                167
                        ENDIF                                           168
                        VARTBL(INDX,VTYPE)=VAR                          169
                ENDIF                                                   170
        DO 210 I=1,6                                                    171
 200    FORMAT(5(A8))                                                   172
 210    CONTINUE                                                        173
        ELSE                                                            174
C                                                                       175
C               FIND A VARABLE NAME IN THE LIST                         176
C                                                                       177
                DO 20 I= 1,5                                            178
                        DO 10 J = 1,20                                  179
                        IF (VARTBL(J,I).EQ.VAR) THEN                    180
                                VNTYPE=I                                181
                                INDX=J                                  182
                                GO TO 25                                183
                        ELSE                                            184
                                CONTINUE                                185
                        ENDIF                                           186
 10                     CONTINUE                                        187
 20             CONTINUE                                                188
                CALL ERROR(VAR,503)                                     189
                GO TO 1000                                              190
C                                                                       191
C               ASSIGNING A VARIABLE A VALUE                            192
C                                                                       193
 25             IF(DIR.EQ.1)THEN                                        194
                        IF(VTYPE.EQ.VNTYPE)THEN                         195
                             GO TO (30,40,50,60,70) VNTYPE              196
 30                          IVAL(INDX) = IVALUE                        197
                             GO TO 80                                   198
 40                          RVAL(INDX) = RVALUE                        199
                             GO TO 80                                   200
 50                          DVAL(INDX) = DVALUE                        201
                             GO TO 80                                   202
 60                          CVAL(INDX) = CVALUE                        203
                             GO TO 80                                   204
 70                          LVAL(INDX) = LVALUE                        205
                        ELSE                                            206
                             IF(VTYPE.GT.3) THEN                        207
                                CALL ERROR(VAR,305)                     208
                                GO TO 1000                              209
```

```
                              ELSE                                          210
                                CONTINUE                                    211
                               ENDIF                                        212
                              GO TO(310,320,330)VTYPE                       213
 310                          IF(VNTYPE.EQ.2)RVAL(INDX)=IVALUE              214
                              IF(VNTYPE.EQ.3)DVAL(INDX)=IVALUE              215
                              GO TO 350                                     216
 320                          IF(VNTYPE.EQ.1)IVAL(INDX)=RVALUE              217
                              IF(VNTYPE.EQ.3)DVAL(INDX)=RVALUE              218
                              GO TO 350                                     219
 330                          IF(VNTYPE.EQ.1)IVAL(INDX)=DVALUE              220
                              IF(VNTYPE.EQ.2)RVAL(INDX)=DVALUE              221
 350                       ENDIF                                            222
C                                                                           223
C                    RETRIEVING A VARIABLE VALUE                            224
C                                                                           225
 80                  ELSE IF(DIR.EQ.2) THEN                                 226
                          VTYPE=VNTYPE                                      227
                          GO TO (100,110,120,130,140) VTYPE                 228
 100                      IVALUE = IVAL(INDX)                               229
                          GO TO 150                                         230
 110                      RVALUE = RVAL(INDX)                               231
                          GO TO 150                                         232
 120                      DVALUE = DVAL(INDX)                               233
                          GO TO 150                                         234
 130                      CVALUE = CVAL(INDX)                               235
                          GO TO 150                                         236
 140                      LVALUE = LVAL(INDX)                               237
                    ELSE IF(DIR.EQ.3) THEN                                  238
C                                                                           239
C                   CHANGING A VARABLE NAME TO AN ARRAY                     240
C                                                                           241
                          VTYPE=VNTYPE                                      242
                          VARTBL(INDX,VNTYPE)='      '                      243
                    ELSE                                                    244
                          CALL ERROR(VAR,601)                               245
 150                ENDIF                                                   246
          ENDIF                                                             247
 1000     RETURN                                                           248
          END                                                              249
```

```
C*****************************   SUBROUTINE SARRAY  *********************
C
C       CALLING FORMAT ------  SARRAY(DIR,AVAR,NTYPE,IV,RV,DV,CV,LC)
C
C       PURPOSE:        TO INITIALIZE USER ARRAY NAMES AND TO STORE
C                       AND RETRIEVE USER ARRAY VALUES.
C
C       PARAMETERS:
C
C
C               I/O PARAMETERS:
C
C                       DIR: THE CODE OF TASK TO BE PREFORMED
C                               0    INITIALIZES VARIABLE NAME
C                               1    STORES A VARIABLE VALUE
C                               2    RETRIEVES A VARIABLE VALUE
C
C                       AVAR:   CHARACTER STRING OF USER DEFINED
C                               VARIABLE NAME, INCLUDING SUBSCRIPT
C                               VALUES, IF ANY
C
C                       NTYPE:  THE TYPE OF VALUE BEING PASSED TO
C                               BE STORED:
C                               1:  INTEGER
C                               2:  REAL
C                               3:  DOUBLE PRECISION
C                               4:  CHARACTER
C                               5:  LOGICAL
C
C               THE REMAINING PARAMETERS ARE PATHWAYS TO SEND VALUES
C
C               FOR PARSING:
C
C                       EXPRES: CHARACTER STRING THAT IS SHORTENED
C                               AS A LINE IS PARSED
C                       CHECK:  CHARACTER STRING USED TO CHECK SYNTAX
C                               OF LINE
C                       LOCAT:  A CHARACTER STRING OF THE SUBSCRIPT
C                               VALUES OF THE USER DEFINED ARRAY
C                       VARNAM: CHARACTER STRING OF THE USER DEFINED
C                               ARRAY NAME WITH NO SUBSCRIPTS
C                       VTYPE:  INTEGER VALUE THAT DISCRIBES VARIABLE
C                               TYPE
C                       ATYPE:  INTEGER VALUE THAT DISCRIBES ARRAY TYPE
C                       INDX:   INTEGER VALUE THAT INDEXES WHICH USER
C                               DEFINED VARIABLE IS SOUGHT
C                       ANUM:   INTEGER VALUE THAT INDEXES WHICH USER
C                               DEFINED ARRAY IS SOUGHT
C                       COMMA:  LOCATION OF COMMA IN ARRAY SUBSCRIPT
C                       DIMEN:  STORES VALUE OF THE DIMENSION OF THE
C                               USER DEFINED ARRAY
C                       MAXROW: MAXIMUM NUMBER OF ROWS IN USER DEFINED
C                               ARRAY
C                       MAXCOL: MAXIMUM NUMBER OF COLUMES IN USER
C                               DEFINED ARRAY
C                       COLVAL: THE COLUMN SUBSCRIPT OF USER ARRAY
C                       ROWVAL: THE ROW SUBSCRIPT OF USER ARRAY
C                       I,R,D:  PATHWAYS TO RECIEVE INTEGER,REAL AND
C                               DOUBLE PRECISION VALUES FROM SUBROUTINE
C                               NUMBER
C                       WORD:   CHARACTER STORAGE OF ARRAY SUBSCRIPTS
C                       LLOOK,RLOOK:  INTEGER LOCATIONS OF RIGHT AND
C                               LEFT PARTHESIS
C                       LOOK:   INTEGER LOCATION OF A CHARACTER
C                               USED IN PARSING
C                       SYNCHK: INTEGER VALUE USED TO CHECK SYNTAX
C                               ERRORS
C
```

```
C               TO STORE AND RETRIEVE VARIABLE:                                 69
C                                                                               70
C                       IV,RV,DV,CV,LV:  TEMPORARY STORAGE OF THE               71
C                               VALUE BEING PASSED OF TYPE INTEGER              72
C                               REAL,DOUBLE PRECISION, CHARACTER OR             73
C                               LOGICAL                                         74
C                                                                               75
C               ERROR TRAPPING:                                                 76
C                                                                               77
C                       TYPE:   THE TYPE OF VALUE RETURNED FROM                 78
C                               CALLING SUBROUTINE SCRIPT WHEN AN               79
C                               ARRAY SUBSCRIPT IS SENT.(MUST BE AN             80
C                               INTEGER)                                        81
C                                                                               82
C               COMMON PARAMETERS:                                              83
C                                                                               84
C                       RSTART: A VARIABLE THATS VALUE INDICATES                85
C                               WHICH DIRECTION VMAIN IS TO TAKE:               86
C                                   1:  CONTINUE EXECUTION                      87
C                                   2:  RESTART EXECUTION FROM START            88
C                                   3:  EXECUTE DIFFERENT PROGRAM               89
C                                   4:  STOP                                    90
C                                                                               91
C                       LINE:   CHARACTER STRING OF LINE BEING                  92
C                               INTERPRETED                                     93
C                       EXPRES: CHARCTER STRING THAT IS SHORTENED               94
C                               AS LINE IS PARSED                               95
C                       CHECK:  CHARACTER STRING USED TO CHECK                  96
C                               SYNTAX OF LINE                                  97
C                       ARAYTB: A 2-D CHARACTER ARRAY OF ALL USER               98
C                               DEFINED ARRAY NAMES                             99
C                       CARAY:  A CHARACTER ARRAY OF USER DEFINED              100
C                               CHARACTER ARRAY VALUES                         101
C                       AINDEX: AN INTEGER ARRAY OF THE NUMBER OF              102
C                               ARRAYS DECLARED IN EACH TYPE                   103
C                       AINFO:  AN INTEGER ARRAY THAT STORES THE USER          104
C                               DEFINED ARRAY DIMENSIONS                       105
C                       IARAY,RARAY,DARAY,LARAY: ARRAYS OF USER DEFINED        106
C                               ARRAY VALUES OF TYPE INTEGER,REAL,             107
C                       SETFLG: FLAG USED TO CLEAR TABLES TO BEGIN             108
C                               NEW EXECUTION                                  109
C                                                                              110
C       ERROR NUMBERS CALLED:                                                  111
C                       304                                                    112
C                       305                                                    113
C                       504                                                    114
C                       505                                                    115
C                       506                                                    116
C                       506                                                    117
C                       507                                                    118
C                       509                                                    119
C                                                                              120
C       SUBROUTINES CALLED:                                                    121
C                       VARABL                                                 122
C                       SCRIPT                                                 123
C                                                                              124
C       SUBROUTINES THAT CALL SARRAY:                                          125
C                       VMAIN                                                  126
C                       SREAD                                                  127
C                       SWRITE                                                 128
C                       SEARCH                                                 129
C                                                                              130
C*****************************************************************************131
C                                                                              132
      SUBROUTINE SARRAY(DIR,AVAR,NTYPE,IV,RV,DV,CV,LV)                         133
      INTEGER DIR,TYPE,IV,LOOK, LLOOK,RLOOK,SYNCHK,ATYPE,VINDEX,               134
     $         COMMA,DIMEN,MAXROW,MAXCOL,ROWVAL,COLVAL,AINDEX,IVAL,           135
     $         ANUM,AINFO,IARAY,BUG,NTYPE,RSTART,SETFLG,INDX                  136
      REAL RV,R,RARAY,RVAL                                                     137
      DOUBLE PRECISION DV,D,DARAY,DVAL                                         138
      CHARACTER CV*60,LINE*72,C*8,EXPRES*72,WORD*10,AVAR*15,HOLD*10,           139
     $         ARAYTB*6,CHECK*10,CARAY*60,CLINE*72,VARNAM*6,LOCAT*10           140
```

```
      LOGICAL LV,L,LARAY,LVAL                                                141
      COMMON/ERVAR3/CARAY(100,5)                                             142
      COMMON/EINTEG/VINDEX(5),IVAL(20),AINDEX(5),AINFO(10,5,3),              143
     $              IARAY(100,100,10)                                        144
      COMMON/EREAL/RVAL(20),RARAY(100,100,10)                                145
      COMMON/EDOUBL/DVAL(20),DARAY(100,100,10)                               146
      COMMON/ERVAR5/LVAL(20),LARAY(100,5)                                    147
      COMMON/ERVAR6/ARAYTB(10,5)                                             148
      COMMON/TRAP/CLINE,LINE                                                 149
      COMMON/STATE/RSTART                                                    150
      COMMON/SETUP/SETFLG                                                    151
      COMMON/CHECK/BUG                                                       152
C                                                                            153
C             CLEAR REGISTER FOR NEW EXECUTION                               154
C                                                                            155
      IF(SETFLG.EQ.1) THEN                                                   156
             DO 32 I=1,5                                                     157
                    DO 22 J=1,10                                             158
                           ARAYTB(J,I)='       '                             159
                           DO 12 K=1,3                                       160
                                  AINFO(J,I,K)=0                             161
 12                        CONTINUE                                          162
 22                  CONTINUE                                                163
 32          CONTINUE                                                        164
             GO TO 1000                                                      165
      ELSE                                                                   166
             CONTINUE                                                        167
      ENDIF                                                                  168
C                                                                            169
C      *****  ADD A VARIABLE NAME TO LIST  *****                             170
C                                                                            171
      IF(DIR.EQ.0)THEN                                                       172
             LOOK=INDEX(LINE,' ')                                            173
             CHECK=LINE(1:LOOK)                                              174
             IF(CHECK.NE.'DIMENSION') THEN                                   175
                    CALL ERROR(LINE,507)                                     176
                    GO TO 1000                                               177
             ELSE                                                            178
                    CONTINUE                                                 179
             ENDIF                                                           180
             EXPRES=LINE(LOOK+1:72)                                          181
 5           LLOOK=INDEX(EXPRES,'(')                                         182
             IF(LLOOK.EQ.0) THEN                                             183
                    CALL ERROR(LINE,509)                                     184
                    RETURN                                                   185
             ELSE                                                            186
                    CONTINUE                                                 187
             ENDIF                                                           188
             WORD=EXPRES(1:LLOOK-1)                                          189
             CALL VARABL(3,WORD,ATYPE,I,R,D,C,L)                             190
             IF(RSTART.NE.1)GO TO 1000                                       191
             AINDEX(ATYPE)=AINDEX(ATYPE)+1                                   192
             INDX=AINDEX(ATYPE)                                              193
             ARAYTB(INDX,ATYPE)=WORD                                         194
C                                                                            195
C            PARSE EXPRESSION FOR DIMENSIONS OF ARRAY                        196
C                                                                            197
             RLOOK=INDEX(EXPRES,')')                                         198
             SYNCHK=RLOOK-LLOOK                                              199
             IF(SYNCHK.LT.2) THEN                                            200
                    CALL ERROR(LINE,150)                                     201
                    GO TO 1000                                               202
             ELSE                                                            203
                    CONTINUE                                                 204
             ENDIF                                                           205
             HOLD=EXPRES(LLOOK+1:RLOOK-1)                                    206
             COMMA=INDEX(HOLD,',')                                           207
             IF(COMMA.EQ.0)THEN                                              208
                    AINFO(AINDEX(ATYPE),ATYPE,1)=1                           209
                    CALL SCRIPT(LINE,HOLD,TYPE,I,R,D)                        210
                    IF(RSTART.NE.1) GO TO 1000                               211
```

```
                IF(TYPE.NE.1) THEN                                      212
                        CALL ERROR(LINE,304)                            213
                        GO TO 1000                                      214
                ELSE                                                    215
                        CONTINUE                                        216
                ENDIF                                                   217
                AINFO(AINDEX(ATYPE),ATYPE,2)=I                          218
                AINFO(AINDEX(ATYPE),ATYPE,3)=1                          219
        ELSE                                                            220
                IF(ATYPE.EQ.4.OR.ATYPE.EQ.5)THEN                        221
                        CALL ERROR(LINE,508)                            222
                        RETURN                                          223
                ELSE                                                    224
                        CONTINUE                                        225
                ENDIF                                                   226
                AINFO(AINDEX(ATYPE),ATYPE,1)=2                          227
                WORD=HOLD(1:COMMA-1)                                    228
                CALL SCRIPT(LINE,WORD,TYPE,I,R,D)                       229
                IF(RSTART.NE.1) GO TO 1000                              230
                IF(TYPE.NE.1) THEN                                      231
                        CALL ERROR(LINE,304)                            232
                        GO TO 1000                                      233
                ELSE                                                    234
                        CONTINUE                                        235
                ENDIF                                                   236
                AINFO(AINDEX(ATYPE),ATYPE,2)=I                          237
                WORD=HOLD(COMMA+1:RLOOK-1)                              238
                CALL SCRIPT(LINE,WORD,TYPE,I,R,D)                       239
                IF(RSTART.NE.1) GO TO 1000                              240
                IF(TYPE.NE.1) THEN                                      241
                        CALL ERROR(LINE,304)                            242
                        GO TO 1000                                      243
                ELSE                                                    244
                        CONTINUE                                        245
                ENDIF                                                   246
                AINFO(AINDEX(ATYPE),ATYPE,3)=I                          247
        ENDIF                                                           248
        EXPRES=EXPRES(RLOOK+1:72)                                       249
        COMMA=INDEX(EXPRES,',')                                         250
        IF(COMMA.GT.1) THEN                                             251
                CALL ERROR(LINE,150)                                    252
                GO TO 1000                                              253
        ELSE                                                            254
                CONTINUE                                                255
        ENDIF                                                           256
        IF(COMMA.EQ.0)GO TO 1000                                        257
        EXPRES=EXPRES(COMMA+1:72)                                       258
        GO TO 5                                                         259
ELSE                                                                    260
C                                                                       261
C       *****  PARSE VARABLE NAME OF VALUE TO BE STORED OF RETRIEVED***  262
C                                                                       263
        LLOOK=INDEX(AVAR,'(')                                           264
        RLOOK=INDEX(AVAR,')')                                           265
        IF(RLOOK.LE.LLOOK) THEN                                         266
                CALL ERROR(LINE,150)                                    267
                GO TO 1000                                              268
        ELSE                                                            269
                CONTINUE                                                270
        ENDIF                                                           271
        LOCAT=AVAR(LLOOK+1:RLOOK-1)                                     272
        VARNAM=AVAR(1:LLOOK-1)                                          273
C                                                                       274
C       FIND VARIABLE NAME IN LIST                                      275
C                                                                       276
        DO 20 I=1,5                                                     277
                DO 10 J=1,AINDEX(I)                                     278
                IF(ARAYTB(J,I).EQ.VARNAM) THEN                          279
                        ATYPE=I                                         280
                        ANUM=J                                          281
                        GO TO 50                                        282
```

```
                ELSE                                                    283
                        CONTINUE                                        284
                ENDIF                                                   285
10              CONTINUE                                                286
20      CONTINUE                                                        287
        CALL ERROR(LINE,504)                                            288
        GO TO 1000                                                      289
50      DIMEN=AINFO(ANUM,ATYPE,1)                                       290
        MAXROW=AINFO(ANUM,ATYPE,2)                                      291
        MAXCOL=AINFO(ANUM,ATYPE,3)                                      292
C                                                                       293
C       PARSE EXPRESSION FOR ARRAY ELEMENT IDENTIFIERS                  294
C                                                                       295
        COLVAL=1                                                        296
        COMMA=INDEX(LOCAT,',')                                          297
        IF(COMMA.EQ.0)THEN                                              298
                IF(DIMEN.NE.1)  THEN                                    299
                        CALL ERROR(LINE,505)                            300
                        GO TO 1000                                      301
                ELSE                                                    302
                        CONTINUE                                        303
                ENDIF                                                   304
                CALL SCRIPT(LINE,LOCAT,TYPE,I,R,D)                      305
                IF(RSTART.NE.1)GO TO 1000                               306
                IF(TYPE.NE.1) THEN                                      307
                        CALL ERROR(LINE,304)                            308
                        GO TO 1000                                      309
                ELSE                                                    310
                        CONTINUE                                        311
                ENDIF                                                   312
                ROWVAL=I                                                313
                IF(ROWVAL.GT.MAXROW) THEN                               314
                        CALL ERROR(LINE,506)                            315
                        GO TO 1000                                      316
                ELSE                                                    317
                        CONTINUE                                        318
                ENDIF                                                   319
        ELSE                                                            320
                IF(DIMEN.NE.2) THEN                                     321
                        CALL ERROR(LINE,505)                            322
                        GO TO 1000                                      323
                ELSE                                                    324
                        CONTINUE                                        325
                ENDIF                                                   326
                WORD=LOCAT(1:COMMA-1)                                   327
                CALL SCRIPT(LINE,WORD,TYPE,I,R,D)                       328
                IF(RSTART.NE.1)GO TO 1000                               329
                IF(TYPE.NE.1) THEN                                      330
                        CALL ERROR(LINE,304)                            331
                        GO TO 1000                                      332
                ELSE                                                    333
                        CONTINUE                                        334
                ENDIF                                                   335
                ROWVAL=I                                                336
                IF(ROWVAL.GT.MAXROW)THEN                                337
                        CALL ERROR(LINE,506)                            338
                        GO TO 1000                                      339
                ELSE                                                    340
                        CONTINUE                                        341
                ENDIF                                                   342
                WORD=LOCAT(COMMA+1:RLOOK-1)                             343
                CALL SCRIPT(LINE,WORD,TYPE,I,R,D)                       344
                IF(RSTART.NE.1)GO TO 1000                               345
                IF(TYPE.NE.1)                                           346
                        CALL ERROR(LINE,304)                            347
                        GO TO 1000                                      348
                ELSE                                                    349
                        CONTINUE                                        350
                ENDIF                                                   351
                COLVAL=I                                                352
```

```
                        IF(COLVAL.GT.MAXCOL) THEN                          353
                                CALL ERROR(LINE,506)                       354
                                GO TO 1000                                 355
                        ELSE                                               356
                                CONTINUE                                   357
                        ENDIF                                              358
                ENDIF                                                      359
C                                                                          360
C       *****  STORE A VARIABLE VALUE  ******                              361
C                                                                          362
                IF(DIR.EQ.1)THEN                                           363
                     IF(NTYPE.EQ.ATYPE)THEN                                364
                       GO TO(110,120,130,140,150)ATYPE                     365
 110                    IARAY(ROWVAL,COLVAL,ANUM)=IV                       366
                        GO TO 300                                          367
 120                    RARAY(ROWVAL,COLVAL,ANUM)=RV                       368
                        GO TO 300                                          369
 130                    DARAY(ROWVAL,COLVAL,ANUM)=DV                       370
                        GO TO 300                                          371
 140                    CARAY(ROWVAL,ANUM)=CV                              372
                        GO TO 300                                          373
 150                    LARAY(ROWVAL,ANUM)=LV                              374
                        GO TO 300                                          375
                    ELSE                                                   376
                        IF(NTYPE.GT.3) THEN                                377
                                CALL ERROR(LINE,305)                       378
                                GO TO 1000                                 379
                        ELSE                                               380
                                CONTINUE                                   381
                        ENDIF                                              382
                          GO TO (210,220,230)NTYPE                         383
 210                      IF(ATYPE.EQ.2)RARAY(ROWVAL,COLVAL,ANUM)=IV       384
                          IF(ATYPE.EQ.3)DARAY(ROWVAL,COLVAL,ANUM)=IV       385
                          GO TO 300                                        386
 220                      IF(ATYPE.EQ.1)IARAY(ROWVAL,COLVAL,ANUM)=RV       387
                          IF(ATYPE.EQ.3)DARAY(ROWVAL,COLVAL,ANUM)=RV       388
                          GO TO 300                                        389
 230                      IF(ATYPE.EQ.1)IARAY(ROWVAL,COLVAL,ANUM)=DV       390
                          IF(ATYPE.EQ.2)RARAY(ROWVAL,COLVAL,ANUM)=DV       391
 300                 ENDIF                                                 392
                ELSE                                                       393
C                                                                          394
C       *****  RETRIEVING A VARAIBLE   *******                             395
C                                                                          396
                   NTYPE=ATYPE                                             397
                   GO TO (310,320,330,340,350)NTYPE                        398
 310               IV=IARAY(ROWVAL,COLVAL,ANUM)                            399
                   GO TO 400                                               400
 320               RV=RARAY(ROWVAL,COLVAL,ANUM)                            401
                   GO TO 400                                               402
 330               DV=DARAY(ROWVAL,COLVAL,ANUM)                            403
                   GO TO 400                                               404
 340               CV=CARAY(ROWVAL,ANUM)                                   405
                   GO TO 400                                               406
 350               LV=LARAY(ROWVAL,ANUM)                                   407
                   GO TO 400                                               408
 400            ENDIF                                                      409
        ENDIF                                                              410
 1000   RETURN                                                             411
        END                                                                412
```

```
C*****************************   SUBROUTINE ARITH  **********************
C
C       FORMAT  ------ ARITH(LINE) -----
C
C       PURPOSE:        TO EVALUATE ARITHMATIC EXPRESSIONS, BY BUILDING
C                       A TABLE TO REPRESENT THE EPRESSION AND THEN
C                       EVALUATING  THE TABLE
C
C       PARAMETERS:
C
C               I/O PARAMETERS:
C
C                       LINE:   LINE OF CODE OF MATHMATICAL EXPRESSION
C
C               PARSING PARAMETER:
C
C                       EQUAL:  INTEGER LOCATION OF EQUAL SIGN
C                       LPOINT,RPOINT:  LEFT AND RIGHT POINTERS OF WHAT
C                               PART OF THE LINE IS BEING PARSED
C                       LASTEL: FLAG TO INDICATE LAST ELEMENT, END
C                               OF THE LINE
C                       FUNCFG: FLAG TO INDICATE AN INTRISIC FUNCTION
C                       ARAYFL: FLAG TO INDICATE AN ARRAY ELEMENT
C                       OLDOP:  THE TYPE OF OPERATOR TO THE LEFT OF
C                               OPERATOR BEING PARSED
C                       NEXTPT: INTEGER LOCATION OF NEXT OPERATOR
C                       PFLAG:  FLAG INDICATING IF A PARENTHESIS IS
C                               PRESENT
C                       POWFLG: FLAG TO INDICATE THE OPERATOR IS A RAISE
C                               TO A POWER, TAKING UP 2 SPACES RATHER
C                               THAN ONE
C
C               PARAMETERS USED IN EVALUATING THE EXPRESSION
C
C                       OPRAND: CODE OF OPERATOR TYPES:
C                                       1:      +
C                                       2:      -
C                                       3:      /
C                                       4:      (
C                                       5:      )
C                                       6:      *
C                                       7:      **
C                       ARTHTB: TABLE THAT IS BUILT TO REPRESENT THE
C                               ARITHMATIC EXPRESSION
C                       RARTH:  EXTENSION OF ARTHTB FOR REAL VALUES
C                       DARTH:  EXTERSION OF ARTHTB FOR DOUBLE
C                               PRECISION VALUES
C                       TOP,BOTTOM:  POINTERS THAT BRACKET THE PORTION
C                               OF THE TABLE BEING EVALUATED
C                       START:  POINTER OF WHAT NEEDS TO BE EVALUATED
C                               NEXT
C                       FINISH: THE END OF THE TABLE BEING EVALUATED
C
C       ERRORS CALLED:
C               117
C
C       SUBROUTINES CALLED:
C               MATH
C               NUMBER
C               VARABL
C               SARRAY
C               DOFUNC
C               FCNAME
C
C       SUBROUTINES THAT CALL ARITH:
C               SEARCH
C
C*******************************************************************************
C
```

```
      SUBROUTINE ARITH(LINE)                                                70
      INTEGER CHECK,PRIOR,OPRAND(6),ARTHTB(72,4),EQUALS,BUG,LOK,            71
     $       OPTYPE,LPOINT,RPOINT,NEXTPT,START,FINISH,PFLAG,RSTART,         72
     $       TOP,BOTTOM,LASTEL,TYPE,VTYPE,OLDOP,FUNCFG,ARAYFG               73
      REAL VALUE,RARTH(72)                                                  74
      DOUBLE PRECISION DARTH(72)                                            75
      CHARACTER SYMBOL(6)*1,LINE*72,RESULT*15,EXPRES*72,                    76
     $          LOOK*1,NUMBER*10,DIG*1,VARNAM*15,C*60,TRY*72               77
      LOGICAL POWFLG,CONTFG,L                                               78
       COMMON/STATE/RSTART                                                  79
       COMMON/CHECK/BUG                                                     80
      SYMBOL(1)='+'                                                         81
      SYMBOL(2)='-'                                                         82
      SYMBOL(3)='/'                                                         83
      SYMBOL(4)='('                                                         84
      SYMBOL(5)=')'                                                         85
      SYMBOL(6)='*'                                                         86
      OLDOP=0                                                               87
      FUNCFG=0                                                              88
      CONTFG=.FALSE.                                                        89
      DO 50 I = 1,72                                                        90
         DO 20 J = 1,4                                                      91
         ARTHTB(I,J)=0                                                      92
 20       CONTINUE                                                          93
 50      CONTINUE                                                           94
      EQUALS=INDEX(LINE,'=')                                                95
      RESULT=LINE(1:EQUALS-1)                                               96
      LPOINT=EQUALS                                                         97
 10   EXPRES=LINE(LPOINT+1:72)                                              98
      POWFLG=.FALSE.                                                        99
      NEXTPT=100                                                           100
C                                                                          101
C         FIND NEXT OPERAND IN STATEMENT                                   102
C                                                                          103
      DO 100 J=1,6                                                         104
        OPRAND(J)=INDEX(EXPRES,SYMBOL(J))                                  105
        IF(OPRAND(J).NE.0.AND.OPRAND(J).LT.NEXTPT)THEN                     106
          NEXTPT=OPRAND(J)                                                 107
          OPTYPE=J                                                         108
        ELSE                                                               109
          CONTINUE                                                         110
        ENDIF                                                              111
 100  CONTINUE                                                             112
      IF(NEXTPT.NE.100) THEN                                               113
C                                                                          114
C         PLACE OPERAND IN TABLE                                           115
C                                                                          116
            RPOINT=LPOINT+NEXTPT                                           117
            GO TO (105,105,110,120,120,130),OPTYPE                         118
 105        ARTHTB(RPOINT,1)=5                                             119
            ARTHTB(RPOINT,2)=OPTYPE                                        120
            GO TO 140                                                      121
 110        ARTHTB(RPOINT,1)=4                                             122
            ARTHTB(RPOINT,2)=OPTYPE                                        123
            GO TO 140                                                      124
 120        ARTHTB(RPOINT,1)=2                                             125
            ARTHTB(RPOINT,2)=OPTYPE                                        126
            GO TO 140                                                      127
 130        LOOK=LINE(RPOINT+1:RPOINT+1)                                   128
            IF(LOOK.EQ.'*')THEN                                            129
                 ARTHTB(RPOINT,1)=3                                        130
                 POWFLG=.TRUE.                                             131
            ELSE                                                           132
                 ARTHTB(RPOINT,1)=4                                        133
                 ARTHTB(RPOINT,2)=OPTYPE                                   134
            ENDIF                                                          135
      ELSE                                                                 136
```

```
C                                                                       137
C         YOU ARE AT END OF STATEMENT-INDEX TO END OF LAST WORD         138
C                                                                       139
             NEXTPT=INDEX(EXPRES,' ')                                   140
             RPOINT=LPOINT+NEXTPT                                       141
             CONTFG=.TRUE.                                              142
       ENDIF                                                            143
  140  WIDTH=RPOINT-LPOINT-1                                            144
       IF (WIDTH .EQ. 0) THEN                                           145
C                                                                       146
C          TWO OPERANDS NEXT TO EACH OTHER CHECK IF LEGAL               147
C                                                                       148
         IF (CONTFG) GO TO 195                                          149
         IF(OPTYPE .EQ. 4)THEN                                          150
                 IF(OLDOP.EQ.5) THEN                                    151
                         CALL ERROR(LINE,1117)                          152
                         GO TO 1000                                     153
                 ELSE                                                   154
                         CONTINUE                                       155
                 ENDIF                                                  156
         ELSE IF (OLDOP .EQ. 4) THEN                                    157
                IF(.NOT.(OPTYPE.EQ.1.OR.OPTYPE.EQ.2)) THEN              158
                         CALL ERROR(LINE,2117)                          159
                         GO TO 1000                                     160
                 ELSE                                                   161
                         CONTINUE                                       162
                 ENDIF                                                  163
         ELSE IF (OLDOP .EQ. 5) THEN                                    164
                 IF(OPTYPE .EQ. 4) THEN                                 165
                         CALL ERROR(LINE,3117)                          166
                         GO TO 1000                                     167
                 ELSE                                                   168
                         CONTINUE                                       169
                 ENDIF                                                  170
         ELSE                                                           171
                 CALL ERROR(LINE,1)                                     172
         ENDIF                                                          173
         OLDOP=OPTYPE                                                   174
         LPOINT=RPOINT                                                  175
         GO TO 10                                                       176
       ELSE                                                             177
         OLDOP=OPTYPE                                                   178
       ENDIF                                                            179
             VARNAM=LINE(LPOINT+1:RPOINT-1)                             180
             ARAYFG=0                                                   181
             FUNCFG=0                                                   182
             IF(OPTYPE.EQ.4) CALL FCNAME(ARTHTB,RPOINT,VARNAM,FUNCFG)   183
             IF(RSTART.NE.1)GO TO 1000                                  184
             IF(FUNCFG.EQ.1) GO TO 199                                  185
             IF(OPTYPE.EQ.4) THEN                                       186
                TRY=LINE(RPOINT:72)                                     187
                LOK=INDEX(TRY,')')                                      188
                ARTHTB(RPOINT,1)=0                                      189
                RPOINT=RPOINT+LOK-1                                     190
                VARNAM=LINE(LPOINT+1:RPOINT)                            191
                CALL SARRAY(2,VARNAM,VTYPE,IVALUE,RVALUE,DVALUE,C,L)    192
                IF(RSTART.NE.1)GO TO 1000                               193
                GO TO 191                                               194
             ELSE                                                       195
                CONTINUE                                                196
             ENDIF                                                      197
         CALL NUMBER(LINE,VARNAM,VTYPE,IVALUE,RVALUE,DVALUE)            198
         IF(RSTART.NE.1)GO TO 1000                                      199
         IF (VTYPE.GT.3) WRITE(3,*)'CHARACTER VALUE IN NUMERIC EXP.'    200
 C                                                                      201
 C          PLACE VALUE OF WORD IN TABLE                                202
 C                                                                      203
  191    GO TO (192,194,196)VTYPE                                       204
  192      ARTHTB(LPOINT+1,4)=IVALUE                                    205
           ARTHTB(LPOINT+1,3)=1                                         206
           GO TO 198                                                    207
```

```
 194      RARTH(LPOINT+1)=RVALUE                                          208
          ARTHTB(LPOINT+1,3)=2                                            209
          GO TO 198                                                       210
 196      DARTH(LPOINT+1)=DVALUE                                          211
          ARTHTB(LPOINT+1,3)=3                                            212
 198  ARTHTB(LPOINT+1,1)=9                                                213
 199  LPOINT=RPOINT                                                       214
      IF(POWFLG) LPOINT=LPOINT+1                                          215
      IF (CONTFG) GO TO 195                                               216
      GO TO 10                                                            217
C                                                                         218
C         TABLE IS BUILT                                                  219
C                                                                         220
 195    LASTEL=RPOINT                                                     221
        DO 17 I=1,35                                                      222
  18    FORMAT(I3,4(2X,I3),2X,F8.3,2X,F8.3)                               223
  17    CONTINUE                                                          224
        START=1                                                           225
        FINISH=0                                                          226
C                                                                         227
C               LOOK FOR CLOSE PARENTHESIS                                228
C                                                                         229
 200    DO 210 I= START,LASTEL                                            230
               IF(ARTHTB(I,1).EQ.2.AND.ARTHTB(I,2).EQ.5) THEN             231
                       BOTTOM=I                                           232
                       PFLAG=1                                            233
                       GO TO 220                                          234
               ELSE                                                       235
                       PFLAG=0                                            236
               ENDIF                                                      237
 210    CONTINUE                                                          238
 220    IF(PFLAG.EQ.0) THEN                                               239
               FINISH=1                                                   240
               TOP=1                                                      241
               BOTTOM=LASTEL                                              242
        ELSE                                                              243
               DO 230 I = BOTTOM,7,-1                                     244
               IF (ARTHTB(I,1).EQ.2.AND.ARTHTB(I,2).EQ.4) THEN            245
                        TOP=I                                             246
                        GO TO 240                                         247
               ELSE                                                       248
                        CONTINUE                                          249
               ENDIF                                                      250
 230           CONTINUE                                                   251
        ENDIF                                                             252
 240    DO 260 I=3,5                                                      253
               DO 250 J= TOP,BOTTOM                                       254
               IF (ARTHTB(J,1).EQ.I) CALL MATH(J,TOP,ARTHTB,RARTH,        255
     $                                         DARTH)                   256
               IF(RSTART.NE.1)GO TO 1000                                  257
 250           CONTINUE                                                   258
 260    CONTINUE                                                          259
        IF (FINISH.EQ.1) GO TO 270                                        260
        ARTHTB(TOP,1)=0                                                   261
        ARTHTB(BOTTOM,1)=0                                                262
      IF(PFLAG.EQ.1.AND.ARTHTB(TOP-1,1).EQ.7)CALL DOFUNC(LINE,ARTHTB,     263
     $                        RARTH,DARTH,TOP)                          264
        IF(RSTART.NE.1)GO TO 1000                                         265
        START=BOTTOM                                                      266
        GO TO 200                                                         267
 270    DO 280 I=EQUALS+1,LASTEL                                          268
               IF(ARTHTB(I,1).EQ.9) THEN                                  269
                       TYPE=ARTHTB(I,3)                                   270
                       GO TO 300                                          271
               ELSE                                                       272
                       CONTINUE                                           273
               ENDIF                                                      274
 280    CONTINUE                                                          275
 300    GO TO (310,320,330)TYPE                                           276
 310        IVALUE=ARTHTB(I,4)                                            277
            GO TO 350                                                     278
```

```
320         RVALUE=RARTH(I)                                                279
            GO TO 350                                                      280
330         DVLAUE=DARTH(I)                                                281
350     LLOOK=INDEX(RESULT,'(')                                            282
        RLOOK=INDEX(RESULT,')')                                            283
        IF(LLOOK.NE.0.OR.RLOOK.NE.0) THEN                                  284
           CALL SARRAY(1,RESULT,TYPE,IVALUE,RVALUE,DVALUE,C,L)             285
           IF(RSTART.NE.1)GO TO 1000                                       286
        ELSE                                                               287
           CALL VARABL(1,RESULT,TYPE,IVALUE,RVALUE,DVALUE,C,L)             288
           IF(RSTART.NE.1)GO TO 1000                                       289
        ENDIF                                                              290
1000    RETURN                                                             291
        END                                                                292
```

```
C*****************************   SUBROUTINE MATH   **********************
C
C       FORMAT  ------ MATH(X,TOP,TABLE,RTABLE,DTABLE) ------
C
C       PURPOSE:        TO EVALUATE THE ARITH TABLE BY PERFORMING
C                       ARITHMATIC OPERATIONS
C
C       PARAMETERS:
C
C               I/O PARAMETERS:
C
C                       X:      THE ROW OF THE ARITH TABLE THAT HOLDS
C                               THE OPERATOR ABOUT TO BE EVALUATED
C                       TOP:    THE HIGHEST ROW OF THE ARITH TABLE THAT
C                               IS PRESENTLY BEING EVALUATED
C                       TABLE:  THE ARITH TABLE
C                       RTABLE: EXTENTION OF ARITH TABLE FOR REAL NUMBER
C                       DTABLE: EXTENTION OF ARITH TABLE FOR DOUBLE
C                               PRECISION NUMBERS
C
C               OTHER PARAMETER:
C
C                       VAR1,RVAR1,DVAR1:  THE OPERANDS TO THE LEFT
C                               OF THE OPERATOR BEING EVALUTED OF TYPES
C                               INTEGER, REAL, DOUBLE PRECISION
C                       VAR2,RVAR2,DVAR2:  THE OPERANDS TO THE RIGHT
C                               OF THE OPERATOR BEING EVALUTED OF TYPES
C                               INTEGER, REAL, DOUBLE PRECISION
C                       SS:     A FLAG THAT INDICATES IT IS SINGLE SIDED
C                               (ONLY ONE OPERATOR AND ONE OPERAND -3)
C                       ANSTYP: THE DATA TYPE OF THE RESULTING ANSWER
C                       VAR1PT,VAR2PT: THE ROWS OF ARITH TABLE THAT THE
C                               OPERAND BRACKETING THE OPERATOR ARE ON
C                       OPTYPE: THE CODE FOR THE TYPE OF OPERATOR:
C                                       1:      +
C                                       2:      -
C                                       3:      /
C                                       4:      (  NOTHING DONE
C                                       5:      )  NOTHING DONE
C                                       6:      *
C                                       7:      **
C
C       ERRORS NUMBERS CALLED
C               NONE
C
C       SUBROUTINES CALLED:
C               NONE
C
C       SUBROUTINES THAT CALL MATH:
C               ARITH
C
C*****************************************************************************
C
      SUBROUTINE MATH(X,TOP,TABLE,RTABLE,DTABLE)
      INTEGER X,OPTYPE,TEST,VAR1,VAR2,VAR1PT,TABLE(72,3),ANSTYP,
     $        VAR2PT,TOP,SS,BUG
      REAL RTABLE(72),RVAR1,RVAR2
      DOUBLE PRECISION DTABLE(72),DVAR1,DVAR2
      CHARACTER LINE*72
      COMMON/CHECK/BUG
      COMMON/CONTR2/LINE
      SS=0
      OPTYPE=TABLE(X,2)
      IF (OPTYPE.EQ.0) OPTYPE=7
      TABLE(X,1)=0
      DO 300 I= X,TOP,-1
         IF (TABLE(I,1).EQ.9) THEN
            TYPE1=TABLE(I,3)
            VAR1PT=I
            TABLE(I,1)=0
            GO TO 310
         ELSE
```

```
          CONTINUE                                                      73
        ENDIF                                                           74
300   CONTINUE                                                          75
        SS=1                                                            76
        TYPE1=0                                                         77
        DVAR1=0.0E0                                                     78
        RVAR1=0.0                                                       79
        VAR1=0                                                          80
310   DO 320 I= X,73                                                    81
          IF  (TABLE(I,1).EQ.9) THEN                                    82
               TYPE2=TABLE(I,3)                                         83
               VAR2PT=I                                                 84
               TABLE(I,1)=0                                             85
               GO TO 340                                                86
          ELSE                                                          87
               CONTINUE                                                 88
          ENDIF                                                         89
 320  CONTINUE                                                          90
 340    IF(TYPE1.EQ.3.OR.TYPE2.EQ.3)THEN                                91
C                                                                       92
C                       ARITHMATIC PROCEDURES FOR DOUBLE PRECISION VARIA 93
C                                                                       94
                ANSTYP=3                                                95
                IF(TYPE1.EQ.1)DVAR1=TABLE(VAR1PT,4)                     96
                IF(TYPE1.EQ.2)DVAR1=RTABLE(VAR1PT)                      97
                IF(TYPE1.EQ.3)DVAR1=DTABLE(VAR1PT)                      98
                IF(TYPE2.EQ.1)DVAR2=TABLE(VAR2PT,4)                     99
                IF(TYPE2.EQ.2)DVAR2=RTABLE(VAR2PT)                     100
                IF(TYPE2.EQ.3)DVAR2=DTABLE(VAR2PT)                     101
                GO TO(410,420,430,440,440,450,460)OPTYPE               102
 410            DTABLE(X)=DVAR1+DVAR2                                  103
                GO TO 700                                              104
 420            DTABLE(X)=DVAR1-DVAR2                                  105
                GO TO 700                                              106
 430            IF(DVAR2.EQ.0)THEN                                     107
                        CALL ERROR(LINE,212)                           108
                        RETURN                                         109
                ELSE                                                   110
                        CONTINUE                                       111
                ENDIF                                                  112
                DTABLE(X)=DVAR1/DVAR2                                  113
 440            GO TO 700                                              114
 450            DTABLE(X)=DVAR1*DVAR2                                  115
                GO TO 700                                              116
 460            DTABLE(X)=DVAR1**DVAR2                                 117
                GO TO 700                                              118
C                                                                      119
C                    ARITHMATIC PROCEDURES FOR REAL NUMBERS            120
C                                                                      121
        ELSE IF (TYPE1.EQ.2 .OR. TYPE2.EQ.2) THEN                      122
                ANSTYP=2                                               123
                IF(TYPE1 .EQ. 1)RVAR1=TABLE(VAR1PT,4)                  124
                IF(TYPE1 .EQ. 2)RVAR1=RTABLE(VAR1PT)                   125
                IF(TYPE2 .EQ. 1)RVAR2=TABLE(VAR2PT,4)                  126
                IF(TYPE2 .EQ. 2)RVAR2=RTABLE(VAR2PT)                   127
                GO TO (510,520,530,540,550,560)OPTYPE                  128
 510            RTABLE(X)=RVAR1+RVAR2                                  129
                GO TO 700                                              130
 520            RTABLE(X)=RVAR1-RVAR2                                  131
                GO TO 700                                              132
 530            IF(RVAR.EQ.0) THEN                                     133
                        CALL ERROR(LINE,212)                           134
                        RETURN                                         135
                ELSE                                                   136
                        CONTINUE                                       137
                ENDIF                                                  138
                RTABLE(X)=RVAR1/RVAR2                                  139
 540            GO TO 700                                              140
 550            RTABLE(X)=RVAR1*RVAR2                                  141
                GO TO 700                                              142
 560            RTABLE(X)=RVAR1**RVAR2                                 143
                GO TO 700                                              144
        ELSE
```

```
C                                                                       145
C                                                                       146
C                       ARITHMATIC PROCEDURES FOR INTEGER VARIABLES     147
C                                                                       148
                IF (SS.EQ.0) THEN                                       149
                        VAR1=TABLE(VAR1PT,4)                            150
                        VAR2=TABLE(VAR2PT,4)                            151
                ELSE                                                    152
                        VAR2=TABLE(VAR2PT,4)                            153
                ENDIF                                                   154
                ANSTYP=1                                                155
                GO TO (610,620,630,640,640,650,660)OPTYPE               156
 610            TABLE(X,4)=VAR1+VAR2                                    157
                GO TO 700                                               158
 620            TABLE(X,4)=VAR1-VAR2                                    159
                GO TO 700                                               160
 630            IF(VAR2.EQ.0)THEN                                       161
                        CALL ERROR(LINE,212)                            162
                        RETURN                                          163
                ELSE                                                    164
                        CONTINUE                                        165
                ENDIF                                                   166
                TABLE(X,4)=VAR1/VAR2                                    167
 640            GO TO 700                                               168
 650            TABLE(X,4)=VAR1*VAR2                                    169
                GO TO 700                                               170
 660            TABLE(X,4)=VAR1**VAR2                                   171
                GO TO 700                                               172
        ENDIF                                                           173
 700    TABLE(X,3)=ANSTYP                                               174
        TABLE(X,1)=9                                                    175
        RETURN                                                          176
        END                                                             177
```

```
C*************************  SUBROUTINE FCNAME  ***********************
C
C       FORMAT  -----  FCNAME(TABLE,RPOINT,VARNAM,FLAG) ------
C
C       PURPOSE:        TO CODE WHICH INTRINSIC FUNCTION IS BEING USED
C
C       PARAMETERS:
C
C               I/O PARAMETERS:
C
C                       TABLE:  THE BRANCH TABLE USED TO MAP THE
C                               PROGRAM AS IT IS INTERPRETED
C                       RPOINT: THE RIGHT HAND COLUMN OF THE FUNCTION
C                               CHARACTER STRING. USED TO POINT
C                               PLACEMENT OF CODE IN TABLE
C                       VARNAM: THE CARACTER STRING OF THE FUNCTION
C                               NAME
C                       FLAG:   A FLAG WHEN SET INDICATES THE CHARACTER
C                               STRING SENT WAS A FUNCTION
C
C               OTHER PARAMETERS:
C
C                       FUNCT:  CHARACTER ARRAY OF ALL INTRINSIC
C                               FUNCTION NAMES. CODES ARE:
C                                       1       SIN
C                                       2       COS
C                                       3       TAN
C                                       4       ASIN
C                                       5       ACOS
C                                       6       ASIN
C                                       7       ABS
C                                       8       SQRT
C                                       9       EXP
C                                      10       LOG
C
C
C
C       ERROR NUMBERS CALLED:
C               NONE
C
C       SUBROUTINES CALLED:
C               NONE
C
C       SUBROUTINES THAT CALL FCNAME:
C               ARITH
C
C*****************************************************************************
C
      SUBROUTINE FCNAME(TABLE,RPOINT,VARNAM,FLAG)
      INTEGER RPOINT,TABLE(72,4),FLAG,BUG
      CHARACTER FUNCT(10)*6,VARNAM*6
      COMMON/CHECK/BUG
      FUNCT(1)='SIN'
      FUNCT(2)='COS'
      FUNCT(3)='TAN'
      FUNCT(4)='ASIN'
      FUNCT(5)='ACOS'
      FUNCT(6)='ATAN'
      FUNCT(7)='ABS'
      FUNCT(8)='SQRT'
      FUNCT(9)='EXP'
      FUNCT(10)='LOG'
      FLAG=0
      DO 10 I=1,10
            IF(FUNCT(I).EQ.VARNAM) GO TO 20
10    CONTINUE
      RETURN
20    TABLE(RPOINT-1,2)=I
      TABLE(RPOINT-1,1)=7
      FLAG=1
      RETURN
      END
```

```
C*****************************   SUBROUTINE DOFUNC   ********************
C
C       FORMAT  ------DOFUNC(LINE,TABLE,RTABEL,DTABLE,POINT)-----
C
C       PURPOSE:        TO PERFORM THE INTRINSIC FUNCTIONS OF A
C                       MATHMATICAL EXPRESSION
C
C       PARAMETERS:
C
C               I/O PARAMETERS:
C
C                       LINE:   THE LINE OF CODE THE MATHMATICAL
C                               EXPRESSION IS IN
C                       TABLE:  THE ARTHTB THAT REPRESENTS THE
C                               MATHMATICAL EXPRESSION TO BE
C                               EVALUATED
C                       RTABLE,DTABLE:  THE ROW OF ARTHTB  USED FOR
C                               STORAGE OF REAL AND DOUBLE PRECISION
C                               VALUES, RESPECTIVELY
C                       POINT:  THE ROW OF ARTHTB THAT THE OPEN
C                               PARENTHESIS IS ON
C
C
C               OTHER PARAMETERS:
C
C                       NUMTYP: THE TYPE OF THE NUMBER TO BE USED BY THE
C                               FUNCTION
C                       VALUE,RVALUE,DVALUE:  THE VALUE OF THE NUMBER
C                               FOR THE FUNCTION TO USE OF TYPE INTEGER
C                               REAL OR DOUBLE PRECISION, RESPECTIVELY
C                       RCHECK,DCHECK:  VALUES CALCULATED TO CHECK
C                               ACCEPTABLE RANGES BEFORE A FUNCTION IS
C                               ATTEMPTED
C                       FNTYPE: THE CODE FOR THE FUNCTION TO BE
C                               PREFORMED
C                                       1       SIN
C                                       2       COS
C                                       3       TAN
C                                       4       ASIN
C                                       5       ACOS
C                                       6       ATAN
C                                       7       ABS
C                                       8       SQRT
C                                       9       EXP
C                                      10       LOG
C
C       ERRORS CALLED:
C               201
C               202
C               203
C
C       SUBROUTINES CALLED:
C               NONE
C
C       SUBROUTINES THAT CALL DOFUNC:
C               ARITH
C*************************************************************************
C
        SUBROUTINE DOFUNC(LINE,TABLE,RTABLE,DTABLE,POINT)
        INTEGER TABLE(72,4),NUMTYP,VALUE,POINT,FNTYPE,BUG
        REAL RVALUE,RTABLE(72),RCHECK
        DOUBLE PRECISION DVALUE,DTABLE(72),DCHECK
        CHARACTER C*60,LINE*72
        COMMON/CHECK/BUG
        FNTYPE=TABLE(POINT-1,2)
        DO 100 I=POINT,80
                IF(TABLE(I,1).EQ.9)GO TO 200
 100    CONTINUE
 200    NUMTYP = TABLE(I,3)
        TABLE(I,1)=0
C
```

```
C       BRANCH TO TYPE OF VALUE TO BE EVALUATED                          72
C                                                                        73
      GO TO (300,400,500)NUMTYP                                          74
C                                                                        75
C       ABSOLUTE VALUE FUNCTION PREFORMED ON INTEGER VALUE               76
C                                                                        77
 300    IF(FNTYPE.EQ.7)THEN                                              78
               VALUE=TABLE(I,4)                                          79
               TABLE(POINT-1,4)=ABS(VALUE)                               80
               TABLE(POINT-1,3)=1                                        81
               TABLE(POINT-1,1)=9                                        82
        ELSE                                                             83
               CALL ERROR(LINE,201)                                      84
               GO TO 1000                                                85
        ENDIF                                                            86
        GO TO 600                                                        87
C                                                                        88
C       FUNCTION TO BE PREFORMED ON REAL VALUE                           89
C                                                                        90
 400    RVALUE=RTABLE(I)                                                 91
      GO TO (401,410,420,430,440,450,460,470,480,490)FNTYPE              92
 401    RTABLE(POINT-1)=SIN(RVALUE)                                      93
        GO TO 495                                                        94
 410    RTABLE(POINT-1)=COS(RVALUE)                                      95
        GO TO 495                                                        96
 420    RTABLE(POINT-1)=TAN(RVALUE)                                      97
        GO TO 495                                                        98
 430    RCHECK=ABS(RVALUE)                                               99
        IF (RCHECK.GT.1.0) THEN                                         100
               CALL ERROR(LINE,202)                                     101
               GO TO 1000                                               102
        ELSE                                                            103
               CONTINUE                                                 104
        ENDIF                                                           105
        RTABLE(POINT-1)=ASIN(RVALUE)                                    106
        GO TO 495                                                       107
 440    RCHECK=ABS(RVALUE)                                              108
        IF (RCHECK.GT.1.0) THEN                                         109
               CALL ERROR(LINE,202)                                     110
               GO TO 1000                                               111
        ELSE                                                            112
               CONTINUE                                                 113
        ENDIF                                                           114
        RTABLE(POINT-1)=ACOS(RVALUE)                                    115
        GO TO 495                                                       116
 450    RTABLE(POINT-1)=ATAN(RVALUE)                                    117
        GO TO 495                                                       118
 460    RTABLE(POINT-1)=ABS(RVALUE)                                     119
        GO TO 495                                                       120
 470    IF(RVALUE.LT.0.0) THEN                                          121
               CALL ERROR(LINE,203)                                     122
               GO TO 1000                                               123
        ELSE                                                            124
               CONTINUE                                                 125
        ENDIF                                                           126
        RTABLE(POINT-1)=SQRT(RVALUE)                                    127
        GO TO 495                                                       128
 480    RTABLE(POINT-1)=EXP(RVALUE)                                     129
        GO TO 495                                                       130
 490    RTABLE(POINT-1)=LOG(RVALUE)                                     131
C                                                                       132
C       STORE IN TABLE CODE FOR NUMBER AND NUMBER TYPE                  133
C                                                                       134
 495    TABLE(POINT-1,1)=9                                              135
        TABLE(POINT-1,3)=2                                              136
        GO TO 600                                                       137
500     DVALUE=DTABLE(I)                                                138
C                                                                       139
```

```
C       FUNCTIONS TO BE FREFORMED ON DOUBLE PRECISION VALUES                140
C                                                                           141
      GO TO (501,510,520,530,540,550,560,570,580,590)FNTYPE                 142
501    DTABLE(POINT-1)=SIN(DVALUE)                                          143
       GO TO 595                                                            144
510    DTABLE(POINT-1)=COS(DVALUE)                                          145
       GO TO 595                                                            146
520    DTABLE(POINT-1)=TAN(DVALUE)                                          147
       GO TO 595                                                            148
 530   DCHECK=DABS(DVALUE)                                                  149
       IF (DCHECK.GT.1.0) THEN                                              150
              CALL ERROR(LINE,202)                                          151
              GO TO 1000                                                    152
       ELSE                                                                 153
              CONTINUE                                                      154
       ENDIF                                                                155
       DTABLE(POINT-1)=ASIN(DVALUE)                                         156
       GO TO 595                                                            157
 540   DCHECK=DABS(DVALUE)                                                  158
       IF (DCHECK.GT.1.0) THEN                                              159
              CALL ERROR(LINE,202)                                          160
              GO TO 1000                                                    161
       ELSE                                                                 162
              CONTINUE                                                      163
       ENDIF                                                                164
       DTABLE(POINT-1)=ACOS(DVALUE)                                         165
       GO TO 595                                                            166
 550   DTABLE(POINT-1)=ATAN(DVALUE)                                         167
       GO TO 595                                                            168
 560   DTABLE(POINT-1)=DABS(DVALUE)                                         169
       GO TO 595                                                            170
 570   IF (DVALUE.LT.0.0) THEN                                              171
              CALL ERROR(LINE,203)                                          172
              GO TO 1000                                                    173
       ELSE                                                                 174
              CONTINUE                                                      175
       ENDIF                                                                176
       DTABLE(POINT-1)=DSQRT(DVALUE)                                        177
       GO TO 595                                                            178
 580   DTABLE(POINT-1)=DEXP(DVALUE)                                         179
       GO TO 595                                                            180
 590   DTABLE(POINT-1)=DLOG(DVALUE)                                         181
C                                                                           182
C       STORE IN TABLE CODE FOR NUMBER AND NUMBER TYPE                      183
C                                                                           184
 595   TABLE(POINT-1,1)=9                                                   185
       TABLE(POINT-1,3)=3                                                   186
 1000  RETURN                                                               187
       END                                                                  188
```

```
C*****************************   SUBROUTINE NUMBER   ********************
C
C       FORMAT  ------NUMBER(PLINE,WORD,VTYPE,IVALUE,RVALUE,DVALUE)
C
C       PURPOSE:        TO CONVERT A CHARACTER STRING INTO A VALUE BY
C                       READING THE DIGITS OR FINDING THE VALUE OF THE
C                       VARIABLE NAME
C
C       PARAMETERS:
C
C               I/O PARAMETERS:
C
C                       LINE:   THE LINE OF CODE FROM WHICH THE VALUE
C                               IS BEING SOUGHT
C                       WORD:   THE CHARACTER STRING THAT IS BE
C                               CONVERTED
C                       VTYPE:  THE TYPE OF VALUE FOUND
C                       IVALUE,RVALUE,DVALUE,CVALUE,LVALUE: PATHWAYS
C                               FOR INTEGER,REAL,DOUBLE PRECISION,
C                               CHARACTER AND LOGICAL VALUES TO BE
C                               PASSED TO AND FROM SUBROUTINES
C
C               OTHER PARAMETERS:
C
C                       DECMAL: THE LOCATION OF A DECMAL IN CHARACTER
C                               STRING BEING PARSED
C                       LOOK:   A LOCATION OF A CHARACTER IN THE
C                               CHARACTER STRING BEING PARSED
C                       COUNT:  LOOP PROTECTOR IN CASE OF A BLANK
C                               CHARACTER STRING BEING SENT
C                       DIG:    THE FIRST CHARACTER OF THE STRING SENT
C                       VARNAM  THE CHARACTER STRING SENT. COULD BE A
C                               NUMBER OR VARIABLE NAME
C                       DIGIT:  ARRAY OF THE 10 ALPHA NUMERIC DIGITS
C                       NUMFLG: FLAG TO INDICATE CHARACTER STRING SENT
C                               IS A NUMBER
C                       DECFLG: FLAG TO INDICATE THAT THE NUMBER IS A
C                               REAL NUMBER
C
C       ERRORS CALLED:
C               150
C               301
C               302
C
C       SUBROUTINES CALLED:
C               SARRAY
C               VARABL
C               ERROR
C
C       SUBROUTINES THAT CALL NUMBER:
C               ARITH
C               SREAD
C               SWRITE
C               INTERP
C               VAROUT
C               VARCHG
C               SPAUSE
C               EVALIF
C
C*******************************************************************************
C
      SUBROUTINE NUMBER(PLINE,WORD,VTYPE,IVALUE,RVALUE,DVALUE)
      INTEGER START,IVALUE,DECMAL,VTYPE,BUG,LOOK,COUNT,RSTART
      REAL RVALUE
      DOUBLE PRECISION DVALUE
      CHARACTER WORD*10,DIG*1,DIGIT(0:9)*1,VARNAM*6,PLINE*72,CVALUE,C*60
      LOGICAL NUMFLG,DECFLG,LVALUE
      COMMON/STATE/RSTART
      COMMON/CHECK/BUG
      DIGIT(0)='0'
      DIGIT(1)='1'
      DIGIT(2)='2'
      DIGIT(3)='3'
```

```
      DIGIT(4)='4'                                                          74
      DIGIT(5)='5'                                                          75
      DIGIT(6)='6'                                                          76
      DIGIT(7)='7'                                                          77
      DIGIT(8)='8'                                                          78
      DIGIT(9)='9'                                                          79
C                                                                           80
C               ELIMINATE LEADING BLANKS                                    81
C                                                                           82
        COUNT=0                                                             83
 100    BLANK=INDEX(WORD,' ')                                               84
        IF (BLANK .EQ. 1) THEN                                              85
                WORD=WORD(2:10)                                             86
                COUNT=COUNT+1                                               87
                IF(COUNT.GT.70)THEN                                         88
                        CALL ERROR(PLINE,150)                               89
                        GO TO 1000                                          90
                ELSE                                                        91
                        CONTINUE                                            92
                ENDIF                                                       93
                GO TO 100                                                   94
        ELSE                                                                95
                WORD=WORD(1:BLANK-1)                                        96
        ENDIF                                                               97
C                                                                           98
C               IS THE WORD AN ARRAY ELEMENT                                99
C                                                                          100
        LOOK=INDEX(WORD,'(')                                               101
        IF(LOOK.EQ.0) GO TO 120                                            102
        CALL SARRAY(2,WORD,TYPE,IVALUE,RVALUE,DVALUE,CVALUE,LVALUE)        103
        GO TO 1000                                                         104
C                                                                          105
C         READ 'WORD' BETWEEN OPERANDS                                     106
C                                                                          107
 120    NUMFLG=.FALSE.                                                     108
        DO 155 I=1,8                                                       109
              DIG=WORD(I:I)                                                110
              DO 150 J= 0,9                                                111
                   IF (DIG.EQ.DIGIT(J)) THEN                               112
                        NUMFLG=.TRUE.                                      113
                        GO TO 157                                          114
                   ELSE                                                    115
                        CONTINUE                                           116
                   ENDIF                                                   117
 150           CONTINUE                                                    118
               IF (DIG.NE.' ')GO TO 157                                    119
 155    CONTINUE                                                           120
 157    START=I                                                            121
         IF (NUMFLG) THEN                                                  122
C                                                                          123
C         'WORD' IS A NUMBER                                               124
C                                                                          125
            IVALUE=0                                                       126
            RVALUE=0.00                                                    127
            DECMAL=INDEX(WORD,'.')                                         128
            IF (DECMAL.EQ.1) THEN                                          129
                CALL ERROR(PLINE,301)                                      130
                GOTO 1000                                                  131
            ELSE                                                           132
                CONTINUE                                                   133
            ENDIF                                                          134
C                                                                          135
C       REAL NUMBER TO BE EVALUATED                                        136
C                                                                          137
            IF (DECMAL.NE.0) THEN                                          138
               DECFLG=.FALSE.                                              139
               VTYPE=2                                                     140
               DO 170 J= START,8                                           141
                  DIG=WORD(J:J)                                            142
                  IF(DIG.EQ.'.')DECFLG=.TRUE.                              143
                  IF (DECFLG)THEN                                          144
                        I=J-1                                              145
```

```
                ELSE                                                    146
                     I=J                                                147
                ENDIF                                                   148
                DO 160 K = 0,9                                          149
                IF(DIG.EQ.DIGIT(K))RVALUE=RVALUE+K*10.**(DECMAL-I-1)    150
 160            CONTINUE                                                151
 170          CONTINUE                                                  152
C                                                                       153
C       INTEGER TO BE EVALUATED                                         154
C                                                                       155
          ELSE                                                          156
            VTYPE=1                                                     157
            DO 190 J = START,10                                         158
               DIG=WORD(J:J)                                            159
               DO 180 K = 0,9                                           160
                  IF (DIG .EQ. DIGIT(K))IVALUE=IVALUE*10+K              161
 180              CONTINUE                                              162
 190           CONTINUE                                                 163
              ENDIF                                                     164
         ELSE                                                           165
C                                                                       166
C         'WORD' IS A VARIABLE OR FUNCTION NAME                         167
C                                                                       168
            VARNAM=WORD(START:8)                                        169
            CALL VARABL(2,VARNAM,VTYPE,IVALUE,RVALUE,DVALUE,C,L)        170
            IF(RSTART.NE.1)GO TO 1000                                   171
         ENDIF                                                          172
       IF (VTYPE.GT.3) CALL ERROR(PLINE,302)                            173
 1000  RETURN                                                           174
       END                                                              175
```

```
C*****************************   SUBROUTINE FORWRD   ********************
C
C       FORMAT  ------ FORWRD(WHAT,LINUM) -------
C
C       PURPOSE:        TO SCAN FORWARD THROUGH A PROGRAM TO FIND A
C                       STATEMENT NUMBER, AN ELSE, OR AN
C                       ENDIF STAATEMENT AND TO MAP DO AND IF
C                       STRUCTURES AS IT GOES
C
C       PARAMETERS:
C
C               I/O PARAMETERS:
C
C                       WHAT:   IS AN INTEGER CODE OF WHAT IS BEING
C                               SEARCHED FOR
C                                       1:  STATEMENT NUMBER
C                                       2:  ELSE STATEMENT
C                                       3:  ENDIF STATEMENT
C
C               OTHER PARAMETER:
C
C                       IVAL,R,D,C,L:  PATHWAYS FOR VALUES TO BE SENT
C                               TO AND FROM SUBROUTINE NUMBER, OF TYPES
C                               INTEGER, REAL, DOUBLE PRECISION,
C                       TYPE:   THE DATA TYPE RETURNED FOR SUBROUTINE
C                               NUMBER
C                       BLANK:  INTEGER LOCATION OF A BLANK USED IN
C                               PARSING
C                       COMMA:  INTEGER LOCATION OF A COMMA
C                       FOOT:   STATEMENT NUMBER OF DO FOOT
C                       EXPRES: CHARACTER STRING OF LINE BEING
C                               PARSED THAT IS SHORTENED WHEN PARSED
C                       NUM:    STATEMENT NUMBER OF LINE BEING ENTERED
C                       LINFLG: FLAG TO INDICATE A STATEMENT NUMBER
C                               EXISTS
C                       FINDFG: FLAG TO INDICATE THE ITEM SOUGHT
C                               HAS BEEN FOUND
C                       COMENT: INTEGER LOCATION OF CHARACTER 'C'
C
C               COMMON PARAMETERS:
C
C                       RSTART: A VARIABLE THATS VALUE INDICATES
C                               WHICH DIRECTION VMAIN IS TO TAKE:
C                                       1:  CONTINUE EXECUTION
C                                       2:  RESTART EXECUTION FROM START
C                                       3:  EXECUTE DIFFERENT PROGRAM
C                                       4:  STOP
C                       BRANCH: AN ARRAY TABLE THAT HOLDS THE MAP OF
C                               THE PROGRAM. USED FOR BRANCHING AND
C                               TO CHECK PROGRAM STRUCTURE
C                       BINDEX: THE ROW OF BRANCH THE NEXT ENTRY IS TO
C                               BE MADE ON
C                       DO:     INTEGER ARRAY THATS COLUMNS HOLD THE
C                                   STATEMENT NUMBER OF DO FOOT
C                                   THE LINE NUMBER OF DO STATMENT
C                                   NUMBER OF INTERATIONS LEFT IN LOOP
C                       REALDO: REAL ARRAY THATS COLUMNS HOLD THE
C                                   INITIAL VALUE OF DO INDEX
C                       DINDEX: THE ROW THAT THE NEXT ENTRY TO ARRAYS
C                               DO AND REALDO IS TO BE MADE IN
C                       FINDEX: THE ROW OF ARRAY DO AND REALDO THAT
C                               CORRISPONDES WITH THE PRESENT DO
C                               BEING CHECKED
C                       LINDEX: LINE NUMBER OF LINE READ FURTHEST IN
C                               PROGRAM
C                       LPOINT: LINE NUMBER OF THE LINE BEING INTERPRETE
C                               INTERPERED
C                       LEVEL:  THE CURRENT IF LEVEL
C                       MAXLIN: MAXIMUM NUMBER OF LINES ALLOWED IN THE
C                               USER'S PROGRAM
C                       SCLINE: A CHARACTER ARRAY OF THE CODE OF THE
C                               USER'S PROGRAM
```

```
C                                                                               73
C                                                                               74
C       ERRORS NUMBERS CALLED                                                   75
C              107                                                              76
C              303                                                              77
C              404                                                              78
C                                                                               79
C       SUBROUTINES CALLED:                                                     80
C              NUMBER                                                           81
C                                                                               82
C       SUBROUTINES THAT CALL FORWRD:                                           83
C              SIF                                                              84
C              SGOTO                                                            85
C                                                                               86
C*******************************************************************************87
        SUBROUTINE FORWRD(WHAT,LINUM)                                           88
        INTEGER LINDEX,LPOINT,WHAT,LINUM,LEVEL,FINDFG,COMENT,                   89
     $          BLANK,LINFLG,IVAL,TYPE,NUM,BINDEX,BRANCH,DO,COUNT               90
     $          DINDEX,FINDEX,LOOK,COMMA,FOOT,BUG,MAXLIN,SAVELX                 91
        REAL R,REALDO                                                           92
        DOUBLE PRECISION D                                                      93
        CHARACTER SCLINE*72,CLINE*72,LINE*72,EXPRES*72,WORD*6,HOLD*6            94
        COMMON/CONTRL/LINDEX,LPOINT,LEVEL,MAXLIN                                95
        COMMON/BLOCK2/BRANCH(20,6),BINDEX                                       96
        COMMON/DOBLK/DO(15,3),REALDO(15,2),DINDEX,FINDEX                        97
        COMMON/CONTR2/SCLINE(50)                                                98
        COMMON/STATE/RSTART                                                     99
        COMMON/CHECK/BUG                                                       100
        FINDFG=0                                                               101
        SAVELX=LINDEX                                                          102
 10     LINDEX=LINDEX+1                                                        103
        LINFLG=0                                                               104
        READ(100,210,END=950)SCLINE(LINDEX)                                    105
        CLINE=SCLINE(LINDEX)                                                   106
C                                                                              107
C               CHECK FOR COMMENT LINE                                         108
C                                                                              109
        COMENT=INDEX(CLINE,'C')                                                110
        IF(COMENT.EQ.1) GO TO 10                                               111
C                                                                              112
C               LOOK FOR LINE NUMBER                                           113
C                                                                              114
        EXPRES=CLINE(1:72)                                                     115
        DO 15 J=1,5                                                            116
                BLANK=INDEX(EXPRES,' ')                                        117
                IF(BLANK.NE.1) THEN                                            118
                        LINFLG=1                                               119
                        WORD=CLINE(1:5)                                        120
                        CALL NUMBER(CLINE,WORD,TYPE,IVAL,R,D)                  121
                        IF(RSTART.NE.1)GO TO 1000                              122
                        IF(TYPE.NE.1) THEN                                     123
                                CALL ERROR(CLINE,303)                          124
                                GO TO 1000                                     125
                        ELSE                                                   126
                                CONTINUE                                       127
                        ENDIF                                                  128
                        NUM=IVAL                                               129
                        GO TO 30                                               130
                ELSE                                                           131
                        EXPRES=CLINE(J:72)                                     132
                ENDIF                                                          133
 15     CONTINUE                                                               134
        GO TO 100                                                              135
 30     BINDEX=BINDEX+1                                                        136
        BRANCH(BINDEX,1)=NUM                                                   137
        BRANCH(BINDEX,2)=LINDEX                                                138
        BRANCH(BINDEX,5)=BRANCH(BINDEX-1,5)                                    139
        BRANCH(BINDEX,6)=BRANCH(BINDEX-1,6)                                    140
C                                                                              141
```

```
C               IS THIS LINE NUMBER THE ONE WE ARE LOOKING FOR
C
      IF(WHAT.NE.1) GO TO 40
      IF(LINUM.NE.NUM) GO TO 40
      FINDFG=1
C
C               IS THIS LINE A 'DO' FOOT
C
 40   IF(LINFLG.EQ.0) GO TO 100
      DO 50 I=1,DINDEX
              IF(DO(I,1).EQ.NUM) GO TO 70
 50   CONTINUE
      GO TO 100
 70   BRANCH(BINDEX,3)=2
      BRANCH(BINDEX,4)=0
      BRANCH(BINDEX,5)=BRANCH(BINDEX-1,5)-1
      BRANCH(BINDEX,6)=BRANCH(BINDEX-1,6)
      IF(FINDFG.EQ.1) GO TO 1000
      GO TO 900
C
C               ELEMINANT BLANK CHARACTERS FROM BEGINNING OF LINE
C
 100  LINE=CLINE(7:72)
      COUNT=0
 110  POINT=INDEX(LINE,' ')
      IF (POINT .EQ. 1) THEN
              LINE=LINE(2:72)
              COUNT=COUNT+1
              IF(COUNT.GT.80)THEN
                      CALL ERROR(LINE,150)
                      GO TO 1000
              ELSE
                      CONTINUE
              ENDIF
              GO TO 110
      ELSE
              CONTINUE
      ENDIF
C
C               IS THIS LINE A 'IF' STATEMENT
C
 200  LOOK=INDEX(LINE,'IF')
      IF(LOOK.NE.1) GO TO 300
      LOOK=INDEX(LINE,'THEN')
      IF(LOOK.EQ.0)GO TO 900
      IF(LINFLG.EQ.0)BINDEX=BINDEX+1
      BRANCH(BINDEX,2)=LPOINT
      BRANCH(BINDEX,3)=3
      BRANCH(BINDEX,5)=BRANCH(BINDEX-1,5)
      BRANCH(BINDEX,6)=BRANCH(BINDEX-1,6)+1
      IF(FINDFG.EQ.1)GO TO 1000
      GO TO 900
C
C               IS THIS LINE AN 'ELSE' STATEMENT
C
 300  LOOK=INDEX(LINE,'ELSE')
      IF(LOOK.NE.1) GO TO 400
C
C               RECORD ELSE STATEMENT
C
      LPOINT=LINDEX
      IF(LPOINT.GT.BRANCH(BINDEX,2))THEN
              BINDEX=BINDEX+1
              BRANCH(BINDEX,2)=LPOINT
              BRANCH(BINDEX,3)=4
              BRANCH(BINDEX,5)=BRANCH(BINDEX-1,5)
              BRANCH(BINDEX,6)=BRANCH(BINDEX-1,6)
      ELSE IF(LPOINT.EQ.BRANCH(BINDEX,2))THEN
              BRANCH(BINDEX,3)=4
              BRANCH(BINDEX,5)=BRANCH(BINDEX-1,5)
              BRANCH(BINDEX,6)=BRANCH(BINDEX-1,6)
      ELSE
```

```
              CONTINUE                                                  214
            ENDIF                                                       215
            IF(WHAT.EQ.2.AND.LEVEL.EQ.BRANCH(BINDEX,6))FINDFG=1         216
            IF(FINDFG.EQ.1) GO TO 1000                                  217
            GO TO 900                                                   218
C                                                                       219
C                   IS THIS LINE A 'ENDIF' STATEMENT (FINDFG)           220
C                                                                       221
 400        LOOK=INDEX(LINE,'ENDIF')                                    222
            IF(LOOK.NE.1) GO TO 500                                     223
            IF(WHAT.EQ.3.AND.LEVEL.EQ.BRANCH(BINDEX,6))FINDFG=1         224
            LPOINT=LINDEX                                               225
            IF(FINDFG.EQ.1)GO TO 1000                                   226
            GO TO 900                                                   227
C                                                                       228
C                   IS THIS LINE A 'DO' STATEMENT                       229
C                                                                       230
 500        LOOK=INDEX(LINE,'DO ')                                      231
            IF(LOOK.NE.1) GO TO 600                                     232
            EXPRES=LINE(LOOK+2:72)                                      233
            BLANK=INDEX(EXPRES,' ')                                     234
            IF(BLANK.NE.1) THEN                                         235
                    CALL ERROR(LINE,107)                                236
                    GO TO 1000                                          237
            ELSE                                                        238
                    CONTINUE                                            239
            ENDIF                                                       240
            EXPRES=EXPRES(2:72)                                         241
            BLANK=INDEX(EXPRES,' ')                                     242
            COMMA=INDEX(EXPRES,',')                                     243
            IF (COMMA .LT. BLANK) THEN                                  244
                    HOLD=EXPRES(1:COMMA-1)                              245
            ELSE                                                        246
                    HOLD=EXPRES(1:BLANK-1)                              247
            ENDIF                                                       248
            CALL NUMBER(LINE,HOLD,TYPE,I,R,D)                           249
            IF(RSTART.NE.1)GO TO 1000                                   250
            IF(TYPE.NE.1)THEN                                           251
                    CALL ERROR(LINE,303)                                252
                    GO TO 1000                                          253
            ELSE                                                        254
                    CONTINUE                                            255
            ENDIF                                                       256
            FOOT=I                                                      257
            IF(LINDEX.LE.DO(DINDEX,2)) GO TO 550                        258
            DINDEX=DINDEX+1                                             259
            DO(DINDEX,1)=FOOT                                           260
            DO(DINDEX,2)=LINDEX                                         261
 550        IF(LINFLG.EQ.0)BINDEX=BINDEX+1                              262
            BRANCH(BINDEX,2)=LINDEX                                     263
            BRANCH(BINDEX,3)=1                                          264
            BRANCH(BINDEX,4)=FOOT                                       265
            BRANCH(BINDEX,5)=BRANCH(BINDEX-1,5)+1                       266
            BRANCH(BINDEX,6)=BRANCH(BINDEX-1,6)                         267
            IF(FINDFG.EQ.1)GO TO 1000                                   268
            GO TO 900                                                   269
C                                                                       270
C                   CHECK FOR END OF PROGRAM                            271
C                                                                       272
 600        LOOK=INDEX(LINE,'END ')                                     273
            IF(LOOK.EQ.1) THEN                                          274
                    CALL ERROR(LINE,404)                                275
                    GO TO 1000                                          276
            ELSE                                                        277
                    CONTINUE                                            278
            ENDIF                                                       279
            IF(FINDFG.EQ.1)GO TO 1000                                   280
 900        GO TO 10                                                    281
 950        CALL ERROR(SCLINE(SAVELX),404)                              282
1000        RETURN                                                      283
 210        FORMAT(A72)                                                 284
            END                                                         285
```

```
C*****************************   SUBROUTINE DOBEGN   ********************
C
C       FORMAT  ------ DOBEGN(LINE,LPOINT,SETFLG) -------
C
C       PURPOSE:        TO PARSE DO STATEMENT AND RECORD INFORMATION
C                               IN APPROPRATE TABLES
C
C       PARAMETERS:
C
C               I/O PARAMETERS:
C
C                       LINE:   THE LINE OF CODE BEING PARSED
C                       LPOINT: THE LINE NUMBER OF THE LINE BEING
C                               CHECKED
C                       SETFLG: FLAG TO INDICATE NEW EXECUTION AND
C                               TO CLEAR ALL TABLES AND REGISTERS
C
C               OTHER PARAMETER:
C
C                       I,R,D,C,L:  PATHWAYS FOR VALUES TO BE SENT
C                               TO AND FROM SUBROUTINE NUMBER, OF TYPES
C                               INTEGER, REAL, DOUBLE PRECISION,
C                               CHARACTER AND LOGICAL,RESPECTIVELY
C                       TYPE:   THE DATA TYPE RETURNED FOR SUBROUTINE
C                               NUMBER
C                       POINT:  POINTER USED TO SHOW HOW FAR PARSING
C                               HAS OCCURED
C                       EQUALS: INTEGER LOCATION OF AN EQUALS SIGN
C                       BLANK:  INTEGER LOCATION OF A BLANK USED IN
C                               PARSING
C                       COMMA,COMMA2: INTEGER LOCATION OF A COMMA
C                       LINEND: FLAG TO INDICATE END IF LINE
C                       FOOT:   STATEMENT NUMBER OF DO FOOT
C                       EXPRES: CHARACTER STRING OF LINE BEING
C                               PARSED THAT IS SHORTENED WHEN PARSED
C                       START:  CHARACTER STRING OF INITIAL DO VALUE
C                       LAST:   CHARACTER STRING OF FINAL DO VALUE
C                       STEP:   CHARACTER STRING OF INCREMENTAL VALUE
C
C               COMMON PARAMETERS:
C
C                       RSTART: A VARIBLE THATS VALUE INDICATES
C                               WHICH DIRECTION VMAIN IS TO TAKE:
C                                       1:  CONTINUE EXECUTION
C                                       2:  RESTART EXECUTION FROM START
C                                       3:  EXECUTE DIFFERENT PROGRAM
C                                       4:  STOP
C                       BRANCH: AN ARRAY TABLE THAT HOLDS THE MAP OF
C                               THE PROGRAM. USED FOR BRANCHING AND
C                               TO CHECK PROGRAM STRUCTURE
C                       BINDEX: THE ROW OF BRANCH THE NEXT ENTRY IS TO
C                               BE MADE ON
C                       DO:     INTEGER ARRAY THATS COLUMNS HOLD THE
C                                   STATEMENT NUMBER OF DO FOOT
C                                   THE LINE NUMBER OF DO STATMENT
C                                   NUMBER OF INTERATIONS LEFT IN LOOP
C                       REALDO: REAL ARRAY THATS COLUMNS HOLD THE
C                                   INITIAL VALUE OF DO INDEX
C                       DINDEX: THE ROW THAT THE NEXT ENTRY TO ARRAYS
C                               DO AND REALDO IS TO BE MADE IN
C                       FINDEX: THE ROW OF ARRAY DO AND REALDO THAT
C                               CORRISPONDES WITH THE PRESENT DO
C                               BEING CHECKED
C                       CONTER: THE DO INDEX
C
C       ERRORS NUMBERS CALLED
C               106
C               107
C               211
C
```

```
C       SUBROUTINES CALLED:                                                    71
C               NUMBER                                                         72
C               VARABL                                                         73
C                                                                              74
C       SUBROUTINES THAT CALL DOBEGN:                                          75
C               VMAIN                                                          76
C               SEARCH                                                         77
C                                                                              78
C*************************************************************************     79
C                                                                              80
        SUBROUTINE DOBEGN(LINE,LPOINT,SETFLG)                                  81
        INTEGER POINT,BLANK,EQUALS,COMMA,COMMA2,LINEND,DINDEX,BINDEX,          82
     $          BUG,FOOT,I,TYPE,LPOINT,SETFLG,DO,BRANCH,RSTART                 83
        REAL R,INT,INC,FINAL,REALDO                                            84
        DOUBLE PRECISION D                                                     85
        CHARACTER LINE*72,EXPRES*72,CONTER*9,START*10,LAST*10,                 86
     $          HOLD*10,STEP*10,C,HOLDCT*9                                     87
        LOGICAL L                                                              88
        COMMON/DOBLK/DO(15,3),REALDO(15,2),DINDEX,FINDEX                       89
        COMMON/BLOCK2/BRANCH(20,6),BINDEX                                      90
        COMMON/BLOCKC/CONTER(15)                                               91
        COMMON/STATE/RSTART                                                    92
        COMMON/CHECK/BUG                                                       93
C                                                                              94
C               CLEAR ALL TABLES TO BEGIN NEW PROGRAM                          95
C                                                                              96
        IF(SETFLG.EQ.1)THEN                                                    97
                DINDEX=0                                                       98
                DO 200 I=1,20                                                  99
                        DO 100 J=1,5                                          100
                        DO(I,J)=0                                             101
 100                    CONTINUE                                              102
 200            CONTINUE                                                      103
                GO TO 1000                                                    104
        ELSE                                                                  105
                CONTINUE                                                      106
        ENDIF                                                                 107
        IF(LPOINT.LE.DO(DINDEX,2))GO TO 210                                   108
        DINDEX=DINDEX+1                                                       109
C                                                                             110
C               PARSE THE DO STATEMENT                                        111
C                                                                             112
 210    POINT=INDEX(LINE,'DO')                                                113
        IF(POINT .NE. 1) THEN                                                 114
                CALL ERROR(LINE,106)                                          115
                GO TO 1000                                                    116
        ELSE                                                                  117
                CONTINUE                                                      118
        ENDIF                                                                 119
        EXPRES=LINE(POINT+2:72)                                               120
        BLANK=INDEX(EXPRES,' ')                                               121
        IF(BLANK.NE.1) THEN                                                   122
                CALL ERROR(LINE,107)                                          123
                GO TO 1000                                                    124
        ELSE                                                                  125
                CONTINUE                                                      126
        ENDIF                                                                 127
        EXPRES=EXPRES(2:72)                                                   128
        BLANK=INDEX(EXPRES,' ')                                               129
        COMMA=INDEX(EXPRES,',')                                               130
        IF (COMMA .LT. BLANK) THEN                                            131
                HOLD=EXPRES(1:COMMA-1)                                        132
                POINT=COMMA+1                                                 133
        ELSE                                                                  134
                HOLD=EXPRES(1:BLANK-1)                                        135
                POINT=BLANK+1                                                 136
        ENDIF                                                                 137
        CALL NUMBER(LINE,HOLD,TYPE,I,R,D)                                     138
        IF(RSTART.NE.1)GO TO 1000                                             139
        FOOT=I                                                                140
```

```
      EXPRES=EXPRES(POINT:72)                                          141
      EQUALS=INDEX(EXPRES,'=')                                         142
      HOLDCT=EXPRES(1:EQUALS-1)                                        143
      COMMA=INDEX(EXPRES,',')                                          144
      START=EXPRES(EQUALS+1:COMMA-1)                                   145
      CALL NUMBER(LINE,START,TYPE,I,R,D)                               146
      IF(RSTART.NE.1)GO TO 1000                                        147
      IF(TYPE.GT.2) THEN                                               148
              CALL ERROR(LINE,211)                                     149
              GO TO 1000                                               150
      ELSE                                                             151
              CONTINUE                                                 152
      ENDIF                                                            153
      IF(TYPE.EQ.1)THEN                                                154
              INT=I                                                    155
      ELSE                                                             156
              INT=R                                                    157
      ENDIF                                                            158
      EXPRES=EXPRES(COMMA+1:72)                                        159
      COMMA2=INDEX(EXPRES,',')                                         160
      IF(COMMA2 .GT. 0 ) THEN                                          161
              LAST=EXPRES(1:COMMA2-1)                                  162
              CALL NUMBER(LINE,LAST,TYPE,I,R,D)                        163
              IF(RSTART.NE.1)GO TO 1000                                164
              IF(TYPE.GT.2) THEN                                       165
                      CALL ERROR(LINE,211)                             166
                      GO TO 1000                                       167
              ELSE                                                     168
                      CONTINUE                                         169
              ENDIF                                                    170
              IF(TYPE.EQ.1) THEN                                       171
                      FINAL=I                                          172
              ELSE                                                     173
                      FINAL=R                                          174
              ENDIF                                                    175
              EXPRES=EXPRES(COMMA2+1:72)                               176
              LINEND=INDEX(EXPRES,' ')                                 177
              STEP=EXPRES(1:LINEND-1)                                  178
              CALL NUMBER(LINE,STEP,TYPE,I,R,D)                        179
              IF(RSTART.NE.1)GO TO 1000                                180
              IF(TYPE.GT.2) THEN                                       181
                      CALL ERROR(LINE,211)                             182
                      GO TO 1000                                       183
              ELSE                                                     184
                      CONTINUE                                         185
              ENDIF                                                    186
              IF(TYPE.EQ.1)THEN                                        187
                      INC=I                                            188
              ELSE                                                     189
                      INC=R                                            190
              ENDIF                                                    191
       ELSE                                                            192
              LINEND=INDEX(EXPRES,' ')                                 193
              LAST=EXPRES(1:LINEND-1)                                  194
              CALL NUMBER(LINE,LAST,TYPE,I,R,D)                        195
              IF(RSTART.NE.1)GO TO 1000                                196
              IF(TYPE.GT.2) THEN                                       197
                      CALL ERROR(LINE,211)                             198
                      GO TO 1000                                       199
              ELSE                                                     200
                      CONTINUE                                         201
              ENDIF                                                    202
              IF(TYPE.EQ.1) THEN                                       203
                      FINAL=I                                          204
              ELSE                                                     205
                      FINAL=R                                          206
              ENDIF                                                    207
              INC=1                                                    208
       ENDIF                                                           209
C                                                                      210
```

```
C               FIND CORRECT FINDEX                                     211
C                                                                       212
      DO 300 I=1,DINDEX                                                 213
             IF(DO(I,1).EQ.FOOT) THEN                                   214
                    FINDEX=I                                            215
                    GO TO 400                                           216
             ELSE                                                       217
                    FINDEX=DINDEX                                       218
             ENDIF                                                      219
 300   CONTINUE                                                         220
 400   CONTER(FINDEX)=HOLDCT                                            221
       ITER=(FINAL-INT+INC)/INC                                         222
       REALDO(FINDEX,1)=INT                                             223
       REALDO(FINDEX,2)=INC                                             224
       CALL VARABL(1,CONTER(FINDEX),2,I,INT,D,C,L)                      225
       IF(RSTART.NE.1)GO TO 1000                                        226
       DO(DINDEX,1)=FOOT                                                227
       DO(FINDEX,2)=LPOINT                                              228
       DO(FINDEX,3)=ITER                                                229
C                                                                       230
C              PLACE INFO IN BRANCH TABLE                               231
C                                                                       232
       IF(LPOINT.LT.BRANCH(BINDEX,2))GO TO 1000                         233
       IF(LPOINT.GT.BRANCH(BINDEX,2))BINDEX=BINDEX+1                    234
       BRANCH(BINDEX,2)=LPOINT                                          235
       BRANCH(BINDEX,3)=1                                               236
       BRANCH(BINDEX,4)=FOOT                                            237
       BRANCH(BINDEX,5)=BRANCH(BINDEX-1,5)+1                            238
       BRANCH(BINDEX,6)=BRANCH(BINDEX-1,6)                              239
 1000  RETURN                                                           240
       END                                                              241
```

```
C*************************   SUBROUTINE DCHECK   ********************
C
C       FORMAT  ------ DCHECK(SCLINE,LPOINT,DOFLG,ENTRY)
C
C       PURPOSE:        TO SEE IF A LINE IS A DO FOOT AND IF SO TO:
C                               MAKE SURE STATEMENT IS LEGAL
C                               RECORD INFORMATION IN BRANCH TABLE
C                               CHECK DO STRUCTURE
C       PARAMETERS:
C
C               I/O PARAMETERS:
C
C                       SCLINE: THE LINE OF CODE BEING CHECKED
C                       LPOINT: THE LINE NUMBER OF THE LINE BEING
C                               CHECKED
C                       DOFLG:  FLAG TO INDICATE IF LINE IS A DO
C                               FOOT
C                       ENTRY:  LINE NUMBER OF BEGINNING OF DO LOOP
C
C               OTHER PARAMETER:
C
C                       I,R,D:  PATHWAYS FOR VALUES TO BE SENT
C                               TO AND FROM SUBROUTINE NUMBER, OF TYPES
C                               INTEGER, REAL, DOUBLE PRECISION
C                       TYPE:   THE DATA TYPE RETURNED FOR SUBROUTINE
C                               NUMBER
C                       NUM:    THE LINE NUMBER OF STATEMENT BEING
C                               INTERPRETED
C                       TEST:   INTEGER LOCATION OF A CHARACTER STRING
C                               BEING CHECKED FOR
C                       BLANK:  INTEGER LOCATION OF A BLANK USED IN
C                               PARSING
C
C               COMMON PARAMETERS:
C
C                       RSTART: A VARIABLE THATS VALUE INDUCATES
C                               WHICH DIRECTION VMAIN IS TO TAKE:
C                                    1:  CONTINUE EXECUTION
C                                    2:  RESTART EXECUTION FROM START
C                                    3:  EXECUTE DIFFERENT PROGRAM
C                                    4:  STOP
C                       BRANCH: AN ARRAY TABLE THAT HOLDS THE MAP OF
C                               THE PROGRAM. USED FOR BRANCHING AND
C                               TO CHECK PROGRAM STRUCTURE
C                       BINDEX: THE ROW OF BRANCH THE NEXT ENTRY IS TO
C                               BE MADE ON
C                       DO:     INTEGER ARRAY THATS COLUMNS HOLD THE
C                                   STATEMENT NUMBER OF DO FOOT,
C                                   THE LINE NUMBER OF DO STATMENT,
C                                   NUMBER OF INTERATIONS LEFT IN LOOP
C                       REALDO: REAL ARRAY THATS COLUMNS HOLD THE
C                                   INITIAL VALUE OF DO INDEX
C                       DINDEX: THE ROW THAT THE NEXT ENTRY TO ARRAYS
C                               DO AND REALDO IS TO BE MADE IN
C                       FINDEX: THE ROW OF ARRAY DO AND REALDO THAT
C                               CORRISPONDES WITH THE PRESENT DO
C                               BEING CHECKED
C                       CONTER: THE DO INDEX
C
C       ERRORS NUMBERS CALLED
C                150
C                205
C                206
C                207
C                208
C                209
C                210
C                303
C
C       SUBROUTINES CALLED:
C                NUMBER
C
```

```
C       SUBROUTINES THAT CALL DCHECK:                                          73
C              VMAIN                                                           74
C                                                                              75
C*************************************************************************     76
C                                                                              77
        SUBROUTINE DCHECK(SCLINE,LPOINT,DOFLG,ENTRY)                           78
        INTEGER LPOINT,TYPE,I,NUM,DINDEX,DOFLG,FINDEX,ENTRY,DO(15,3),          79
     $             BLANK,TEST,BRANCH,BINDEX,RSTART,BUG                         80
        REAL R,REALDO(15,2)                                                    81
        DOUBLE PRECISION D                                                     82
        CHARACTER SCLINE*72,WORD*10,LINE*72,TESTLN*72,HOLD*10,CLINE*72,        83
     $                  CONTER*9                                              84
        DIMENSION SCLINE(15)                                                   85
        COMMON/DOBLK/DO,REALDO,DINDEX,FINDEX                                   86
        COMMON/BLOCK2/BRANCH(20,6),BINDEX                                      87
        COMMON/BLOCKC/CONTER(15)                                               88
        COMMON/STATE/RSTART                                                    89
        COMMON/CHECK/BUG                                                       90
        CLINE=SCLINE(LPOINT)                                                   91
        WORD=CLINE(1:5)                                                        92
        CALL NUMBER(CLINE,WORD,TYPE,I,R,D)                                     93
        IF(RSTART.NE.1)GO TO 1000                                              94
        IF(TYPE.NE.1) THEN                                                     95
                CALL ERROR(CLINE,303)                                          96
                GO TO 1000                                                     97
        ELSE                                                                   98
                CONTINUE                                                       99
        ENDIF                                                                 100
        NUM=I                                                                 101
C                                                                             102
C               IS THIS LINE A DO-LOOP FOOT                                   103
C                                                                             104
        DO 100 I=1,DINDEX                                                     105
                IF (DO(I,1) .EQ. NUM)GO TO 150                                106
  100   CONTINUE                                                              107
        DOFLG=0                                                               108
        GO TO 1000                                                            109
C                                                                             110
C               FIND CORRECT FINDEX                                           111
C                                                                             112
  150   DOFLG=1                                                               113
        DO 170 J=1,DINDEX                                                     114
                IF(NUM.EQ.DO(J,1))THEN                                        115
                        FINDEX=J                                              116
                        GO TO 200                                             117
                ELSE                                                          118
                        CONTINUE                                              119
                ENDIF                                                         120
  170   CONTINUE                                                            121
C                                                                           122
C               ELIMINATE LEADING BLANKS                                    123
C                                                                           124
        LINE=CLINE(7:72)                                                    125
  200   BLANK=INDEX(LINE,' ')                                               126
        IF (BLANK .EQ. 1) THEN                                              127
                LINE=LINE(2:72)                                             128
                GO TO 200                                                   129
        ELSE                                                                130
                CONTINUE                                                    131
        ENDIF                                                               132
C                                                                           133
C               IS THE FOOT STATEMENT LEGAL                                 134
C                                                                           135
        TEST=INDEX(LINE,'GO ')                                              136
        IF(TEST .EQ. 1)CALL ERROR(CLINE,205)                                137
        TEST=INDEX(LINE,'GOTO ')                                            138
        IF(TEST .EQ. 1) THEN                                                139
                CALL ERROR(CLINE,205)                                       140
                GO TO 1000                                                  141
        ELSE                                                                142
                CONTINUE                                                    143
        ENDIF                                                               144
```

```
      TEST=INDEX(LINE,'RETURN ')                                              145
      IF(TEST .EQ. 1) THEN                                                    146
             CALL ERROR(CLINE,206)                                            147
             GO TO 1000                                                       148
      ELSE                                                                    149
             CONTINUE                                                         150
      ENDIF                                                                   151
      TEST=INDEX(LINE,'END ')                                                 152
      IF(TEST .EQ. 1) THEN                                                    153
             CALL ERROR(CLINE,207)                                            154
             GO TO 1000                                                       155
      ELSE                                                                    156
             CONTINUE                                                         157
      ENDIF                                                                   158
      TEST=INDEX(LINE,'STOP ')                                                159
      IF(TEST .EQ. 1) THEN                                                    160
             CALL ERROR(CLINE,208)                                            161
             GO TO 1000                                                       162
      ELSE                                                                    163
             CONTINUE                                                         164
      ENDIF                                                                   165
C                                                                             166
C            PLACE INFO INTO BRANCH TABLE                                     167
C                                                                             168
      IF(LPOINT.LT.BRANCH(BINDEX,2))GO TO 300                                 169
      IF(BRANCH(BINDEX,1).NE.NUM.OR.BRANCH(BINDEX,2).NE.LPOINT)THEN           170
             CALL ERROR(CLINE,150)                                            171
             GO TO 1000                                                       172
      ELSE                                                                    173
             CONTINUE                                                         174
      ENDIF                                                                   175
C                                                                             176
      BRANCH(BINDEX,3)=2                                                      177
      BRANCH(BINDEX,4)=0                                                      178
      BRANCH(BINDEX,5)=BRANCH(BINDEX-1,5)-1                                   179
      BRANCH(BINDEX,6)=BRANCH(BINDEX-1,6)                                     180
C                                                                             181
C            CHECK DO AND IF RANGES                                           182
C                                                                             183
 300  IF(BRANCH(BINDEX,6).NE.0) THEN                                          184
             CALL ERROR(CLINE,210)                                            185
             GO TO 1000                                                       186
      ELSE                                                                    187
             CONTINUE                                                         188
      ENDIF                                                                   189
      DLEVEL=BRANCH(BINDEX,5)                                                 190
      DO 350 I=BINDEX-1,0,-1                                                  191
        IF(BRANCH(I,5).LT.DLEVEL) THEN                                        192
             CALL ERROR(CLINE,209)                                            193
             GO TO 1000                                                       194
      ELSE                                                                    195
             CONTINUE                                                         196
      ENDIF                                                                   197
        IF(BRANCH(I,5).EQ.DLEVEL)THEN                                         198
           IF(BRANCH(I+1,4).NE.BRANCH(BINDEX,1))CALL ERROR(CLINE,209)         199
           GO TO 1000                                                         200
        ELSE                                                                  201
           CONTINUE                                                           202
        ENDIF                                                                 203
 350  CONTINUE                                                                204
 1000 RETURN                                                                  205
      END                                                                     206
```

```
C*****************************   SUBROUTINE DOFOOT   ********************
C
C       FORMAT  ------ DOFOOT(DOFLG,ENTRY) ------
C
C       PURPOSE:        TO DECREMENT INCREMENTAL COUNTER OF DO LOOP,
C                       CHECK TO SEE IF LOOP NEEDS TO BE REPEATED
C                       AND FIGURE NEW VALUE OF DO INDEX
C       PARAMETERS:
C
C               I/O PARAMETERS:
C
C                       DOFLG:  FLAG TO INDICATE IF LOOP NEEDS
C                               REPEATING OR NOT
C                       ENTRY:  LINE NUMBER OF BEGINNING OF DO LOOP
C
C               OTHER PARAMETER:
C
C                       I,R,D,C,L:  PATHWAYS FOR VALUES TO BE SENT
C                               TO AND FROM SUBROUTINE VARABL, OF TYPES
C                               INTEGER, REAL,DOUBLE PRECISION, CHARACTE
C                               AND LOGICAL, RESPECTIVELY
C
C               COMMON PARAMETERS:
C
C                       DO:     INTEGER ARRAY THATS COLUMNS HOLD THE
C                                   STATEMENT NUMBER OF DO FOOT
C                                   THE LINE NUMBER OF DO STATMENT
C                                   NUMBER OF ITERATIONS LEFT IN LOOP
C                       REALDO: REAL ARRAY THATS COLUMNS HOLD THE
C                                   INITIAL VALUE OF DO INDEX
C                       DINDEX: THE ROW THAT THE NEXT ENTRY TO ARRAYS
C                               DO AND REALDO IS TO BE MADE IN
C                       FINDEX: THE ROW OF ARRAY DO AND REALDO THAT
C                               CORRISPONDES WITH THE PRESENT DO
C                               BEING CHECKED
C                       CONTER: THE DO INDEX
C
C       ERRORS NUMBERS CALLED
C               NONE
C
C       SUBROUTINES CALLED:
C               VARABL
C
C       SUBROUTINES THAT CALL DOFOOT:
C               VMAIN
C
C*******************************************************************************
C
        SUBROUTINE DOFOOT(DOFLG,ENTRY)
        INTEGER DINDEX,DOFLG,FINDEX,ENTRY,DO,BUG,I
        REAL REALDO,R
        CHARACTER CONTER*9,C*60
        LOGICAL L
        DOUBLE PRECISION D
        COMMON/DOBLK/DO(15,3),REALDO(15,2),DINDEX,FINDEX
        COMMON/BLOCKC/CONTER(15)
        COMMON/CHECK/BUG
```

```
C                                                                       58
C               UPDATE ITERATION VALUE                                  59
C                                                                       60
      DO(FINDEX,3)=DO(FINDEX,3)-1                                       61
C                                                                       62
C               IS DO LOOP STILL ALIVE                                  63
C                                                                       64
      IF(DO(FINDEX,3).LT.1) THEN                                        65
              DOFLG=0                                                   66
              GO TO 1000                                                67
      ELSE                                                              68
              CONTINUE                                                  69
      ENDIF                                                             70
C                                                                       71
C               UPDATE LOOP PARAMETERS AND FIND BEGINNING OF LOOP       72
C                                                                       73
      ENTRY=DO(FINDEX,2)                                                74
      REALDO(FINDEX,1)=REALDO(FINDEX,1)+REALDO(FINDEX,2)                75
      CALL VARABL(1,CONTER(FINDEX),2,I,REALDO(FINDEX,1),D,C,L)          76
 1000 RETURN                                                            77
      END                                                               78
```

```
C*****************************   SUBROUTINE SIF    **********************      1
C                                                                               2
C       FORMAT  ------ SIF(LINE,PATH)                                           3
C                                                                               4
C       PURPOSE:        TO EXECUTE AN IF STATEMENT, RECORD IF,ELSE AND          5
C                       ENDIF STATEMENTS IN BRANCH TABLE AND TO EXECUTE         6
C                       ENDIF STATEMENTS.                                       7
C                                                                               8
C       PARAMETERS:                                                             9
C                                                                              10
C               I/O PARAMETERS:                                                11
C                                                                              12
C                       LINE:   THE LINE OF CODE THAT CAUSED THE               13
C                               SUBROUTINE TO BE CALLED                        14
C                       PATH:   CODE OF WHAT STATEMENT TYPE THE                15
C                               SUBROUTINE WAS CALLED FROM                     16
C                                       1 :     IF statement                   17
C                                       2 :     ELSE statement                 18
C                                       3 :     ENDIF statement                19
C                                                                              20
C               PARSING PARAMETERS:                                            21
C                                                                              22
C                       BLANK:  INTEGER LOCATION OF A BLANK                    23
C                       LLOOK,RLOOK: INTEGER LOCATION OF OPEN AND              24
C                               CLOSE PARENTHESES                              25
C                       EXPRES: LINE OF CODE THAT IS SHORTENEND AS IT          26
C                               IS PARSED                                      27
C                       LOOK:   INTEGER VALUE OF LOCATION OF ELEMENT           28
C                               BEING PARSED FOR                               29
C                                                                              30
C               FOR EVALUATING IF STATEMENT                                    31
C                                                                              32
C                       LOGEXP: CHARACTER STRING OF THE LOGICAL                33
C                               EXPRESSION TO BE EVALUATED                     34
C                       TRUTH:  LOGICAL VARIABLE THAT RETURNS THE              35
C                               VALUE OF A LOGICAL EXPRESSION                  36
C                                                                              37
C               COMMON PARAMETERS:                                             38
C                                                                              39
C                       LINDEX: THE LINE NUMBER OF THE MOST ADVANCED           40
C                               LINE READ                                      41
C                       LPOINT: THE LINE NUMBER OF THE LINE BEING              42
C                               INTERPRETED                                    43
C                       LEVEL:  THE LEVEL OR DEPTH OF NESTING OF THE           44
C                               IF STRUCTURE                                   45
C                       SCLINE: CHATACTER ARRAY OF PROGRAM BEING               46
C                               INTERPRETED                                    47
C                       BRANCH: A TABLE THAT MAPS THE CODE BEING               48
C                               INTERPRETED                                    49
C                       BINDEX: WHAT LINE THE NEXT ENTRY TO BRANCH IS          50
C                               TO BE ON                                       51
C                       IFTABL: AN ARRAY THAT INDICATES IF AN ELSE             52
C                               STATEMENT HAS BEEN ENCOUNTERED AND IF          53
C                               A BLOCK OF STATEMENTS HAS BEEN EXECUTED        54
C                       IFLINE: CHARACTER STRING OF THE STATEMENT              55
C                               FOLLOWING A LOGICAL IF                         56
C                       RSTART: A VARIABLE THATS VALUE INDUCATES               57
C                               WHICH DIRECTION VMAIN IS TO TAKE:              58
C                                   1:  CONTINUE EXECUTION                     59
C                                   2:  RESTART EXECUTION FORM START           60
C                                   3:  EXECUTE DIFFERENT PROGRAM              61
C                                   4:  STOP                                   62
C       ERRORS NUMBERS CALLED                                                  63
C               109                                                            64
C               150                                                            65
C               405                                                            66
C               407                                                            67
C                                                                              68
C       SUBROUTINES CALLED:                                                    69
C               FORWRD                                                         70
C               EVALIF                                                         71
C               ERROR                                                          72
```

```
C                                                                               73
C       SUBROUTINES THAT CALL SIF:                                              74
C              SEARCH                                                           75
C                                                                               76
C*******************************************************************************77
C                                                                               78
        SUBROUTINE SIF(LINE,PATH)                                               79
        INTEGER BLANK,LLOOK,RLOOK,BUG,LEVEL,LINDEX,LPOINT,IFTABL(5,2),          80
     $                  BINDEX,BRANCH(20,6),PATH,IFFLAG,MAXLIN,RSTART           81
        CHARACTER LINE*72,LOGEXP*72,SCLINE*72,EXPRES*72,                        82
     $                             IFLINE*72                                    83
        LOGICAL TRUTH                                                           84
        DIMENSION SCLINE(20)                                                    85
        COMMON/CONTRL/LINDEX,LPOINT,LEVEL,MAXLIN                                86
        COMMON/CONTR2/SCLINE                                                    87
        COMMON/BLOCK2/BRANCH,BINDEX                                             88
        COMMON/IFBLK/IFTABL,IFFLAG                                              89
        COMMON/IFBLKC/IFLINE                                                    90
        COMMON/STATE/RSTART                                                     91
        COMMON/CHECK/BUG                                                        92
        IFFLAG=0                                                                93
        GO TO (10,300,400)PATH                                                  94
 10     LOOK=INDEX(LINE,'THEN ')                                                95
        IF(LOOK.EQ.0)IFFLAG=1                                                   96
C                                                                               97
C               RECORD 'IF' IN BRANCH TABLE                                     98
C                                                                               99
        IF(IFFLAG.EQ.1) THEN                                                   100
              CONTINUE                                                         101
        ELSE                                                                   102
          LEVEL=LEVEL+1                                                        103
          IF(LEVEL.GT.5) THEN                                                  104
                CALL ERROR(LINE,407)                                           105
                GO TO 1000                                                     106
          ELSE                                                                 107
                CONTINUE                                                       108
          ENDIF                                                                109
          IF(LPOINT.GT.BRANCH(BINDEX,2))THEN                                   110
                 BINDEX=BINDEX+1                                               111
                 BRANCH(BINDEX,2)=LPOINT                                       112
                 BRANCH(BINDEX,3)=3                                            113
                 BRANCH(BINDEX,5)=BRANCH(BINDEX-1,5)                           114
                 BRANCH(BINDEX,6)=BRANCH(BINDEX-1,6)+1                         115
          ELSE                                                                 116
                  CONTINUE                                                     117
          ENDIF                                                                118
          IF(LPOINT.EQ.BRANCH(BINDEX,2))THEN                                   119
                 BRANCH(BINDEX,3)=3                                            120
                 BRANCH(BINDEX,5)=BRANCH(BINDEX-1,5)                           121
                 BRANCH(BINDEX,6)=BRANCH(BINDEX-1,6)+1                         122
          ELSE                                                                 123
                CONTINUE                                                       124
          ENDIF                                                                125
        ENDIF                                                                  126
C                                                                              127
C               PARSE LINE FOR LOGICAL EXPRESSION                              128
C                                                                              129
 100    LLOOK=INDEX(LINE,'(')                                                  130
        RLOOK=INDEX(LINE,')')                                                  131
        IF(LLOOK.EQ.0.OR.RLOOK.EQ.0) THEN                                      132
                CALL ERROR(LINE,109)                                           133
                GO TO 1000                                                     134
        ELSE                                                                   135
                CONTINUE                                                       136
        ENDIF                                                                  137
        LOGEXP=LINE(LLOOK+1:RLOOK-1)                                           138
C                                                                              139
C               EVALUATE LOGICAL EXPRESSION                                    140
C                                                                              141
        CALL EVALIF(LINE,LOGEXP,TRUTH)                                         142
        IF(RSTART.NE.1)GO TO 1000                                              143
        IF(TRUTH) THEN                                                         144
```

```
C                                                                       145
C               LOGICAL ARGUMENT IS TRUE                                146
C                                                                       147
              IF(IFFLAG.EQ.0) THEN                                      148
                IFTABL(LEVEL,1)=1                                       149
                GO TO 1000                                              150
              ELSE                                                      151
                IFLINE=LINE(RLOOK+1:72)                                 152
                GO TO 1000                                              153
              ENDIF                                                     154
        ELSE                                                            155
C                                                                       156
C               LOGICAL ARGUMENT IS FALSE                               157
C                                                                       158
              IF(IFFLAG.EQ.1) THEN                                      159
                IFFLAG=0                                                160
                GO TO 1000                                              161
              ELSE                                                      162
                CONTINUE                                                163
              ENDIF                                                     164
              IF(LPOINT.LT.LINDEX)THEN                                  165
                DO 210 I = 1,BINDEX                                     166
                IF(BRANCH(I,3).EQ.4.AND.BRANCH(I,6).EQ.LEVEL) THEN      167
                        IF(LPOINT.GE.BRANCH(I,2))GO TO 210              168
                        LPOINT=BRANCH(I,2)                              169
                        GO TO 200                                       170
                ELSE                                                    171
                        CONTINUE                                        172
                ENDIF                                                   173
 210            CONTINUE                                                174
            ELSE                                                        175
                LPOINT=LINDEX                                           176
                CALL FORWRD(2,0)                                        177
                IF(RSTART.NE.1)GO TO 1000                               178
                LPOINT=LINDEX                                           179
            ENDIF                                                       180
        ENDIF                                                           181
C                                                                       182
C               FOUND AN ELSE                                           183
C                                                                       184
 200    LINE=SCLINE(LPOINT)                                             185
        LOOK=INDEX(LINE,'ELSE')                                         186
        LINE=LINE(LOOK:72)                                              187
        LOOK=INDEX(LINE,'IF ')                                          188
        LOOK1=INDEX(LINE,'IF(')                                         189
        IF(LOOK.NE.0.OR.LOOK1.NE.0)THEN                                 190
                LOOK=INDEX(LINE,'THEN')                                 191
                IF(LOOK.EQ.0) THEN                                      192
                        CALL ERROR(LINE,109)                            193
                        GO TO 1000                                      194
                ELSE                                                    195
                        CONTINUE                                        196
                ENDIF                                                   197
                GO TO 100                                               198
        ELSE                                                            199
              IF(IFTABL(LEVEL,2).NE.0) THEN                             200
                        CALL ERROR(SCLINE(LPOINT),405)                  201
                        GO TO 1000                                      202
                ELSE                                                    203
                        CONTINUE                                        204
                ENDIF                                                   205
              IFTABL(LEVEL,1)=1                                         206
              IFTABL(LEVEL,2)=1                                         207
        ENDIF                                                           208
        GO TO 1000                                                      209
 C                                                                      210
```

```
C   ***************************************************************************   211
C                                                                                 212
C               ENTRANCE TO SUBROUTINE FROM 'ELSE' STATEMENT                      213
C                                                                                 214
 300    LOOK=INDEX(LINE,'ELSE')                                                   215
        IF(LOOK.NE.1) THEN                                                        216
                CALL ERROR(LINE,150)                                              217
                GO TO 1000                                                        218
        ELSE                                                                      219
                CONTINUE                                                          220
        ENDIF                                                                     221
        EXPRES=LINE                                                               222
C                                                                                 223
C               CHECK IF-TABLE TO MAKE SURE EXECUTION HAS OCCURED                 224
C                                                                                 225
        IF(IFTABL(LEVEL,1).NE.1) THEN                                             226
                CALL ERROR(LINE,405)                                              227
                GO TO 1000                                                        228
        ELSE                                                                      229
                CONTINUE                                                          230
        ENDIF                                                                     231
C                                                                                 232
C               RECORD ELSE STATEMENT                                             233
C                                                                                 234
        IF(LPOINT.GT.BRANCH(BINDEX,2))THEN                                        235
                BINDEX=BINDEX+1                                                   236
                BRANCH(BINDEX,2)=LPOINT                                           237
                BRANCH(BINDEX,3)=4                                                238
                BRANCH(BINDEX,5)=BRANCH(BINDEX-1,5)                               239
                BRANCH(BINDEX,6)=BRANCH(BINDEX-1,6)                               240
        ELSE IF(LPOINT.EQ.BRANCH(BINDEX,2))THEN                                   241
                BRANCH(BINDEX,3)=4                                                242
                BRANCH(BINDEX,5)=BRANCH(BINDEX-1,5)                               243
                BRANCH(BINDEX,6)=BRANCH(BINDEX-1,6)                               244
        ELSE                                                                      245
                CONTINUE                                                          246
        ENDIF                                                                     247
C                                                                                 248
C               IT IS AN 'ELSE IF' STATEMENT (LOOKING FOR 'ELSE')                 249
C                                                                                 250
 305    LOOK=INDEX(EXPRES,'IF')                                                   251
        IF(LOOK.NE.0) THEN                                                        252
            IF(LPOINT.LT.LINDEX)THEN                                              253
                DO 310 I=1,BINDEX                                                 254
                    IF(BRANCH(I,3).EQ.4.AND.BRANCH(I,6).EQ.LEVEL)THEN             255
                        IF(LPOINT.GE.BRANCH(I,2))GO TO 310                        256
                        LPOINT=BRANCH(I,2)                                        257
                        EXPRES=SCLINE(LPOINT)                                     258
                        LOOK=INDEX(EXPRES,'ELSE')                                 259
                        IF(LOOK.EQ.0) THEN                                        260
                                CALL ERROR(LINE,150)                              261
                                GO TO 1000                                        262
                        ELSE                                                      263
                                CONTINUE                                          264
                        ENDIF                                                     265
                        GO TO 305                                                 266
                    ELSE                                                          267
                        CONTINUE                                                  268
                    ENDIF                                                         269
 310            CONTINUE                                                         270
            ELSE                                                                  271
                CONTINUE                                                          272
            ENDIF                                                                 273
            CALL FORWRD(2,0)                                                      274
            IF(RSTART.NE.1) GO TO 1000                                            275
            EXPRES=SCLINE(LPOINT)                                                 276
            LOOK=INDEX(EXPRES,'ELSE')                                             277
            IF(LOOK.EQ.0)THEN                                                     278
                CALL ERROR(LINE,150)                                              279
                GO TO 1000                                                        280
            ELSE                                                                  281
                CONTINUE                                                          282
```

```
          ENDIF                                                                 283
          GO TO 305                                                             284
        ELSE                                                                    285
          CONTINUE                                                              286
        ENDIF                                                                   287
 320    IFTABL(LEVEL,2)=1                                                       288
C                                                                               289
C               LOOK FOR 'ENDIF STATEMENT                                       290
C                                                                               291
        IF(LPOINT.LT.LINDEX)THEN                                                292
               DO 330 I=1,BINDEX                                                293
                IF(BRANCH(I,3).EQ.5.AND.BRANCH(I-1,6).EQ.LEVEL)THEN             294
                     LPOINT=BRANCH(I,2)-1                                       295
                     GO TO 1000                                                 296
                  ELSE                                                          297
                     CONTINUE                                                   298
                  ENDIF                                                         299
 330           CONTINUE                                                        300
        ELSE                                                                    301
               CONTINUE                                                         302
        ENDIF                                                                   303
        CALL FORWRD(3,0)                                                        304
        IF(RSTART.NE.1)GO TO 1000                                               305
        LPOINT=LPOINT-1                                                         306
        GO TO 1000                                                              307
C                                                                               308
C   ************************************************************************    309
C                                                                               310
C               ENTRANCE TO SUBROUTINE FROM 'ENDIF' STATEMENT                   311
C                                                                               312
 400    IF(IFTABL(LEVEL,1).NE.1.OR.IFTABL(LEVEL,2).NE.1) THEN                   313
               CALL ERROR(EXPRES,405)                                           314
               GO TO 1000                                                       315
        ELSE                                                                    316
               CONTINUE                                                         317
        ENDIF                                                                   318
        IFTABL(LEVEL,1)=0                                                       319
        IFTABL(LEVEL,2)=0                                                       320
        IF(LPOINT.GT.BRANCH(BINDEX,2))THEN                                      321
             BINDEX=BINDEX+1                                                    322
             BRANCH(BINDEX,2)=LPOINT                                            323
             BRANCH(BINDEX,3)=5                                                 324
             BRANCH(BINDEX,5)=BRANCH(BINDEX-1,5)                                325
             BRANCH(BINDEX,6)=BRANCH(BINDEX-1,6)-1                              326
             LEVEL=LEVEL-1                                                      327
             GO TO 1000                                                         328
        ELSE IF(LPOINT.EQ.BRANCH(BINDEX,2))THEN                                 329
             BRANCH(BINDEX,3)=5                                                 330
             BRANCH(BINDEX,5)=BRANCH(BINDEX-1,5)                                331
             BRANCH(BINDEX,6)=BRANCH(BINDEX-1,6)-1                              332
             LEVEL=LEVEL-1                                                      333
             GO TO 1000                                                         334
        ELSE                                                                    335
             LEVEL=LEVEL-1                                                      336
             GO TO 1000                                                         337
        ENDIF                                                                   338
 1000   RETURN                                                                  339
        END                                                                     340
```

```
C*****************************   SUBROUTINE EVALIF   ********************
C
C       FORMAT  ------ EVALIF(LINE,LOGEXP,TRUTH)
C
C       PURPOSE:        TO EVALUATE THE LOGICAL EXPRESSIONS OF IF
C                       STATEMENTS
C
C       PARAMETERS:
C
C               I/O PARAMETERS:
C
C                       LINE:   THE LINE OF CODE THAT CONTAINS THE
C                               IF STATEMENT
C                       LOGEXP: CHARACTER STRING OF THE LOGICAL
C                               EXPRESSION TO BE EVALUATED
C                               TRUTH:  LOGICAL VARIABLE WITH THE
C                                       VALUE OF EVALUATED LOGICAL
C                                       EXPESSION
C
C               PARSING PARAMETERS:
C
C                       LOOK:   INTEGER VALUE OF LOCATION OF ELEMENT
C                               BEING PARSED
C                       DOUBLE: A VALUE OF 1 INDICATES A 2 PART LOGICAL
C                               EXPRESSION
C                       VALUE1,VALUE2,VALUE3,VALUE4: THE REAL VALUES
C                               OF EACH OPERAND IN LOGICAL EXPRESSION
C                       CHECK:  VARIABLE USED TO SEE IF THE CHARACTER
C                               FOLLOWING A PERIOD IS AN OPERATER OR
C                               A NUMBER
C                       IVAL,RVAL,DVAL,CVAL,LVAL:  PATHWAYS FOR THE
C                               VALUES OF THE OPERAND TO BE PASSED
C                       TYPE:   THE TYPE OF VALUE RETURNED
C                       RELOP:  CHARACTER STRING THAT REPRESENTS THE
C                               RATIONAL OPERATOR USED
C                       LOGOP:  CHARACTER STRING THAT REPRESENTS THE
C                               LOGICAL OPERATOR USED
C
C               TO EVALUATE EXPRESSION:
C
C                       CODE1,CODE2:  INTEGER CODES AS TO WHICH
C                               RELATIONAL OPERATOR IS BEING USED
C                                       1:  LT
C                                       2:  LE
C                                       3:  EQ
C                                       4:  NE
C                                       5:  GT
C                                       6:  GE
C                       PART1,PART2:  LOGICAL VARIABLES THAT HOLD THE
C                               VALUE OF RELATONAL EXPRESSION EVALUATED
C                       LCODE:  INTEGER CODE AS TO WHICH LOGICAL
C                               OPTERATOR IS BEING USED:
C                                       1: AND
C                                       2: OR
C                       RSTART: A VARIABLE THATS VALUE INDICATES
C                               WHICH DIRECTION VMAIN IS TO TAKE:
C                                   1:   CONTINUE EXECUTION
C                                   2:   RESTART EXECUTION FROM START
C                                   3:   EXECUTE DIFFERENT PROGRAM
C                                   4:   STOP
C       ERRORS NUMBERS CALLED
C               109
C
C       SUBROUTINES CALLED:
C               NUMBER
C               VARABL
C               OPERAT
C               LOPER
C               ERROR
C
C       SUBROUTINES THAT CALL EVALIF:
C               SIF
```

```
C                                                                               73
C*******************************************************************************74
C                                                                               75
      SUBROUTINE EVALIF(LINE,LOGEXP,TRUTH)                                      76
      INTEGER LOOK,TYPE,IVAL,CODE1,DOUBLE,CODE2,BUG,RSTART                      77
      REAL RVAL,VALUE1,VALUE2,VALUE3,VALUE4                                     78
      DOUBLE PRECISION DVAL                                                     79
      CHARACTER LINE*72,LOGEXP*72,STRING*72,WORD*10,RELOP*2,LOGOP*3,            80
     $                  CVAL*60                                                 81
      LOGICAL PART1,PART2,TRUTH                                                 82
      COMMON/STATE/RSTART                                                       83
      COMMON/CHECK/BUG                                                          84
C                                                                               85
C              FIND FIRST OPERAND                                               86
C                                                                               87
      STRING=LOGEXP                                                             88
      LOOK=INDEX(STRING,'.')                                                    89
      WORD=STRING(1:LOOK-1)                                                     90
      CALL NUMBER(LINE,WORD,TYPE,IVAL,RVAL,DVAL)                                91
      IF(RSTART.NE.1)GO TO 1000                                                 92
      GO TO (10,20,30) TYPE                                                     93
 10   VALUE1=IVAL                                                               94
      GO TO 50                                                                  95
 20   VALUE1=RVAL                                                               96
      GO TO 50                                                                  97
 30   VALUE1=DVAL                                                               98
C                                                                               99
C              FIND FIRST RELATIONAL OPERATOR                                  100
C                                                                              101
 50   STRING=STRING(LOOK+1:72)                                                  102
      LOOK=INDEX(STRING,'.')                                                    103
      RELOP=STRING(1:LOOK-1)                                                    104
      CALL OPERAT(LINE,RELOP,CODE1)                                             105
      IF(RSTART.NE.1) GO TO 1000                                                106
C                                                                               107
C             FIND SECOND OPERAND AND CHECK FOR TWO PART EXPRESSION             108
C                                                                               109
      STRING=STRING(LOOK+1:72)                                                  110
      LOOK=INDEX(STRING,'.')                                                    111
      IF(LOOK.EQ.0) THEN                                                        112
              DOUBLE=0                                                          113
              LOOK=INDEX(STRING,' ')                                            114
      ELSE                                                                      115
              DOUBLE=1                                                          116
      ENDIF                                                                     117
      WORD=STRING(1:LOOK-1)                                                     118
      CALL NUMBER(LINE,WORD,TYPE,IVAL,RVAL,DVAL)                                119
      IF(RSTART.NE.1)GO TO 1000                                                 120
      GO TO (110,120,130)TYPE                                                   121
 110  VALUE2=IVAL                                                               122
      GO TO 150                                                                 123
 120  VALUE2=RVAL                                                               124
      GO TO 150                                                                 125
 130  VALUE2=DVAL                                                               126
C                                                                               127
C              EVALUATE PART ONE OF THE EXPRESSION                              128
C                                                                               129
 150  PART1=.FALSE.                                                             130
      GO TO (210,220,230,240,250,260) CODE1                                     131
 210  IF(VALUE1 .LT. VALUE2) PART1=.TRUE.                                       132
      GO TO 300                                                                 133
 220  IF(VALUE1 .LE. VALUE2) PART1=.TRUE.                                       134
      GO TO 300                                                                 135
 230  IF(VALUE1 .EQ. VALUE2) PART1=.TRUE.                                       136
      GO TO 300                                                                 137
 240  IF(VALUE1 .NE. VALUE2) PART1=.TRUE.                                       138
      GO TO 300                                                                 139
 250  IF(VALUE1 .GT. VALUE2) PART1=.TRUE.                                       140
      GO TO 300                                                                 141
 260  IF(VALUE1 .GE. VALUE2) PART1=.TRUE.                                       142
```

```
 300    IF(DOUBLE.EQ.0) THEN                                              143
               TRUTH=PART1                                                144
               GO TO 1000                                                 145
        ELSE                                                              146
               CONTINUE                                                   147
        ENDIF                                                             148
C                                                                         149
C              FIND LOGICAL OPERATER OF TWO PART EXPRESSION               150
C                                                                         151
        STRING=STRING(LOOK+1:72)                                          152
        LOOK=INDEX(STRING,'.')                                            153
        LOGOP=STRING(1:LOOK-1)                                            154
C                                                                         155
C              FIND THIRD OPERAND                                         156
C                                                                         157
        STRING=STRING(LOOK+1:72)                                          158
        LOOK=INDEX(STRING,'.')                                            159
        WORD=STRING(1:LOOK-1)                                             160
        CALL NUMBER(LINE,WORD,TYPE,IVAL,RVAL,DVAL)                        161
        IF(RSTART.NE.1)GO TO 1000                                         162
        GO TO (310,320,330) TYPE                                          163
 310    VALUE3=IVAL                                                       164
        GO TO 350                                                         165
 320    VALUE3=RVAL                                                       166
        GO TO 350                                                         167
 330    VALUE3=DVAL                                                       168
C                                                                         169
C              FIND SECOND RATIONAL OPERATOR                              170
C                                                                         171
 350    STRING=STRING(LOOK+1:72)                                          172
        LOOK=INDEX(STRING,'.')                                            173
        RELOP=STRING(1:LOOK-1)                                            174
        CALL OPERAT(LINE,RELOP,CODE2)                                     175
        IF(RSTART.NE.1)GO TO 1000                                         176
C                                                                         177
C              FIND FORTH OPERAND                                         178
C                                                                         179
        STRING=STRING(LOOK+1:72)                                          180
        LOOK=INDEX(STRING,'.')                                            181
        IF(LOOK.NE.0) THEN                                                182
               CALL ERROR(LINE,109)                                       183
               GO TO 1000                                                 184
        ELSE                                                              185
               CONTINUE                                                   186
        ENDIF                                                             187
        LOOK=INDEX(STRING,' ')                                            188
        IF(LOOK.EQ.1) THEN                                                189
               CALL ERROR(LINE,109)                                       190
               GO TO 1000                                                 191
        ELSE                                                              192
               CONTINUE                                                   193
        ENDIF                                                             194
        WORD=STRING(1:LOOK-1)                                             195
        CALL NUMBER(LINE,WORD,TYPE,IVAL,RVAL,DVAL)                        196
        IF (RSTART.NE.1)GO TO 1000                                        197
        GO TO (410,420,430) TYPE                                          198
 410    VALUE4=IVAL                                                       199
        GO TO 450                                                         200
 420    VALUE4=RVAL                                                       201
        GO TO 450                                                         202
 430    VALUE4=DVAL                                                       203
C                                                                         204
C              EVALUATE PART TWO OF THE EXPRESSION                        205
C                                                                         206
 450    PART2=.FALSE.                                                     207
        GO TO (510,520,530,540,550,560) CODE2                             208
 510    IF(VALUE3 .LT. VALUE4) PART2=.TRUE.                               209
        GO TO 600                                                         210
 520    IF(VALUE3 .LE. VALUE4) PART2=.TRUE.                               211
        GO TO 600                                                         212
 530    IF(VALUE3 .EQ. VALUE4) PART2=.TRUE.                               213
        GO TO 600                                                         214
```

```
 540    IF(VALUE3 .NE. VALUE4) PART2=.TRUE.                                    215
        GO TO 600                                                              216
 550    IF(VALUE3 .GT. VALUE4) PART2=.TRUE.                                    217
        GO TO 600                                                              218
 560    IF(VALUE3 .GE. VALUE4) PART2=.TRUE.                                    219
C                                                                              220
C              PREFORM LOGICAL EVALUATION ON PART1 AND PART2                   221
C                                                                              222
 600    CALL LOPER(LINE,LOGOP,LCODE)                                           223
        IF(RSTART.NE.1) GO TO 1000                                             224
        TRUTH=.FALSE.                                                          225
        GO TO (610,620) LCODE                                                  226
 610    IF(PART1 .AND. PART2) TRUTH=.TRUE.                                     227
        GO TO 1000                                                             228
 620    IF(PART1 .OR. PART2) TRUTH=.TRUE.                                      229
 1000   RETURN                                                                 230
        END                                                                    231
C                                                                              232
C*************************** SUBROUTINE LOPER  ************************       233
C                                                                              234
C       FORMAT  -----  LOPER(LINE,LOGOP,CODE) -----                            235
C                                                                              236
C       PURPOSE:        TO CODE WHICH LOGICAL OPERATOR IS BEING USED           237
C                                                                              238
C       PARAMETERS:                                                            239
C                                                                              240
C               I/O PARAMETERS:                                                241
C                                                                              242
C                       LINE:   CHARACTER STRING OF IF STATEMENT               243
C                       LOGOP:  CHARACTER STRING OF LOGICAL OPERATORS          244
C                       CODE:   LOGICAL OPERATOR CODE                          245
C                               1:  AND                                        246
C                               2:  OR                                         247
C                                                                              248
C               OTHER PARAMETERS:                                              249
C                                                                              250
C                       LOGTYP: CHARACTER ARRAY OF ALLOWED LOGICAL             251
C                               OPERATORS                                      252
C                                                                              253
C       ERROR NUMBERS CALLED:                                                  254
C               110                                                            255
C                                                                              256
C       SUBROUTINES CALLED:                                                    257
C               NONE                                                           258
C                                                                              259
C       SUBROUTINES THAT CALL LOPER:                                           260
C               EVALIF                                                         261
C                                                                              262
C*******************************************************************************263
C                                                                              264
        SUBROUTINE LOPER(LINE,LOGOP,CODE)                                      265
        INTEGER CODE                                                           266
        CHARACTER LOGTYP*3,LINE*72,LOGOP*3                                     267
        DIMENSION LOGTYP(2)                                                    268
        LOGTYP(1)='AND'                                                        269
        LOGTYP(2)='OR'                                                         270
        DO 100 I=1,2                                                           271
              IF(LOGOP.EQ.LOGTYP(I)) GO TO  200                                272
  100   CONTINUE                                                               273
        CALL ERROR(LINE,110)                                                   274
        GO TO 1000                                                             275
  200   CODE=I                                                                 276
  1000  RETURN                                                                 277
        END                                                                    278
```

```
C                                                                       279
C*************************** SUBROUTINE OPERAT ************************** 280
C                                                                       281
C       FORMAT  -----  OPERAT(LINE,RELOP,CODE) -----                    282
C                                                                       283
C       PURPOSE:        TO CODE WHICH RELATIONAL OPERATOR IS BEING USED  284
C                                                                       285
C       PARAMETERS:                                                     286
C                                                                       287
C               I/O PARAMETERS:                                         288
C                                                                       289
C                       LINE:   CHARACTER STRING OF IF STATEMENT        290
C                       LOGOP:  CHARACTER STRING OF RELATIONAL          291
C                               OPERATOR                                292
C                       CODE:   RELATIONAL OPERATOR CODE                293
C                               1:  LT                                  294
C                               2:  LE                                  295
C                               3:  EQ                                  296
C                               4:  NE                                  297
C                               5:  GT                                  298
C                               6:  GE                                  299
C                                                                       300
C               OTHER PARAMETERS:                                       301
C                                                                       302
C                       OPTYPE: CHARACTER ARRAY OF ALLOWED LOGICAL      303
C                               OPERATORS                               304
C                                                                       305
C       ERROR NUMBERS CALLED:                                           306
C               110                                                     307
C                                                                       308
C       SUBROUTINES CALLED:                                             309
C               NONE                                                    310
C                                                                       311
C       SUBROUTINES THAT CALL LOPER:                                    312
C               EVALIF                                                  313
C                                                                       314
C*********************************************************************** 315
C                                                                       316
        SUBROUTINE OPERAT(LINE,RELOP,CODE)                              317
        INTEGER CODE                                                    318
        CHARACTER OPTYPE*2,LINE*72,RELOP*2                              319
        DIMENSION OPTYPE(6)                                             320
        OPTYPE(1)='LT'                                                  321
        OPTYPE(2)='LE'                                                  322
        OPTYPE(3)='EQ'                                                  323
        OPTYPE(4)='NE'                                                  324
        OPTYPE(5)='GT'                                                  325
        OPTYPE(6)='GE'                                                  326
        DO 100 I=1,6                                                    327
            IF(RELOP.EQ.OPTYPE(I)) GO TO 200                            328
  100   CONTINUE                                                        329
        CALL ERROR(LINE,110)                                            330
        GO TO 1000                                                      331
  200   CODE=I                                                          332
  1000  RETURN                                                          333
        END                                                             334
```

```
C*****************************   SUBROUTINE SGOTO   **********************
C
C       FORMAT  ------ SGOTO(LINE) -------
C
C       PURPOSE:        TO PARSE A GOTO STATEMENT AND FIND THE
C                       STATEMENT NUMBER AND LINE NUMBER OF THE NEXT
C                       STATEMENT TO BE INTERPRETED
C
C       PARAMETERS:
C
C               I/O PARAMETERS:
C
C                       LINE:   LINE OF CODE BEING INTERPRETED
C
C               OTHER PARAMETER:
C
C                       I,R,D,C,L:  PATHWAYS FOR VALUES TO BE SENT
C                               TO AND FROM SUBROUTINE NUMBER, OF TYPES
C                               INTEGER, REAL, DOUBLE PRECISION,
C                               CHARACTER AND LOGICAL
C                       TYPE:   THE DATA TYPE RETURNED FOR SUBROUTINE
C                               NUMBER
C                       COMMA:  INTEGER LOCATION OF A COMMA
C                       NUM:    STATEMENT NUMBER OF LINE BEING ENTERED
C                       BEXIT:  POINT IN BRANCH TABLE THAT IS EXITED
C                               FROM
C                       BENTRY: POINT IN BRANCH TABLE THAT IS ENTERED
C                               INTO
C                       POINT:  A POINTER THAT SHOWS WHERE PARSING IS
C                               BEING DONE
C                       DLEVEL: DO LEVEL
C                       FLEVEL: IF LEVEL
C                       LOOK,LOOK1: INTEGER LOCATIONS OF CHARACTER
C                               BEING PARSED FOR
C                       UPFLAG: FLAG INDICATING THE LINE TO GOTO IS
C                               ABOVE THE PRESENT LINE
C                       EXPRES: LINE OF CODE BEING INTERPRETED THAT
C                               BECOMES SHORTENED AS IT IS PARSED
C
C               COMMON PARAMETERS:
C
C                       RSTART: A VARIABLE THATS VALUE INDICATES
C                               WHICH DIRECTION VMAIN IS TO TAKE:
C                                       1:  CONTINUE EXECUTION
C                                       2:  RESTART EXECUTION FROM START
C                                       3:  EXECUTE DIFFERENT PROGRAM
C                                       4:  STOP
C                       BRANCH: AN ARRAY TABLE THAT HOLDS THE MAP OF
C                               THE PROGRAM. USED FOR BRANCHING AND
C                               TO CHECK PROGRAM STRUCTURE
C                       BINDEX: THE ROW OF BRANCH THE NEXT ENTRY IS TO
C                               BE MADE ON
C                       LINDEX: LINE NUMBER OF LINE READ FURTHEST IN
C                               PROGRAM
C                       LPOINT: LINE NUMBER OF THE LINE BEING INTERPRETE
C                               INTERPERED
C                       LEVEL:  THE CURRENT IF LEVEL
C                       MAXLIN: MAXIMUM NUMBER OF LINES ALLOWED IN THE
C                               USER'S PROGRAM
C
C
C       ERRORS NUMBERS CALLED
C               108
C               303
C               402
C               403
C
C       SUBROUTINES CALLED:
C               NUMBER
C               VARABL
C               FORWRD
C
```

```
C       SUBROUTINES THAT CALL SGOTO:                                            73
C               SEARCH                                                          74
C               SGOTO                                                           75
C                                                                               76
C*******************************************************************************77
C                                                                               78
        SUBROUTINE SGOTO(LINE)                                                  79
        INTEGER BRANCH,BINDEX,LPOINT,BEXIT,BENTRY,POINT,TYPE,RSTART,            80
     $   MAXLIN,NUM,DLEVEL,FLEVEL,BUG,LOOK,LOOK1,COMMA,LEVEL,UPFLAG             81
        REAL R                                                                  82
        CHARACTER LINE*72,WORD*6,EXPRES*72,C*60,PLINE*72,HELP*72                83
        DOUBLE PRECISION D                                                      84
        LOGICAL L                                                               85
        COMMON/BLOCK2/BRANCH(20,6),BINDEX                                       86
        COMMON/CONTRL/LINDEX,LPOINT,LEVEL,MAXLIN                                87
        COMMON/STATE/RSTART                                                     88
        COMMON/CHECK/BUG                                                        89
        PLINE=LINE                                                              90
        TYPE=0                                                                  91
        UPFLAG=0                                                                92
C                                                                               93
C               FIND 'GOTO' EXIT POINT                                          94
C                                                                               95
        IF(LPOINT.GT.BRANCH(BINDEX,2))THEN                                      96
               BEXIT=BINDEX                                                     97
               GO TO 100                                                        98
        ELSE                                                                    99
               CONTINUE                                                        100
        ENDIF                                                                  101
        DO 50 I= BINDEX,0,-1                                                   102
           IF(LPOINT.LT.BRANCH(I+1,2).AND.LPOINT.GE.BRANCH(I,2))THEN           103
               BEXIT=I                                                         104
               GO TO 100                                                       105
           ELSE                                                                106
               CONTINUE                                                        107
           ENDIF                                                               108
 50     CONTINUE                                                               109
C                                                                              110
C               PARSE STATEMENT FOR COMPUTED 'GO TO'                           111
C                                                                              112
 100    LOOK=INDEX(LINE,')')                                                   113
        IF(LOOK.EQ.0) GO TO 120                                                114
        EXPRES=LINE(LOOK+1:72)                                                 115
        COMMA=INDEX(EXPRES,',')                                                116
        IF(COMMA.NE.0)EXPRES=EXPRES(COMMA+1:72)                                117
        WORD=EXPRES(1:6)                                                       118
        CALL VARABL(2,WORD,TYPE,I,R,D,C,L)                                     119
        IF(RSTART.NE.1)GO TO 1000                                              120
        IF(TYPE.NE.1) THEN                                                     121
               CALL ERROR(LINE,303)                                            122
               GO TO 1000                                                      123
        ELSE                                                                   124
               CONTINUE                                                        125
        ENDIF                                                                  126
        EXPRES=LINE(1:72)                                                      127
        LOOK=INDEX(EXPRES,'(')                                                 128
        EXPRES=EXPRES(LOOK+1:72)                                               129
        DO 110 J=1,I-1                                                         130
               LOOK=INDEX(EXPRES,',')                                          131
               IF(LOOK.EQ.0) THEN                                              132
                      CALL ERROR(LINE,108)                                     133
                      GO TO 1000                                               134
               ELSE                                                            135
                      CONTINUE                                                 136
               ENDIF                                                           137
               EXPRES=EXPRES(LOOK+1:72)                                        138
 110    CONTINUE                                                               139
        LOOK=INDEX(EXPRES,',')                                                 140
        LOOK1=INDEX(EXPRES,')')                                                141
        IF(LOOK.NE.0.AND.LOOK.LT.LOOK1) THEN                                   142
               WORD=EXPRES(1:LOOK-1)                                           143
        ELSE                                                                   144
```

```
                WORD=EXPRES(1:LOOK1-1)                                  145
        ENDIF                                                           146
        CALL NUMBER(PLINE,WORD,TYPE,I,R,D)                              147
        IF(RSTART.NE.1)GO TO 1000                                       148
        IF(TYPE.NE.1) THEN                                              149
                CALL ERROR(LINE,303)                                    150
                GO TO 1000                                              151
        ELSE                                                            152
                CONTINUE                                                153
        ENDIF                                                           154
        NUM=I                                                           155
        GO TO 130                                                       156
C                                                                       157
C               PARSE STATEMENT FOR UNCONDITIONAL 'GO TO'               158
C                                                                       159
 120    POINT=INDEX(LINE,'TO')                                          160
        WORD=LINE(POINT+2:POINT+8)                                      161
        HELP=LINE                                                       162
        CALL NUMBER(PLINE,WORD,TYPE,I,R,D)                              163
        IF(RSTART.NE.1)GO TO 1000                                       164
        IF(TYPE.NE.1) THEN                                              165
                CALL ERROR(LINE,303)                                    166
                GO TO 1000                                              167
        ELSE                                                            168
                CONTINUE                                                169
        ENDIF                                                           170
        NUM=I                                                           171
C                                                                       172
C               FIND ENTRY INTO BRANCH ARRAY                            173
C                                                                       174
 130    DO 150 J=BINDEX,0,-1                                            175
                IF(BRANCH(J,1).EQ.NUM) THEN                             176
                        BENTRY=J                                        177
                        LPOINT=BRANCH(BENTRY,2)                         178
                        UPFLAG=1                                        179
                        GO TO 200                                       180
                ELSE                                                    181
                        CONTINUE                                        182
                ENDIF                                                   183
 150    CONTINUE                                                        184
        CALL FORWRD(1,NUM,0)                                            185
        IF(RSTART.NE.1)GO TO 1000                                       186
        BENTRY=BINDEX                                                   187
        LPOINT=BRANCH(BENTRY,2)                                         188
C                                                                       189
C               CHECK IF 'GO TO' ENTRY IS LEGAL                         190
C                                                                       191
 200    DLEVEL=BRANCH(BENTRY-1,5)                                       192
        FLEVEL=BRANCH(BENTRY-1,6)                                       193
        IF(BRANCH(BEXIT,5).LT.DLEVEL) GO TO 410                         194
        IF(BRANCH(BEXIT,6).LT.FLEVEL) GO TO 420                         195
        IF(UPFLAG.EQ.1) THEN                                            196
            DO 250 I=BENTRY,BEXIT                                       197
                    IF(BRANCH(I,5).LT.DLEVEL) GO TO 410                 198
                    IF(BRANCH(I,6).LT.FLEVEL) GO TO 420                 199
 250        CONTINUE                                                    200
        ELSE                                                            201
            DO 350 I=BENTRY,BEXIT,-1                                    202
                    IF(BRANCH(I,5).LT.DLEVEL) GO TO 410                 203
                    IF(BRANCH(I,6).LT.FLEVEL) GO TO 420                 204
 350        CONTINUE                                                    205
        ENDIF                                                           206
        LPOINT=BRANCH(BENTRY,2)-1                                       207
 1000   RETURN                                                          208
 410    CALL ERROR(HELP,402)                                            209
        RETURN                                                          210
 420    CALL ERROR(HELP,403)                                            211
        RETURN                                                          212
        END                                                             213
```

```
C*****************************   SUBROUTINE SPAUSE  **********************
C
C       CALLING FORMAT ------  SPAUSE ------
C
C       PURPOSE:        TO ALLOW THE USER TO VIEW AND CHANGE
C                       VARIABLES DURING THE INTERPRETATION OF
C                       THEIR PROGRAM
C
C       I/O PARAMETERS:
C
C               INPUT:  NONE
C
C               OUTPUT: THE VIEWING AND CHANGING OF VARIABLE VALUES
C
C       OTHER PARAMETERS:
C
C               TO CREAT MENU:
C
C                       PNAME:  CHARACTER STRING OF MENU NAME
C                       PLINE:  CHARCTER ARRAY OF THE CHOICES NEMU HAS
C                       CHOICE: THE NUMBER OF THE CHOICE THE USER HAS
C                               SELECTED FROM THE MENU
C                               VALUES OF THE USER DEFINED ARRAY
C
C               COMMON PARAMETERS:
C
C                       RSTART: A VARIABLE THATS VALUE INDUCATES
C                               WHICH DIRECTION VMAIN IS TO TAKE:
C                                   1:  CONTINUE EXECUTION
C                                   2:  RESTART EXECUTION FROM START
C                                   3:  EXECUTE DIFFERENT PROGRAM
C                                   4:  STOP
C
C                               CHARACTER VARIABLE VALUES
C
C       ERROR NUMBERS CALLED:
C                       NONE
C
C       SUBROUTINES CALLED:
C                       MENU
C                       NUMBER
C                       VAROUT
C                       VARCHG
C                       PATHWY
C
C       SUBROUTINES THAT CALL SPAUSE:
C                       SEARCH
*************************************************************************
C
        SUBROUTINE SPAUSE
        INTEGER BUG,CHOICE,RSTART
        CHARACTER PLINE*72,PNAME*20
        DIMENSION PLINE(8)
        COMMON/STATE/RSTART
        COMMON/CHECK/BUG
        PNAME='PAUSE
        PLINE(1)='DISPLAY VARIABLE VALUES'
```

```
      PLINE(2)='CHANGE A VARIABLE VALUE'                                58
      PLINE(3)='CONTINUE EXECUTION FROM THIS POINT'                     59
      PLINE(4)='START EXECUTION FROM BEGINNING OF PROGRAM'              60
      PLINE(5)='QUIT'                                                   61
C                                                                       62
C             CHOSE DIRECTION FROM PAUSE STATEMENT                      63
C                                                                       64
      WRITE(IW,*)'YOU HAVE ENCOUNTERED A PAUSE STATEMENT'               65
 10   CALL MENU(PNAME,PLINE,5,CHOICE)                                   66
      GO TO (100,200,300,400,500)CHOICE                                 67
C                                                                       68
C             DISPLAY VARIABLES                                         69
C                                                                       70
 100  CALL VAROUT                                                       71
      IF(RSTART.NE.1)GO TO 1000                                         72
      GO TO 10                                                          73
C                                                                       74
C             CHANGE VARIABLE VALUE                                     75
C                                                                       76
 200  CALL VARCHG                                                       77
      IF(RSTART.NE.1) GO TO 1000                                        78
      GOTO10                                                            79
C                                                                       80
C             CONTINUE EXECUTION FROM PRESENT LOCATION                  81
C                                                                       82
 300  RSTART=1                                                          83
      REWIND 3                                                          84
      RETURN                                                            85
C                                                                       86
C             RESTART EXECUTION FROM BEGINNING OF PROGRAM               87
C                                                                       88
 400  REWIND 100                                                        89
      RSTART=2                                                          90
      RETURN                                                            91
C                                                                       92
C             QUIT (RETURN TO ORGINAL OPTIONS)                          93
C                                                                       94
 500  REWIND 100                                                        95
      CALL PATHWY                                                       96
 1000 RETURN                                                            97
      END                                                               98
```

```
C*****************************   SUBROUTINE VAROUT   ********************
C
C       CALLING FORMAT ------  VAROUT  -----
C
C       PURPOSE:        TO OUTPUT ONE OR ALL USER DEFINED VARIABLES
C                       AND THEIR VALUES.
C
C       I/O PARAMETERS:
C
C               INPUT:  ALL INFORMATION NEEDED IS PASTED IN COMMON
C                       BLOCKS
C
C               OUTPUT: A LISTING OF ONE OR ALL USER DEFINED VARIABLES
C                       AND THEIR VALUES.
C
C       OTHER PARAMETERS:
C
C               TO CREAT MENU:
C
C                       VNAME:  CHARACTER STRING OF MENU NAME
C                       VLINE:  CHARCTER ARRAY OF THE CHOICES MENU HAS
C                       CHOICE: THE NUMBER OF THE CHOICE THE USER HAS
C                               SELECTED FROM THE MENU
C
C               FOR PARSING:
C
C                       VAR:    CHARACTER STRING OF USER DEFINED
C                               VARIABLE,NAME, INCLUDING SUBSCRITPT
C                               VALUES
C                       LOCAT:  A CHARACTER STRING OF THE SUBSCRIPT
C                               VALUES OF THE USER DEFINED ARRAY
C                       VARNAM: CHARACTER STRING OF THE USER DEFINED
C                               ARRAY NAME WITH NO SUBSCRIPTS
C                       VTYPE:  INTEGER VALUE THAT DISCRIBES VARIABLE
C                               TYPE
C                       ATYPE:  INTEGER VALUE THAT DISCRIBES ARRAY TYPE
C                       INDX:   INTEGER VALUE THAT INDEXES WHICH USER
C                               DEFINED VARIABLE IS SOUGHT
C                       ANUM:   INTEGER VALUE THAT INDEXES WHICH USER
C                               DEFINED ARRAY IS SOUGHT
C                       COMMA:  LOCATION OF COMMA IN ARRAY SUBSCRIPT
C                       DIMEN:  STORES VALUE OF THE DIMENSION OF THE
C                               USER DEFINED ARRAY
C                       MAXROW: MAXIMUM NUMBER OF ROWS IN USER DEFINED
C                               ARRAY
C                       MAXCOL: MAXIMUM NUMBER OF COLUMES IN USER
C                               DEFINED ARRAY
C                       COLVAL: THE COLUME SUBSCRIPT OF USER ARRAY
C                       ROWVAL: THE ROW SUBSCRIPT OF USER ARRAY
C                       I,R,D:  PATHWAYS TO RECIEVE INTEGER,REAL AND
C                               DOUBLE PRECISION VALUES FROM SUBROUTINE
C                               NUMBER
C
C               TO DETECT ERRORS:
C
C                       LCHECK,RCHECK: VARIABLES USED TO CHECK RIGHT
C                               AND LEFT PARENTHESIS
C                       TYPE:   RETURNS TYPE OF SUBSCRIPT(MUST BE
C                               INTEGER)
C
C               COMMON PARAMETERS:
C
C                       VARTBL: A 2-D CHARACTER ARRAY OF ALL USER
C                               DEFINED VARIABLE NAMES
C                       CVAL:   A CHARACTER ARRAY OF USER DEFINED
C                               CHARACTER VARIABLE VALUES
C                       VINDEX: AN INTEGER ARRAY OF THE NUMBER OF
C                               VARIABLES DECLARED IN EACH TYPE
C                       IVAL,RVAL,DVAL,LVAL: ARRAYS OF USER DEFINED
C                               VARIABLE VALUES OF TYPE INTEGER,REAL
C                               DOUBLE PRECISION AND LOGICAL
```

```
C                     ARAYTB: A 2-D CHARACTER ARRAY OF ALL USER
C                             DEFINED ARRAY NAMES
C                     CARAY:  A CHARACTER ARRAY OF USER DEFINED
C                             CHARACTER ARRAY VALUES
C                     AINDEX: AN INTEGER ARRAY OF THE NUMBER OF
C                             ARRAYS DECLARED IN EACH TYPE
C                     AINFO:  AN INTEGER ARRAY THAT STORES THE USER
C                             DEFINED ARRAY DIMENSIONS
C                     IARAY,RARAY,DARAY,LARAY: ARRAYS OF USER DEFINED
C                             ARRAY VALUES OF TYPE INTEGER,REAL,
C                             DOUBLE PRECISON AND LOGICAL
C       ERROR NUMBERS CALLED:
C                     NONE
C
C       SUBROUTINES CALLED:
C                     MENU
C
C       SUBROUTINES THAT CALL VAROUT:
C                      ERROR
C                      SPAUSE
C
C*************************************************************************
C
        SUBROUTINE VAROUT
        INTEGER CHOICE,IVAL,VTYPE,INDX,VINDEX,DIMEN,BUG,TYPE,
     $          AINFO,MAXROW,MAXCOL,ROWVAL,COLVAL,ATYPE,
     $          ANUM,COMMA,LCHECK,RCHECK,AINDEX,IARAY
        REAL RVAL,RARAY
        DOUBLE PRECISION DVAL,DARAY,D
        CHARACTER VNAME*8,VLINE*72,VARTBL*6,CVAL*60,VAR*15,ARAYTB*6,
     $          WORD*10,VARNAM*6,LOCAT*10,CARAY*60
        LOGICAL LVAL(20),LARAY(100,5)
        DIMENSION VLINE(8)
        COMMON/ERVAR1/VARTBL(20,5),CVAL(20)
        COMMON/EINTEG/VINDEX(5),IVAL(20),AINDEX(5),AINFO(10,5,3),
     $          IARAY(100,100,10)
        COMMON/EREAL/RVAL(20),RARAY(100,100,10)
        COMMON/EDOUBL/DVAL(20),DARAY(100,100,10)
        COMMON/ERVAR3/CARAY(100,5)
        COMMON/ERVAR5/LVAL,LARAY
        COMMON/ERVAR6/ARAYTB(10,5)
        COMMON/CHECK/BUG
        VNAME='VARIABLE                '
        VLINE(1)='DISPLAY ALL VARIABLE VALUES'
        VLINE(2)='DISPLAY ONE VARIABLE VALUE'
        VLINE(3)='RETURN'
C
C               DISPLAY VARAIBLE MENU
C
        REWIND 3
   10   CALL MENU(VNAME,VLINE,3,CHOICE)
        GO TO (100,200,300)CHOICE
C
C               DISPLAY ALL VARIABLES
C
  100   REWIND 3
        DO 110 I=1,VINDEX(1)
                IF(VARTBL(I,1).EQ.'      ')GO TO 110
                WRITE(IW,*)VARTBL(I,1),'=',IVAL(I)
  110   CONTINUE
        DO 112 I=1,VINDEX(2)
                IF(VARTBL(I,2).EQ.'      ')GO TO 112
                WRITE(IW,*)VARTBL(I,2),'=',RVAL(I)
  112   CONTINUE
        DO 114 I=1,VINDEX(3)
                IF(VARTBL(I,3).EQ.'      ')GO TO 114
                WRITE(IW,*)VARTBL(I,3),'=',DVAL(I)
  114   CONTINUE
        DO 116 I=1,VINDEX(4)
                IF(VARTBL(I,4).EQ.'      ')GO TO 116
                WRITE(IW,*)VARTBL(I,4),'=',CVAL(I)
  116   CONTINUE
```

```
      DO 118 I=1,VINDEX(5)                                                  144
             IF(VARTBL(I,5).EQ.'       ')GO TO 118                          145
             WRITE(IW,*)VARTBL(I,5),'=',LVAL(I)                             146
 118  CONTINUE                                                              147
C                                                                           148
C             DISPLAY ALL ARRAY VARIABLES                                   149
C                                                                           150
      DO 126 K=1,AINDEX(1)                                                  151
        DO 124 I=1,AINFO(K,1,2)                                             152
          IF(AINFO(K,1,1).EQ.2)THEN                                         153
             DO 122 J=1,AINFO(K,1,3)                                        154
             WRITE(IW,410)ARAYTB(K,1),'(',I,',',J,')=',IARAY(I,J,K)         155
 122         CONTINUE                                                       156
          ELSE                                                              157
             WRITE(IW,411)ARAYTB(K,1),'(',I,')=',IARAY(I,1,K)               158
          ENDIF                                                             159
 124    CONTINUE                                                           160
 126  CONTINUE                                                             161
      DO 136 K=1,AINDEX(2)                                                  162
        DO 134 I=1,AINFO(K,2,2)                                             163
          IF(AINFO(K,2,1).EQ.2)THEN                                         164
             DO 132 J=1,AINFO(K,2,3)                                        165
             WRITE(IW,420)ARAYTB(K,2),'(',I,',',J,')=',RARAY(I,J,K)         166
 132         CONTINUE                                                       167
          ELSE                                                              168
             WRITE(IW,421)ARAYTB(K,2),'(',I,')=',RARAY(I,1,K)               169
          ENDIF                                                             170
 134    CONTINUE                                                           171
 136  CONTINUE                                                             172
      DO 146 K=1,AINDEX(3)                                                  173
        DO 144 I=1,AINFO(K,3,2)                                             174
          IF(AINFO(K,3,1).EQ.2)THEN                                         175
             DO 142 J=1,AINFO(K,3,3)                                        176
             WRITE(IW,420)ARAYTB(K,3),'(',I,',',J,')=',DARAY(I,J,K)         177
 142         CONTINUE                                                       178
          ELSE                                                              179
             WRITE(IW,421)ARAYTB(K,3),'(',I,')=',DARAY(I,1,K)               180
          ENDIF                                                             181
 144    CONTINUE                                                           182
 146  CONTINUE                                                             183
      DO 156 K=1,AINDEX(4)                                                  184
        IF(AINFO(K,4,1).EQ.2) THEN                                          185
             CALL ERROR(VAR,508)                                            186
        ELSE                                                                187
             CONTINUE                                                       188
        ENDIF                                                               189
        DO 154 I=1,AINFO(K,4,2)                                             190
             WRITE(IW,431)ARAYTB(K,4),'(',I,')=',CARAY(I,K)                 191
 154    CONTINUE                                                           192
 156  CONTINUE                                                             193
      DO 166 K=1,AINDEX(5)                                                  194
             IF(AINFO(K,5,1).EQ.2)THEN                                      195
                    CALL ERROR(VAR,508)                                     196
                    RETURN                                                  197
             ELSE                                                           198
                    CONTINUE                                                199
             ENDIF                                                          200
             DO 164 I=1,AINFO(K,5,2)                                        201
                 WRITE(IW,441)ARAYTB(K,5),'(',I,')=',LARAY(I,K)             202
 164         CONTINUE                                                       203
 166  CONTINUE                                                             204
      GO TO 10                                                              205
C                                                                           206
C             DISPLAY ONLY ONE VARIABLE                                     207
C                                                                           208
 200  REWIND 3                                                             209
      WRITE(IW,*)'ENTER THE NAME OF VARIABLE TO BE DISPLAYED'               210
      READ(IR,400,ERR=290)VAR                                               211
      LCHECK=INDEX(VAR,'(')                                                 212
      IF(LCHECK.EQ.0) GO TO 250                                             213
      VARNAM=VAR(1:LCHECK-1)                                                214
```

```
      RCHECK=INDEX(VAR,')')                                              215
      IF(RCHECK.LT.LCHECK)GO TO 274                                      216
      LOCAT=VAR(LCHECK+1:RCHECK-1)                                       217
      DO 220 I=1,5                                                       218
             DO 210 J=1,AINDEX(I)                                        219
             IF(ARAYTB(J,I).EQ.VARNAM) THEN                              220
                    ATYPE=I                                              221
                    ANUM=J                                               222
                    GO TO 230                                            223
             ELSE                                                        224
                    CONTINUE                                             225
             ENDIF                                                       226
 210         CONTINUE                                                    227
 220   CONTINUE                                                          228
       GO TO 250                                                         229
 230   DIMEN=AINFO(ANUM,ATYPE,1)                                         230
       MAXROW=AINFO(ANUM,ATYPE,2)                                        231
       MAXCOL=AINFO(ANUM,ATYPE,3)                                        232
C                                                                        233
C             PARSE EXPRESSION FOR ARRAY ELEMENT SUBSCRIPTS              234
C                                                                        235
       COLVAL=1                                                          236
       COMMA=INDEX(LOCAT,',')                                            237
       IF(COMMA.EQ.0)THEN                                                238
             IF(DIMEN.NE.1) GO TO 274                                    239
             CALL NUMBER(VAR,LOCAT,TYPE,I,R,D)                           240
             IF(TYPE.NE.1) GO TO 272                                     241
             ROWVAL=I                                                    242
             IF(ROWVAL.GT.MAXROW) GO TO 273                              243
       ELSE                                                              244
             IF(DIMEN.NE.2) GO TO 274                                    245
             WORD=LOCAT(1:COMMA-1)                                       246
             CALL NUMBER(VAR,WORD,TYPE,I,R,D)                            247
             IF(TYPE.NE.1) GO TO 272                                     248
             ROWVAL=I                                                    249
             IF(ROWVAL.GT.MAXROW) GO TO 273                              250
             WORD=LOCAT(COMMA+1:RCHECK-1)                                251
             CALL NUMBER(VAR,WORD,TYPE,I,R,D)                            252
             IF(TYPE.NE.1) GO TO 272                                     253
             COLVAL=I                                                    254
             IF(COLVAL.GT.MAXCOL) GO TO 273                              255
       ENDIF                                                             256
C                                                                        257
C             RETRIEVING AN ARRAY VARAIBLE VALUE                         258
C                                                                        259
                NTYPE=ATYPE                                              260
 240            GO TO (241,242,243,244,245)NTYPE                         261
 241   WRITE(IW,*)VAR,'=',IARAY(ROWVAL,COLVAL,ANUM)                      262
                GO TO 10                                                 263
 242   WRITE(IW,*)VAR,'=',RARAY(ROWVAL,COLVAL,ANUM)                      264
                GO TO 10                                                 265
 243   WRITE(IW,*)VAR,'=',DARAY(ROWVAL,COLVAL,ANUM)                      266
                GO TO 10                                                 267
 244   WRITE(IW,*)VAR,'=',CARAY(ROWVAL,ANUM)                             268
                GO TO 10                                                 269
 245   WRITE(IW,*)VAR,'=',LARAY(ROWVAL,ANUM)                             270
                GO TO 10                                                 271
C                                                                        272
C             RETRIEVING A VARIABLE VALUE                                273
C                                                                        274
 250   DO 270 I=1,5                                                      275
             DO 260 J=1,20                                               276
             IF(VARTBL(J,I).EQ.VAR) THEN                                 277
                    VTYPE=I                                              278
                    INDX=J                                               279
                    GO TO 280                                            280
             ELSE                                                        281
                    CONTINUE                                             282
             ENDIF                                                       283
 260         CONTINUE                                                    284
 270   CONTINUE                                                          285
```

```
 280    GO TO (281,282,283,284,285) VTYPE                                 286
 281    WRITE(IW,*)VAR,'=',IVAL(INDX)                                     287
        GO TO 10                                                          288
 282    WRITE(IW,*)VAR,'=',RVAL(INDX)                                     289
        GO TO 10                                                          290
 283    WRITE(IW,*)VAR,'=',DVAL(INDX)                                     291
        GO TO 10                                                          292
 284    WRITE(IW,*)VAR,'=',CVAL(INDX)                                     293
        GO TO 10                                                          294
 285    WRITE(IW,*)VAR,'=',LVAL(INDX)                                     295
        GO TO 10                                                          296
 272    WRITE(IW,*)'SUBSCRIPTS MUST BE OF TYPE INTEGER'                   297
        GO TO 10                                                          298
 273    WRITE(IW,*)'SUBSCRIPT RANGE OUT OF BOUNDS'                        299
        GO TO 10                                                          300
 274    WRITE(IW,*)'THE NUMBER OR SUBSCRIPTS IN INCONSISTANT'             301
        GO TO 10                                                          302
 275    WRITE(IW,*)'VARIABLE NAME',VAR,'CANNOT BE FOUND'                  303
        GO TO 10                                                          304
C                                                                         305
C              QUIT--RETURN TO ERROR MENU                                 306
C                                                                         307
 290    WRITE(IW,*)'***** VARIABLE NAME ENTERED MUST BE A CHARACTER',     308
     $          ' STRING OF <= TO 6 ELEMENTS *****'                      309
        GO TO 200                                                         310
 300    RETURN                                                            311
 400    FORMAT(A15)                                                       312
 410    FORMAT(1H ,A6,A1,I3,A1,I3,A2,I8)                                  313
 411    FORMAT(1H ,A6,A1,I3,A2,I8)                                        314
 420    FORMAT(1H ,A6,A1,I3,A1,I3,A2,E20.8)                               315
 421    FORMAT(1H ,A6,A1,I3,A2,E20.8)                                     316
 430    FORMAT(1H ,A6,A1,I3,A1,I3,A2,A60)                                 317
 431    FORMAT(1H ,A6,A1,I3,A2,A60)                                       318
 440    FORMAT(1H ,A6,A1,I3,A1,I3,A2,L8)                                  319
 441    FORMAT(1H ,A6,A1,I3,A2,L8)                                        320
        END                                                               321
```

```
C*****************************   SUBROUTINE VARCHG  **********************
C
C       CALLING FORMAT ------  VARCHG ------
C
C       PURPOSE:        TO ALLOW THE USER TO CHANGE VARIABLES
C                       DURING THE INTERPRETATION OF THEIR
C                       PROGRAM
C
C       I/O PARAMETERS:
C
C               INPUT:  ALL INFORMATION NEEDED IS PASTED IN COMMON
C                       BLOCKS
C
C               OUTPUT: THE CHANGING OF VARIABLE VALUES
C
C       OTHER PARAMETERS:
C
C               FOR PARSING:
C
C                       VAR:    CHARACTER STRING OF USER DEFINED
C                               VARIABLE NAME, INCLUDING SUBSCRITPT
C                               VALUES, IF ANY
C                       LOCAT:  A CHARACTER STRING OF THE SUBSCRIPT
C                               VALUES OF THE USER DEFINED ARRAY
C                       VARNAM: CHARACTER STRING OF THE USER DEFINED
C                               ARRAY NAME WITH NO SUBSCRIPTS
C                       VTYPE:  INTEGER VALUE THAT DISCRIBES VARIABLE
C                               TYPE
C                       ATYPE:  INTEGER VALUE THAT DISCRIBES ARRAY TYPE
C                       INDX:   INTEGER VALUE THAT INDEXES WHICH USER
C                               DEFINED VARIABLE IS SOUGHT
C                       ANUM:   INTEGER VALUE THAT INDEXES WHICH USER
C                               DEFINED ARRAY IS SOUGHT
C                       COMMA:  LOCATION OF COMMA IN ARRAY SUBSCRIPT
C                       DIMEN:  STORES VALUE OF THE DIMENSION OF THE
C                               USER DEFINED ARRAY
C                       MAXROW: MAXIMUM NUMBER OF ROWS IN USER DEFINED
C                               ARRAY
C                       MAXCOL: MAXIMUM NUMBER OF COLUMES IN USER
C                               DEFINED ARRAY
C                       COLVAL: THE COLUME SUBSCRIPT OF USER ARRAY
C                       ROWVAL: THE ROW SUBSCRIPT OF USER ARRAY
C                       I,R,D:  PATHWAYS TO RECIEVE INTEGER,REAL AND
C                               DOUBLE PRECISION VALUES FROM SUBROUTINE
C                               NUMBER
C                       WORD:   CHARACTER STORAGE OF ARRAY SUBSCRIPTS
C                       LCHECK,RCHECK:  INTEGER LOCATIONS OF RIGHT AND
C                               LEFT PARENTHESIS
C
C               TO CHANGE VARIABLE:
C
C                       IHOLD,RHOLD,DHOLD,CHOLD,LHOLD:  TEMPERARY
C                               STORAGE OF THE NEW VALUE BEING ENTERED
C                               OF TYPE INTEGER,REAL,DOUBLE PRECISION
C                               CHARACTER OR LOGICAL
C                       ECHO:   THE USERS RESPONSE OF WEATHER THE NEW
C                               VALUE ENTERED IS CORRECT. IT MUST BE A
C                               CHARACTER STRING OF 'YES' OR 'NO'
C
C               ERROR TRAPPING:
C
C                       TYPE:   THE TYPE OF VALUE RETURNED FROM
C                               CALLING SUBROUTINE NUMBER WHEN AN
C                               ARRAY SUBSCRIPT IS SENT.(MUST BE AN
C                               INTEGER)
C
```

```
C               COMMON PARAMETERS:                                              67
C                                                                               68
C                       RSTART: A VARIABLE THATS VALUE INDUCATES                69
C                               WHICH DIRECTION VMAIN IS TO TAKE:               70
C                                   1:  CONTINUE EXECUTION                      71
C                                   2:  RESTART EXECUTION FROM START            72
C                                   3:  EXECUTE DIFFERENT PROGRAM               73
C                                   4:  STOP                                    74
C                                                                               75
C                       VARTBL: A 2-D CHARACTER ARRAY OF ALL USER               76
C                               DEFINED VARIABLE NAMES                          77
C                       CVAL:   A CHARACTER ARRAY OF USER DEFINED               78
C                               CHARACTER VARIABLE VALUES                       79
C                       VINDEX: AN INTEGER ARRAY OF THE NUMBER OF               80
C                               VARIABLES DECLARED IN EACH TYPE                 81
C                       IVAL,RVAL,DVAL,LVAL: ARRAYS OF USER DEFINED             82
C                               VARIABLE VALUES OF TYPE INTEGER,REAL            83
C                               DOUBLE PRECISION AND LOGICAL                    84
C                       ARAYTB: A 2-D CHARACTER ARRAY OF ALL USER               85
C                               DEFINED ARRAY NAMES                             86
C                       CARAY:  A CHARACTER ARRAY OF USER DEFINED               87
C                               CHARACTER ARRAY VALUES                          88
C                       AINDEX: AN INTEGER ARRAY OF THE NUMBER OF               89
C                               ARRAYS DECLARED IN EACH TYPE                    90
C                       AINFO:  AN INTEGER ARRAY THAT STORES THE USER           91
C                               DEFINED ARRAY DIMENSIONS                        92
C                       IARAY,RARAY,DARAY,LARAY: ARRAYS OF USER DEFINED         93
C                               ARRAY VALUES OF TYPE INTEGER,REAL,              94
C                                                                               95
C       ERROR NUMBERS CALLED:                                                   96
C                       NONE                                                    97
C                                                                               98
C       SUBROUTINES CALLED:                                                     99
C                       NUMBER                                                  100
C                                                                               101
C       SUBROUTINES THAT CALL VARCHG:                                           102
C                       SPAUSE                                                  103
C                                                                               104
*********************************************************************************
C                                                                               106
        SUBROUTINE VARCHG                                                       107
        INTEGER BUG,CHOICE,LCHECK,RCHECK,AINDEX,ATYPE,ANUM,DIMEN,               108
     $          MAXROW,MAXCOL,COLVAL,ROWVAL,COMMA,TYPE,I,VTYPE,                 109
     $          INDX,RSTART,VINDEX,IHOLD,IARAY,IVAL,AINFO                       110
        REAL R,RARAY,RVAL                                                       111
        DOUBLE PRECISION D,DHOLD,DARAY,DVAL                                     112
        CHARACTER VAR*15,VARNAM*6,LOCAT*9,ARAYTB*6,                             113
     $          WORD*3,CARAY*60,ECHO*3,VARTBL*6,CVAL*60,CHOLD*60                114
        LOGICAL LHOLD,LVAL,LARAY                                                115
        COMMON/STATE/RSTART                                                     116
        COMMON/ERVAR1/VARTBL(20,5),CVAL(20)                                     117
        COMMON/EINTEG/VINDEX(5),IVAL(20),AINDEX(5),AINFO(10,5,3),               118
     $          IARAY(100,100,10)                                               119
        COMMON/EREAL/RVAL(20),RARAY(100,100,10)                                 120
        COMMON/EDOUBL/DVAL(20),DARAY(100,100,10)                                121
        COMMON/ERVAR3/CARAY(100,5)                                              122
        COMMON/ERVAR5/LVAL(20),LARAY(100,5)                                     123
        COMMON/ERVAR6/ARAYTB(10,5)                                              124
        COMMON/CHECK/BUG                                                        125
C                                                                               126
C               CHOSE DIRECTION FROM PAUSE STATEMENT                            127
C                                                                               128
C                                                                               129
C               CHANGE VARIABLE VALUE                                           130
C                                                                               131
 200    REWIND 3                                                                132
        WRITE(IW,*)'ENTER THE NAME OF VARIABLE YOU WISH TO CHANGE'               133
        READ(IR,610,ERR=1210)VAR                                                134
        LCHECK=INDEX(VAR,'(')                                                   135
        IF(LCHECK.EQ.0) GO TO 250                                               136
C                                                                               137
```

```
C               VARIABLE TO BE CHANGED IS AN ARRAY ELEMENT                138
C                                                                         139
      VARNAM=VAR(1:LCHECK-1)                                              140
      RCHECK=INDEX(VAR,')')                                               141
      LOCAT=VAR(LCHECK+1:RCHECK-1)                                        142
      DO 220 I=1,5                                                        143
              DO 210 J=1,AINDEX(I)                                        144
              IF(ARAYTB(J,I).EQ.VARNAM) THEN                              145
                      ATYPE=I                                             146
                      ANUM=J                                              147
                      GO TO 230                                           148
              ELSE                                                        149
                      CONTINUE                                            150
              ENDIF                                                       151
 210          CONTINUE                                                    152
 220  CONTINUE                                                            153
      WRITE(IW,*)'ARRAY ELEMENT',VAR,'CANNOT BE FOUND'                    154
      GO TO 1000                                                          155
 230  DIMEN=AINFO(ANUM,ATYPE,1)                                           156
      MAXROW=AINFO(ANUM,ATYPE,2)                                          157
      MAXCOL=AINFO(ANUM,ATYPE,3)                                          158
C                                                                         159
C               PARSE EXPRESSION FOR ARRAY ELEMENT IDENTIFIERS            160
C                                                                         161
      COLVAL=1                                                            162
      COMMA=INDEX(LOCAT,',')                                              163
      IF(COMMA.EQ.0)THEN                                                  164
              IF(DIMEN.NE.1) GO TO 1240                                   165
              CALL NUMBER(VAR,LOCAT,TYPE,I,R,D)                           166
              IF(TYPE.NE.1) GO TO 1220                                    167
              ROWVAL=I                                                    168
              IF(ROWVAL.GT.MAXROW) GO TO 1230                             169
      ELSE                                                                170
              IF(DIMEN.NE.2) GO TO 1240                                   171
              IF(ATYPE.EQ.4.OR.ATYPE.EQ.5)GO TO 1300                      172
              WORD=LOCAT(1:COMMA-1)                                       173
              CALL NUMBER(VAR,WORD,TYPE,I,R,D)                            174
              IF(TYPE.NE.1) GO TO 1220                                    175
              ROWVAL=I                                                    176
              IF(ROWVAL.GT.MAXROW) GO TO 1230                             177
              WORD=LOCAT(COMMA+1:RCHECK-1)                                178
              CALL NUMBER(VAR,WORD,TYPE,I,R,D)                            179
              IF(TYPE.NE.1) GO TO 1220                                    180
              COLVAL=I                                                    181
              IF(COLVAL.GT.MAXCOL) GO TO 1230                             182
      ENDIF                                                               183
C                                                                         184
C               RETRIEVING PRESENT VARIABLE VALUE                         185
C                                                                         186
      WRITE(IW,*)'THE PRESENT VALUE IS'                                   187
                GO TO (231,232,233,234,235)ATYPE                          188
 231  WRITE(IW,*)VAR,'=',IARAY(ROWVAL,COLVAL,ANUM)                        189
                GO TO 239                                                 190
 232  WRITE(IW,*)VAR,'=',RARAY(ROWVAL,COLVAL,ANUM)                        191
                GO TO 239                                                 192
 233  WRITE(IW,*)VAR,'=',DARAY(ROWVAL,COLVAL,ANUM)                        193
                GO TO 239                                                 194
 234  WRITE(IW,*)VAR,'=',CARAY(ROWVAL,ANUM)                               195
                GO TO 239                                                 196
 235  WRITE(IW,*)VAR,'=',LARAY(ROWVAL,ANUM)                               197
 239  WRITE(IW,*)'************************************************'       198
 240  WRITE(IW,*)'PLEASE ENTER THE NEW VALUE FOR ',VAR                    199
C                                                                         200
C               READ AND STORE NEW ARRAY VARIABLE VALUE                   201
C                                                                         202
      GO TO (241,242,243,244,245)ATYPE                                    203
 241  READ(IR,*,ERR=1260)IHOLD                                            204
      WRITE(IW,*)'DO YOU WANT',VAR,'=',IHOLD,'(YES/NO)'                   205
      READ(IR,620,ERR=1290)ECHO                                           206
      IF(ECHO.EQ.'NO') GO TO 240                                          207
```

```
      IF(ECHO.EQ.'YES')THEN                                              208
              IARAY(ROWVAL,COLVAL,ANUM)=IHOLD                            209
              GO TO 1000                                                 210
      ELSE                                                               211
              CONTINUE                                                   212
      ENDIF                                                              213
      GO TO 1290                                                         214
242   READ(IR,*,ERR=1260)RHOLD                                           215
      WRITE(IW,*)'DO YOU WANT',VAR,'=',RHOLD,'(YES/NO)'                  216
      READ(IR,620,ERR=1290)ECHO                                          217
      IF(ECHO.EQ.'NO') GO TO 240                                         218
      IF(ECHO.EQ.'YES') THEN                                             219
              RARAY(ROWVAL,COLVAL,ANUM)=RHOLD                            220
              GO TO 1000                                                 221
      ELSE                                                               222
              CONTINUE                                                   223
      ENDIF                                                              224
      GO TO 1290                                                         225
243   READ(IR,*,ERR=1260)DHOLD                                           226
      WRITE(IW,*)'DO YOU WANT',VAR,'=',DHOLD,'(YES/NO)'                  227
      READ(IR,620,ERR=1290)ECHO                                          228
      IF(ECHO.EQ.'NO') GO TO 240                                         229
      IF(ECHO.EQ.'YES') THEN                                             230
              DARAY(ROWVAL,COLVAL,ANUM)=DHOLD                            231
              GO TO 1000                                                 232
      ELSE                                                               233
              CONTINUE                                                   234
      ENDIF                                                              235
      GO TO 1290                                                         236
244   READ(IR,*,ERR=1270)CHOLD                                           237
      WRITE(IW,*)'DO YOU WANT',VAR,'=',CHOLD,'(YES/NO)'                  238
      READ(IR,620,ERR=1290)ECHO                                          239
      IF(ECHO.EQ.'NO') GO TO 240                                         240
      IF(ECHO.EQ.'YES') THEN                                             241
              CARAY(ROWVAL,ANUM)=CHOLD                                   242
              GO TO 1000                                                 243
      ELSE                                                               244
              CONTINUE                                                   245
      ENDIF                                                              246
      GO TO 1290                                                         247
245   READ(IR,*,ERR=1280)LHOLD                                           248
      WRITE(IW,*)'DO YOU WANT',VAR,'=',LHOLD,'(YES,NO)'                  249
      READ(IR,620,ERR=1290)ECHO                                          250
      IF(ECHO.EQ.'NO') GO TO 240                                         251
      IF(ECHO.EQ.'YES') THEN                                             252
              LARAY(ROWVAL,ANUM)=LHOLD                                   253
              GO TO 1000                                                 254
      ELSE                                                               255
              CONTINUE                                                   256
      ENDIF                                                              257
      GO TO 1290                                                         258
C                                                                        259
C             FIND VARAIABLE NAME TO BE CHANGED                          260
C                                                                        261
250   DO 270 I=1,5                                                       262
              DO 260 J=1,20                                              263
              IF(VARTBL(J,I).EQ.VAR) THEN                                264
                      VTYPE=I                                            265
                      INDX=J                                             266
                      GO TO 280                                          267
              ELSE                                                       268
                      CONTINUE                                           269
              ENDIF                                                      270
260           CONTINUE                                                   271
270   CONTINUE                                                           272
280   GO TO (281,282,283,284,285) VTYPE                                  273
281   WRITE(IW,*)VAR,'=',IVAL(INDX)                                      274
      GO TO 289                                                          275
282   WRITE(IW,*)VAR,'=',RVAL(INDX)                                      276
      GO TO 289                                                          277
283   WRITE(IW,*)VAR,'=',DVAL(INDX)                                      278
      GO TO 289                                                          279
```

```
284     WRITE(IW,*)VAR,'=',CVAL(INDX)                                    280
        GO TO 289                                                        281
285     WRITE(IW,*)VAR,'=',LVAL(INDX)                                    282
        GO TO 289                                                        283
289     WRITE(IW,*)'*************************************************'   284
290     WRITE(IW,*)'PLEASE ENTER THE NEW VALUE FOR',VAR                  285
        GO TO (291,292,293,294,295)VTYPE                                 286
291     READ(IR,*,ERR=1260)IHOLD                                         287
        WRITE(IW,*)'DO YOU WANT',VAR,'=',IHOLD,'(YES/NO)'                288
        READ(IR,620,ERR=1291)ECHO                                        289
        IF(ECHO.EQ.'NO')GO TO 290                                        290
        IF(ECHO.EQ.'YES') THEN                                           291
                IVAL(INDX)=IHOLD                                         292
                GO TO 1000                                               293
        ELSE                                                             294
                CONTINUE                                                 295
        ENDIF                                                            296
        GO TO 1291                                                       297
292     READ(IR,*,ERR=1260)RHOLD                                         298
        WRITE(IW,*)'DO YOU WANT',VAR,'=',RHOLD,'(YES/NO)'                299
        READ(IR,620,ERR=1291)ECHO                                        300
        IF(ECHO.EQ.'NO')GO TO 290                                        301
        IF(ECHO.EQ.'YES')THEN                                            302
                RVAL(INDX)=RHOLD                                         303
                GO TO 1000                                               304
        ELSE                                                             305
                CONTINUE                                                 306
        ENDIF                                                            307
        GO TO 1291                                                       308
293     READ(IR,*,ERR=1260)DHOLD                                         309
        WRITE(IW,*)'DO YOU WANT',VAR,'=',DHOLD,'(YES/NO)'                310
        READ(IR,620,ERR=1291)ECHO                                        311
        IF(ECHO.EQ.'NO')GO TO 290                                        312
        IF(ECHO.EQ.'YES')THEN                                            313
                DVAL(INDX)=DHOLD                                         314
                GO TO 1000                                               315
        ELSE                                                             316
                CONTINUE                                                 317
        ENDIF                                                            318
        GO TO 1291                                                       319
294     READ(IR,*,ERR=1270)CHOLD                                         320
        WRITE(IW,*)'DO YOU WANT',VAR,'=',CHOLD,'(YES/NO)'                321
        READ(IR,620,ERR=1291)ECHO                                        322
        IF(ECHO.EQ.'NO')GO TO 290                                        323
        IF(ECHO.EQ.'YES')THEN                                            324
                CVAL(INDX)=CHOLD                                         325
                GO TO 1000                                               326
        ELSE                                                             327
                CONTINUE                                                 328
        ENDIF                                                            329
        GO TO 1291                                                       330
295     READ(IR,*,ERR=1280)LHOLD                                         331
        WRITE(IW,*)'DO YOU WANT',VAR,'=',LHOLD,'(YES/NO)'                332
        READ(IR,620,ERR=1291)ECHO                                        333
        IF(ECHO.EQ.'NO')GO TO 290                                        334
        IF(ECHO.EQ.'YES')THEN                                            335
                LVAL(INDX)=LHOLD                                         336
                GO TO 1000                                               337
        ELSE                                                             338
                CONTINUE                                                 339
        ENDIF                                                            340
        GO TO 1291                                                       341
C                                                                        342
```

```
C               ERROR TRAPING STATEMENTS                                343
C                                                                       344
 1000   RETURN                                                          345
 1210   WRITE(IW,*)'VARIABLE NAME MUST BE A CHARACTER STRING'           346
        GO TO 1000                                                      347
 1220   WRITE(IW,*)'SUBSCRIPTS MUST BE OF TYPE INTEGER'                 348
        GO TO 1000                                                      349
 1230   WRITE(IW,*)'SUBSCRIPT RANGE OUT OF BOUNDS'                      350
        GO TO 1000                                                      351
 1240   WRITE(IW,*)'THE NUMBER OF SUBSCRIPTS IS INCONSISTANT'           352
        GO TO 1000                                                      353
 1250   WRITE(IW,*)'VARIABLE NAME',VAR,'CANNOT BE FOUND'                354
        GO TO 1000                                                      355
 1260   WRITE(IW,*)'VALUE ENTERED UST BE OF TYPE INTEGER,REAL OR',      356
     $                 'DOUBLE PRECISION'                               357
        GO TO 1000                                                      358
 1270   WRITE(IW,*)'VALUE ENTERED MUST BE OF TYPE CHARACTER'            359
        GO TO 1000                                                      360
 1280   WRITE(IW,*)'VALUE ENTERED MUST BE OF TYPE LOGICAL'              361
        GO TO 1000                                                      362
 1290   WRITE(IW,*)'INPUT MUST BE    YES--or--NO'                       363
        GO TO 240                                                       364
 1291   WRITE(IW,*)'INPUT MUST BE    YES--or--NO'                       365
        GO TO 290                                                       366
 1300   WRITE(IW,*)'CHARACTER AND LOGICAL ARRAYS MAY ONLY BE 1-D'       367
        GO TO 1000                                                      368
 610    FORMAT(A15)                                                     369
 620    FORMAT(A3)                                                      370
        END                                                             371
```

```
C*****************************   SUBROUTINE ERROR    ********************      1
C                                                                               2
C       CALLING FORMAT ------  ERROR(LINE,NUM)  ------                          3
C                                                                               4
C       PURPOSE:        TO INFORM USER AN ERROR HAS OCCURED AND ALLOW           5
C                       USER TO DETERMINE WHAT ACTION TO TAKE                   6
C                                                                               7
C       I/O PARAMETERS:                                                         8
C                                                                               9
C               INPUT:                                                         10
C                       LINE:   THE LINE BEING INTERPRETED WHEN AN             11
C                               ERROR WAS DETECTED                             12
C                       NUM:    ERROR NUMBER DISCRIBING THE TYPE OF            13
C                               ERROR                                          14
C                                                                              15
C               OUTPUT: NONE                                                   16
C                                                                              17
C       OTHER PARAMETERS:                                                      18
C                                                                              19
C               TO CREAT MENU:                                                 20
C                                                                              21
C                       ENAME:  CHARACTER STRING OF MENU NAME                  22
C                       ELINE:  CHARACTER ARRAY OF THE CHOICES MENU HAS        23
C                       CHOICE: THE NUMBER OF THE CHOICE THE USER HAS          24
C                               SELECTED FROM THE MENU                         25
C                                                                              26
C               FOR PROGRAM CONTROL:                                           27
C                                                                              28
C                       RSTART: A COMMON VARIABLE COMMON TO SUBROUTINES:       29
C                                       VMAIN                                  30
C                                       ERROR                                  31
C                                       SPAUSE                                 32
C                                       PATHWY                                 33
C                               THIS VARIABLES VALUE IS SET IN ERROR,          34
C                               SPAUES AND PATHWY.  THE VALUE INDUCATES        35
C                               WHICH DIRECTION VMAIN IS TO TAKE               36
C                                                                              37
C                                       1:  CONTINUE EXECUTION                 38
C                                       2:  RESTART EXECUTION FROM START       39
C                                       3.  EXECUTE DIFFERENT PROGRAM          40
C                                       4.  STOP                               41
C                                                                              42
C       ERROR NUMBERS CALLED:                                                  43
C                       NONE                                                   44
C                                                                              45
C       SUBROUTINES CALLED:                                                    46
C                       MENU                                                   47
C                       VAROUT                                                 48
C                       SEDIT                                                  49
C                       PATHWY                                                 50
C                                                                              51
C       SUBROUTINES THAT CALL ERROR:                                           52
C                       ERROR IS CALLED FROM THOUGHOUT THE PROGRAM             53
C                                                                              54
*******************************************************************************55
        SUBROUTINE ERROR(LINE,NUM)                                             56
        INTEGER NUM,BUG,CHOICE,RSTART                                          57
```

```
      CHARACTER LINE*72,ELINE*72,ENAME*20                               58
      DIMENSION ELINE(8)                                                59
      COMMON/STATE/RSTART                                               60
      COMMON/CHECK/BUG                                                  61
      ENAME='ERROR                                                      62
      ELINE(1)='DISPLAY VARIABLE VALUES'                                63
      ELINE(2)='EDIT TEXT'                                              64
      ELINE(3)='RESTART EXECUTION FROM BEGINNING'                       65
      ELINE(4)='QUIT OR EXECUTE DIFFERENT PROGRAM'                      66
      WRITE(3,*)'***ERROR*** #',NUM                                     67
      WRITE(3,5)LINE                                                    68
C                                                                       69
C             ERROR MENU                                                70
C                                                                       71
 100  CALL MENU(ENAME,ELINE,4,CHOICE)                                   72
      GO TO (1000,2000,3000,4000)CHOICE                                 73
C                                                                       74
C             VIEW VARIABLES                                            75
C                                                                       76
 1000 CALL VAROUT                                                       77
      REWIND 3                                                          78
      GO TO 100                                                         79
C                                                                       80
C             EDIT TEXT                                                 81
C                                                                       82
 2000 CALL SEDIT                                                        83
      REWIND 3                                                          84
      GO TO 100                                                         85
C                                                                       86
C             REINTERPRETE PROGRAM FROM BEGINNING                       87
C                                                                       88
 3000 RSTART=2                                                          89
      REWIND 100                                                        90
      GO TO 1500                                                        91
C                                                                       92
C             GO TO PATHWAY STATE                                       93
C                                                                       94
 4000 CALL PATHWY                                                       95
 1500 RETURN                                                            96
  5   FORMAT(1H ,A72)                                                   97
      END                                                               98
```

```
C*****************************   SUBROUTINE SEDIT   **********************
C
C       CALLING FORMAT ------  SEDIT  -----
C
C       PURPOSE:        TO GIVE A USER THE MEANS TO EDIT TEXT OF THE
C                       PROGRAM BEING INTERPRETED
C
C       I/O PARAMETERS:
C
C               INPUT:  ALL INFORMATION NEEDED IS PASTED IN COMMON
C                       BLOCKS
C
C               OUTPUT: REWRITING THE USERS FILE WITH THE EDITED TEXT
C
C       OTHER PARAMETERS:
C
C               TO CREAT MENU:
C
C                       DNAME:  CHARACTER STRING OF MENU NAME
C                       EDLINE:  CHARCTER ARRAY OF THE CHOICES MENU HAS
C                       CHOICE: THE NUMBER OF THE CHOICE THE USER HAS
C                               SELECTED FROM THE MENU
C
C               USED TO EDIT:
C
C                       SAVE:   A TEMPORARY CHARACTER STORAGE LOCATION
C                               USED TO HOLD A PORTIONS OF TEXT DURING
C                               THE EDITING PROCESS
C                       LINUM:  THE LINE NUMBER TO BE INSERTED AFTER
C                               DELETED OR CHANGED
C
C               FOR DISPLAYING TEXT:
C
C                       ERORPT: THE LINE NUMBER OF THE LINE THE ERROR IS
C                               FOUND IN
C                       LASTLN: THE LINE NUMBER OF THE MOST ADVANCED THE
C                               PROGRAM HAS BEEN INTERPRETED
C                       A,B:    THE BOUNDARIES OF THE TEXT TO BE PRINTED
C                       TEST:   VARIABLE USE TO SET BOUNDARIES
C
C               COMMON PARAMETERS:
C
C                       LINDEX: THE LINE NUMBER THAT IS PRESENTLY
C                               BEENING INTERPRETED
C                       LPOINT: THE LINE NUMBER OF THE MOST ADVANCED
C                               THE PROGRAM HAS BEEN INTERPRETED
C                       MAXLIN: THE MAXIMUM NUMBER OF LINES IN THE
C                               USERS PROGRAM
C
C       ERROR NUMBERS CALLED:
C                       NONE
C
C       SUBROUTINES CALLED:
C                       MENU
C
C       SUBROUTINE THAT CALL SEDIT:
C                       ERROR
C
C*******************************************************************************
C
        SUBROUTINE SEDIT
        INTEGER CHOICE,LINUM,LASTLN,TEST,A,B,LINDEX,MAXLIN,BUG,ERORPT,
     $                      LEVEL
        CHARACTER DNAME*20,EDLINE*72,SAVE*72,SCLINE*72,CHECK*72
        DIMENSION EDLINE(8),SAVE(50),CHECK(50)
        COMMON/CONTRL/LINDEX,LPOINT,LEVEL,MAXLIN
        COMMON/CONTR2/SCLINE(50)
        COMMON/CHECK/BUG
        DNAME='EDIT
```

```
      EDLINE(1)='DISPLAY DIFFERENT TEXT'                                70
      EDLINE(2)='INSERT A LINE'                                         71
      EDLINE(3)='DELETE A LINE'                                         72
      EDLINE(4)='CHANGE A LINE'                                         73
      EDLINE(5)='QUIT (RETURN TO ERROR MENU)'                           74
      LASTLN=LINDEX                                                     75
      ERORPT=LPOINT                                                     76
C                                                                       77
C             DISPLAY TEXT AROUND ERROR                                 78
C                                                                       79
      REWIND 3                                                          80
      IF(ERORPT.LE.5)THEN                                               81
            A=1                                                         82
      ELSE                                                              83
            A=ERORPT-5                                                  84
      ENDIF                                                             85
      TEST=LASTLN-ERORPT                                                86
      IF(TEST.LT.5)THEN                                                 87
            B=LASTLN                                                    88
      ELSE                                                              89
            B=ERORPT+5                                                  90
      ENDIF                                                             91
      DO 10 I=A,B                                                       92
            IF(ERORPT.EQ.I)THEN                                         93
                  WRITE(IW,165)'==>',I,SCLINE(I)                        94
            ELSE                                                        95
                  WRITE(IW,160)I,SCLINE(I)                              96
            ENDIF                                                       97
 10   CONTINUE                                                          98
 50   CALL MENU(DNAME,EDLINE,5,CHOICE)                                  99
      WRITE(IW,*)                                                      100
      GO TO (100,200,300,400,500) CHOICE                               101
C                                                                      102
C        ***  DISPLAY DIFFERENT TEXT  ***                              103
C                                                                      104
 100  WRITE(IW,*)'ENTER LINE NUMBER YOU WISH TO BEGIN DISPLAY'          105
      READ(IR,*,ERR=450)LINUM                                          106
      IF(LINUM.GE.LASTLN)THEN                                          107
        WRITE(IW,*)'YOU CANNOT DISPLAY LINES THE HAVE NOT BEEN',       108
     $             'INTERPUTED'                                        109
        GO TO 100                                                      110
      ELSE                                                             111
        CONTINUE                                                       112
      ENDIF                                                            113
      IF(LINUM.LT.1)THEN                                               114
            WRITE(IW,*)'LINE NUMBER',LINUM,'IS NOT RECONIZED'          115
            GO TO 100                                                  116
      ELSE                                                             117
            CONTINUE                                                   118
      ENDIF                                                            119
      TEST=LASTLN-LINUM                                                120
      IF(TEST.LE.10)THEN                                               121
            B=LASTLN                                                   122
      ELSE                                                             123
            B=LINUM+10                                                 124
      ENDIF                                                            125
      REWIND 3                                                         126
      DO 110 I=LINUM,B                                                 127
            IF(I.EQ.ERORPT)THEN                                        128
                  WRITE(IW,165)'==>',I,SCLINE(I)                       129
            ELSE                                                       130
                  WRITE(IW,160)I,SCLINE(I)                             131
            ENDIF                                                      132
 110  CONTINUE                                                         133
      GO TO 50                                                         134
C                                                                      135
```

```
C          ***  INSERT A LINE  ***                                      136
C                                                                       137
 200    WRITE(IW,*)'ENTER LINE NUMBER YOU WISH TO INSERT AFTER'         138
        READ(IR,*,ERR=460)LINUM                                         139
        IF(LINUM.GE.LASTLN) THEN                                        140
               WRITE(IW,*)'YOU CANNOT INSERT A LINE AFTER A LINE',      141
     $                          'THAT HAS NOT BEEN INTERPUTED'          142
               GO TO 200                                                143
        ELSE                                                            144
               CONTINUE                                                 145
        ENDIF                                                           146
        IF(LINUM.LT.1)THEN                                              147
               WRITE(IW,*)'LINE NUMBER',LINUM,'IS NOT RECONIZED'        148
               GO TO 200                                                149
        ELSE                                                            150
               CONTINUE                                                 151
        ENDIF                                                           152
C                                                                       153
C              STORE TEXT BEING EDITED                                  154
C                                                                       155
        IF(LINUM.LT.ERORPT)ERORPT=ERORPT+1                              156
        DO 210 I=1,LINUM                                                157
           SAVE(I)=SCLINE(I)                                            158
 210    CONTINUE                                                        159
 215    WRITE(IW,*)'ENTER TEXT OF LINE TO BE INSERTED'                  160
        READ(IR,150,ERR=470)SAVE(LINUM+1)                               161
        LASTLN=LASTLN+1                                                 162
        DO 220 I=LINUM+1,LASTLN                                         163
               SAVE(I+1)=SCLINE(I)                                      164
 220    CONTINUE                                                        165
        DO 230 I=1,LASTLN                                               166
               SCLINE(I)=SAVE(I)                                        167
 230    CONTINUE                                                        168
C                                                                       169
C              DISPLAY EDITED TEXT                                      170
C                                                                       171
        IF(LINUM.LE.5) THEN                                             172
               A=1                                                      173
        ELSE                                                            174
               A=LINUM-5                                                175
        ENDIF                                                           176
        TEST=LASTLN-LINUM                                               177
        IF(TEST.LE.5) THEN                                              178
               B=LASTLN                                                 179
        ELSE                                                            180
               B=LINUM+5                                                181
        ENDIF                                                           182
        REWIND 3                                                        183
        DO 240 I=A,B                                                    184
               IF(I.EQ.ERORPT)THEN                                      185
                      WRITE(IW,165)'==>',I,SCLINE(I)                    186
               ELSE                                                     187
                      WRITE(IW,160)I,SCLINE(I)                          188
               ENDIF                                                    189
 240    CONTINUE                                                        190
        GO TO 50                                                        191
C                                                                       192
C          ***  DELETE A LINE  ***                                      193
C                                                                       194
 300    WRITE(IW,*)'ENTER LINE NUMBER YOU WISH TO DELETE'               195
        READ(IR,*,ERR=480)LINUM                                         196
        IF(LINUM.GE.LASTLN) THEN                                        197
               WRITE(IW,*)'YOU CANNOT DELETE A LINE',                   198
     $                          'THAT HAS NOT BEEN INTERPUTED'          199
               GO TO 300                                                200
        ELSE                                                            201
               CONTINUE                                                 202
        ENDIF                                                           203
        IF(LINUM.LT.1)THEN                                              204
               WRITE(IW,*)'LINE NUMBER',LINUM,'IS NOT RECONIZED'        205
               GO TO 300                                                206
        ELSE                                                            207
```

```
              CONTINUE                                                  208
        ENDIF                                                           209
C                                                                       210
C               STORE TEXT BEING EDITED                                 211
C                                                                       212
        LASTLN=LASTLN-1                                                 213
        IF(LINUM.LT.ERORPT)ERORPT=ERORPT-1                              214
        DO 310 I=1,LINUM-1                                              215
              SAVE(I)=SCLINE(I)                                         216
 310    CONTINUE                                                        217
        DO 320 I=LINUM,LASTLN                                           218
              SAVE(I)=SCLINE(I+1)                                       219
 320    CONTINUE                                                        220
        DO 330 I=1,LASTLN                                               221
              SCLINE(I)=SAVE(I)                                         222
 330    CONTINUE                                                        223
C                                                                       224
C               DISPLAY EDITED TEXT                                     225
C                                                                       226
        IF(LINUM.LE.5) THEN                                             227
              A=1                                                       228
        ELSE                                                            229
              A=LINUM-5                                                 230
        ENDIF                                                           231
        TEST=LASTLN-LINUM                                               232
        IF(TEST.LE.5) THEN                                              233
              B=LASTLN                                                  234
        ELSE                                                            235
              B=LINUM+5                                                 236
        ENDIF                                                           237
        REWIND 3                                                        238
        DO 340 I=A,B                                                    239
              IF(I.EQ.ERORPT)THEN                                       240
                    WRITE(IW,165)'==>',I,SCLINE(I)                      241
              ELSE                                                      242
                    WRITE(IW,160)I,SCLINE(I)                            243
              ENDIF                                                     244
 340    CONTINUE                                                        245
        GO TO 50                                                        246
C                                                                       247
C         ***   CHANGE A LINE  ***                                      248
C                                                                       249
 400    WRITE(IW,*)'ENTER LINE NUMBER OF THE LINE YOU WISH TO',         250
     $                ' CHANGE'                                         251
        READ(IR,*,ERR=490)LINUM                                         252
        IF(LINUM.GT.LASTLN)THEN                                         253
              WRITE(IW,*)'YOU CANNOT EDIT A LINE WHICH HAS NOT',        254
     $                        ' BEEN INTERPUTED'                        255
              GO TO 400                                                 256
        ELSE                                                            257
              CONTINUE                                                  258
        ENDIF                                                           259
        IF(LINUM.LT.1)THEN                                              260
              WRITE(IW,*)'LINE NUMBER',LINUM,'IS NOT RECONIZED'         261
              GO TO 400                                                 262
        ELSE                                                            263
              CONTINUE                                                  264
        ENDIF                                                           265
 405    WRITE(IW,*)'ENTER LINE AS DESIRED'                              266
        READ(IR,150,ERR=495)SCLINE(LINUM)                               267
```

```
C                                                                       268
C               DISPLAY EDITED TEXT                                     269
C                                                                       270
      IF(LINUM.LE.5)THEN                                                271
            A=1                                                         272
      ELSE                                                              273
            A=LINUM-5                                                   274
      ENDIF                                                             275
      TEST=LASTLN-LINUM                                                 276
      IF(TEST.LE.5)THEN                                                 277
            B=LASTLN                                                    278
      ELSE                                                              279
            B=LINUM+5                                                   280
      ENDIF                                                             281
      REWIND 3                                                          282
      DO 410 I=A,B                                                      283
            IF(I.EQ.LPOINT)THEN                                         284
                  WRITE(IW,165)'==>',I,SCLINE(I)                        285
            ELSE                                                        286
                  WRITE(IW,160)I,SCLINE(I)                              287
            ENDIF                                                       288
 410  CONTINUE                                                          289
      GO TO 50                                                          290
C                                                                       291
C               READ PROTECT ERROR MESSAGES                             292
C                                                                       293
 450  WRITE(IW,*)'**** THE NUMBER ENTERED MUST BE OF TYPE',             294
     $                  ' INTEGER ****'                                 295
      GO TO 100                                                         296
 460  WRITE(IW,*)'**** THE NUMBER ENTERED MUST BE OF TYPE',             297
     $                  ' INTEGER ****'                                 298
      GO TO 200                                                         299
 470  WRITE(IW,*)'**** LINE ENTERED MUST BE A CHARACTER STRING',        300
     $                  ' LESS THEN 72 ELEMENTS LONG ****'              301
      GO TO 215                                                         302
 480  WRITE(IW,*)'**** THE NUMBER ENTERED MUST BE OF TYPE',             303
     $                  ' INTEGER ****'                                 304
      GO TO 300                                                         305
 490  WRITE(IW,*)'**** THE NUMBER ENTERED MUST BE OF TYPE',             306
     $                  ' INTEGER ****'                                 307
      GO TO 400                                                         308
 495  WRITE(IW,*)'**** LINE ENTERED MUST BE A CHARACTER STRING',        309
     $                  ' LESS THEN 72 ELEMENTS LONG ****'              310
      GO TO 405                                                         311
C                                                                       312
C          ***  REWRITE USERS FILE AND QUIT  ***                       313
C                                                                       314
 500  DO 600 I=LASTLN+1,MAXLIN                                          315
           READ(100,150,END=605)SCLINE(I)                               316
 600  CONTINUE                                                          317
 605  REWIND 100                                                        318
      DO 610 I=1,MAXLIN                                                 319
           WRITE(100,150)SCLINE(I)                                      320
 610  CONTINUE                                                          321
 1000 RETURN                                                            322
 150  FORMAT(A72)                                                       323
 160  FORMAT(1H ,3X,I3,A72)                                             324
 165  FORMAT(1H ,A3,I3,A72)                                             325
      END                                                               326
```

```
C*****************************   SUBROUTINE PATHWY   ********************
C
C       CALLING FORMAT ------  PATHWY  ------
C
C       PURPOSE:        TO ALLOW USER THE CHOICE OF REXECUTING THE
C                       PRESENT PROGRAM OR A DIFFERENT PROGRAM OR
C                       TO LEAVE INTERP
C
C       I/O PARAMETERS:
C
C               NONE
C
C       OTHER PARAMETERS:
C
C               TO CREAT MENU:
C
C                       PWNAME:  CHARACTER STRING OF MENU NAME
C                       PWLINE:  CHARACTER ARRAY OF THE CHOICES MENU HAS
C                       CHOICE: THE NUMBER OF THE CHOICE THE USER HAS
C                               SELECTED FROM THE MENU
C
C               FOR PROGRAM CONTROL:
C
C                       RSTART: A COMMON VARIABLE COMMON TO SUBROUTINES:
C                                       VMAIN
C                                       ERROR
C                                       SPAUSE
C                                       PATHWY
C                               THIS VARIABLES VALUE IS SET IN ERROR,
C                               SPAUES AND PATHWY.  THE VALUE INDUCATES
C                               WHICH DIRECTION VMAIN IS TO TAKE
C
C                                       1:  CONTINUE EXECUTION
C                                       2:  RESTART EXECUTION FROM START
C                                       3.  EXECUTE DIFFERENT PROGRAM
C                                       4.  STOP
C
C       ERROR NUMBERS CALLED:
C                       NONE
C
C       SUBROUTINES CALLED:
C                       MENU
C
C       SUBROUTINES THAT CALL PATHWY:
C                       VMAIN
C                       SPAUSE
C                       ERROR
C
*************************************************************************
        SUBROUTINE PATHWY
        INTEGER RSTART,CHOICE,BUG
        CHARACTER PWNAME*20,PWLINE*72
        DIMENSION PWLINE(8)
        COMMON/STATE/RSTART
        COMMON/CHECK/BUG
        PWNAME='EXECUTION CHOICE'
        PWLINE(1)='REXECUTE PROGRAM'
```

```
      PWLINE(2)='EXECUTE A DIFFERENT PROGRAM'                          58
      PWLINE(3)='QUIT WORKING'                                         59
      REWIND 3                                                         60
C                                                                      61
C             DISPLAY MENU                                             62
C                                                                      63
 10   CALL MENU(PWNAME,PWLINE,3,CHOICE)                               64
      GO TO (100,200,300)CHOICE                                        65
C                                                                      66
C             REINTERPRETE PROGRAM                                     67
C                                                                      68
 100  RSTART=2                                                         69
      REWIND 100                                                       70
      GO TO 1000                                                       71
C                                                                      72
C             INTERPRETE DIFFERENT PROGRAM                             73
C                                                                      74
 200  CLOSE(UNIT=100)                                                  75
      RSTART=3                                                         76
      GO TO 1000                                                       77
C                                                                      78
C             COMPLETE EXECUTION OF INTERP                             79
 300  CLOSE(UNIT=100)                                                  80
      RSTART=4                                                         81
      GO TO 1000                                                       82
 1000 RETURN                                                           83
 150  FORMAT(A10)                                                      84
      END                                                              85
```

```
C*****************************   SUBROUTINE MENU   **********************
C
C       CALLING FORMAT ------  MENU(NAME,LINES,NUM,CHOICE)  -----
C
C       PURPOSE:        TO DISPLAY A MENU OF CHOICES TO THE USER AND
C                       TO CHECK THE CHOICE FOR VALIDITY
C
C       I/O PARAMETERS:
C
C               INPUT:
C                       NAME:   CHARACTER STRING OF MENU NAME
C                       LINES:  CHARACTER ARRAY OF THE CHOICES MENU
C                               HAS
C                       NUMBER: THE NUMBER OF CHOICES ALLOWED
C
C               OUTPUT:
C                       CHOICE: THE NUMBER OF THE CHOICE THE USER
C                               SELECTED
C       OTHER PARAMETERS:
C                       NONE
C
C       ERROR NUMBERS CALLED:
C                       NONE
C
C       SUBROUTINES CALLED:
C                       NONE
C
C       SUBROUTINES THAT CALL MENU:
C                       SPAUSE
C                       VAROUT
C                       ERROR
C                       PATHWY
C                       SEDIT
C
*******************************************************************************
        SUBROUTINE MENU(NAME,LINES,NUM,CHOICE)
        INTEGER NUM,CHOICE,BUG
        CHARACTER NAME*20,LINES*72
        DIMENSION LINES(8)
        COMMON/CHECK/BUG
C
C               DISPLAY MENU
C
        WRITE(IW,*)
        WRITE(IW,*)NAME,'MENU : '
        WRITE(IW,*)'**************************************************'
        WRITE(IW,*)
        DO 100 I=1,NUM
                WRITE(IW,300)I,LINES(I)
 100    CONTINUE
C
C               READ CHOICE AND ASSURE VALIDITY
C
 200    WRITE(IW,*)
        WRITE(IW,*)'PLEASE ENTER THE NUMBER OF YOU CHOICE'
        READ(IR,*,ERR=250)CHOICE
        IF(CHOICE.LT.1.OR.CHOICE.GT.NUM)THEN
           WRITE(IW,*)'NUMBER',CHOICE,'IS NOT RECONIZED AS A CHOICE'
           GO TO 200
        ELSE
           CONTINUE
        ENDIF
        GO TO 270
 250    WRITE(IW,*)'***** DATA ENTERED MUST BE OF TYPE INTEGER *****'
        GO TO 200
 270    RETURN
 300    FORMAT(1H ,I2,3X,A40)
        END
```

APPENDIX C

ERROR NUMBERS AND MESSAGES

ERROR NUMBERS AND MESSAGES

SYNTAX ERRORS

101 missing a parenthesis
102 input or output device is not there (number must be an integer)
103 READ and WRITE statements must be free format
104 error in quotes in WRITE statement
105 error in READ or WRITE statement
106 syntax error in DO statement
107 syntax error in computed GOTO statement
108 to few line numbers in computed GOTO
109 syntax error in IF statement
110 logical operator of IF statement not recognizable
111 input data not acceptable in READ statement
112 error in quotes
113 logical assignment error or no zero before decimal
114 error in data
115 missing quote on character string to be written
116 missing parenthesis
117 syntax error in arithmetic statement
150 error in INTERP program (call professor)

FUNCTION ERRORS

201 function cannot be performed on an integer
202 for arcsin/arccos the value to be evaluated must have its absolute value less then one
203 cannot take square root of a negative number
204 do loop parameter must be of type integer or real
205 foot statement of DO loop cannot be a GOTO
206 foot statement of DO loop cannot be a RETURN
207 foot statement of DO loop cannot be a END
208 foot statement of DO loop cannot be a STOP
209 illegal nesting of DO loop
210 illegal structure of IF and DO statements
211 foot statement of DO loop cannot be an IF
212 attempt to divided by zero

NUMBERING ERRORS

301 no digit preceeding a decimal
302 character value in numeric expression
303 line number must be of type integer
304 array subscripts must be if type integer
305 a character or logical value cannot be assigned to a integer, real or double precision variable

PROGRAM STRUCTURE

401 statement is not recognizable (no key words)
402 GOTO statement trying to make illegal entry into DO structure
403 GOTO statement trying to make illegal entry into IF statement
404 one of the following is missing:
    statement number used by GOTO
    ELSE statement
    ENDIF statement
405 improper IF structure
406 end of file found with no END or STOP statement
407 nesting of IF cannot exceed 5 levels

DECLARATION ERRORS

501 variable name to long
502 PRECISION must follow the word DOUBLE
503 variable has not been declared
504 array variable has no been declared
505 number of array subscripts are inconsistent
506 array subscript exceeds array dimension
507 error in dimension statement
508 logical and character arrays must be one dimensional
509 variable in dimension statement with no subscripts
510 only 20 variable names allowed per type

VITA

Susan Jane Zimmer

Candidate for the Degree of

Masters of Science

Thesis: THE DEVELOPMENT OF A FORTRAN 77 INTERPRETER

Major Field: Mechanical Engineering

Biographical:

Personal Data: Born in Sturgis, Michigan, April 8, 1957, the daughter of Harry and Clare Laws. Married to Mark A. Zimmer, D.V.M. on August 5, 1978. First child, Jay Andrew Zimmer, born on July 26, 1983, second child, Jamie Sue Zimmer, born on May 26, 1985.

Education: Graduated from Bronson High School, Bronson, Michigan, in May 1975; completed certificate in Animal Technology at Michigan State University in April 1977; received Bachelor of Science Degree in Mechanical Engineering from Oklahoma State University in December 1983; completed requirements for the Master of Science degree at Oklahoma State University in July, 1986.

Professional Experience: Teaching Assistant, Department of Mechanical Engineering, Oklahoma State University, Spring 1983, Spring 1984, Summer 1984, Fall 1984; Member Eta Kappa Nu beginning Fall 1981; Member Pi Tau Sigma beginning Spring 1982, president Spring 1983; Member National Society of Professional Engineers beginning 1982; Passed engineering intern examination October 1982.